Robust visual data segmentation: Sampling from distribution of model parameters

Sadri, Alireza; Tennakoon, Ruwan; Hoseinnezhad, Reza; Bab-Hadiashar, Alireza

Sadri, A., Tennakoon, R., Hoseinnezhad, R., & Bab-Hadiashar, A. (2018). Robust visual data segmentation: Sampling from distribution of model parameters. Computer Vision and Image Understanding, 174, 82–94. https://doi.org/10.1016/j.cviu.2018.07.001
Document Version: Submitted Version

Published Version: https://doi.org/10.1016/j.cviu.2018.07.001

Computer Vision and Image Understanding
journal homepage: www.elsevier.com

Robust visual data segmentation: Sampling from distribution of model parameters

Alireza Sadri[a,**], Ruwan Tennakoon[1], Reza Hosseinnezhad[1], Alireza Bab-Hadiashar[1]

[a]*Melbourne Royal Institute of Technology, Melbourne, Australia*

ABSTRACT

This paper approaches the problem of geometric multi-model fitting as a data segmentation problem. The proposed solution is based on a sequence of sampling hyperedges from a hypergraph, model selection and hypergraph clustering steps. We developed a sampling method that significantly facilitates solving the segmentation problem using a new form of the Markov-Chain-Monte-Carlo (MCMC) method to effectively sample from hyperedge distribution. To sample from this distribution effectively, our proposed Markov Chain includes new ways of long and short jumps to perform exploration and exploitation of all structures. To enhance the quality of samples, a greedy algorithm is used to exploit nearby structure based on the minimization of the Least k^{th} Order Statistics cost function. Unlike common sampling methods, ours does not require any specific prior knowledge about the distribution of models. The output set of samples leads to a clustering solution by which the final model parameters for each segment are obtained. The method competes favorably with the state-of-the-art both in terms of computation power and segmentation accuracy.

1. Introduction

Visual data segmentation is the task of partitioning data points (in presence of noise and outliers) into segments such that based on a specific measure, points within each segment are more similar to each other than points from different segments. Popular segmentation methods such as Normalized cut (*NCut*) (Shi and Malik (2000)) take a graph theocratic approach to solve this problem. In this approach, a weighted graph $G = (V, E)$ is defined where a vertex $v \in V$ represents a data point and an edge $e \in E$ represents the affinity between two data points. In a weighted graph each edge is associated with an element in a square symmetric adjacency matrix $A = (a_{ij})$, $A \in \mathbb{R}^{N\times N}$ where a_{ij} corresponds to the similarity between data points i and j. The graph is then partitioned into disjoint sub graphs to obtain the final labeling solution.

Defining a pair-wise similarity remains a challenge in computer vision problems where a segment (or structure) is defined by an underlying parametrized mathematical constraint with more than one parameter. Such problems include: identifying multiple independently moving objects in a scene using point correspondences in two views (Torr and Murray (1997)) or in multiple views (Tron and Vidal (2007); Ochs et al. (2014); Vidal (2011)). This list extends to include identifying planes in 3D point clouds (Bab-Hadiashar and Gheissari (2006)), detecting homographies (Tat-Jun et al. (2012); Rao et al. (2010)) and illumination invariance face clustering (Georghiades et al. (2001)). To solve the above robust geometric model fitting problems, it is desired to define a similarity between a subset of points that is larger than two. For example, in identifying multiple planes in a 3D point cloud, any two points will perfectly fit infinite number of planes irrespective of their underlying structure, hence a similarity cannot be derived by just using two points. In such cases, an effective similarity measure has to be devised using higher order affinities. For example, in the above problem, least square error between four or more points will provide a suitable affinity measure that indicates how well those points represent a plane.

Relationships between larger than two data points is usually modeled using a hypergraph. Following (Agarwal et al. (2006)) a hypergraph is defined as $\mathcal{H} = (\mathcal{V}, \mathcal{E})$ where $\mathcal{V}$ and $\mathcal{E}$ are the set of vertices and hyperedges, respectively. A hyperedge $e \in \mathcal{E}$ consists of a subset of the vertices $v_e \subseteq \mathcal{V}$ and in a weighted hypergraph each hyperedge e is also assigned a weight $w(e)$. A hypergraph with all of its hyperedges having cardinality ρ

**Corresponding author: Tel.: +64-406-240-796;
e-mail: Alireza.Sadri@rmit.edu.au (Alireza Sadri)

($\forall e \in \mathcal{E} : |e| = \rho$) is called a ρ-uniform hypergraph. Once the hypergraph is constructed, the *NCut* method presented in (Agarwal et al. (2006)) can be used to partition the whole construct into disjoint segments.

To construct a full hypergraph, one needs to consider all the possible hyperedges which is in order of $O(2^N)$ where $N = |\mathcal{V}|$. Even in a ρ-uniform hypergraph with a small ρ, the number of edges grow exponentially with the number of data points (i.e. for a problem with N data points, number of hyperedges is $|\mathcal{E}| = \binom{N}{\rho}$). As such, building a full hypergraph is not practical in real computer vision problems and needs to be approximated.

In a series of theoretical studies (Agarwal et al. (2005); Zhou et al. (2006)), it has been argued that accurate segmentation can be achieved by using approximate hypergraphs constructed by sampling only a subset of hyperedges. To take advantage of this, several methods including (Chen and Lerman (2009b); Liu and Yan (2012); Jain and Govindu (2013)) had proposed to use random sampling for the construction of approximate hypergraphs. However none of these works provides theoretical justification for the effectiveness of the random sampling based methods.

More recently, the effect of sampling on hypergraph clustering has been studied (Florescu and Perkins (2016); Ghoshdastidar and Dukkipati (2017)) and a theoretical lower bound on the number of sampled hyperedges required to achieve a desired error rate has been established (Theorem 9 in (Ghoshdastidar and Dukkipati (2017))). They also compared the effect of uniform sampling with distribution based sampling on constructing sparse hypergraphs. The segmentation performance plateaus for number of samples beyond $O(N^\rho)$ for uniform sampling, while using a carefully selected sampling distribution, such as the one shown in equation (1) provides similar accuracy with only $O(N(\ln N)^2)$ sampled edges.

$$P_{\mathcal{E}}(e) = \frac{w_e}{\sum_{e' \in \mathcal{E}} w_{e'}} \qquad \text{,for all } e \in \mathcal{E} \tag{1}$$

One issue with sampling from this distribution is that calculating the denominator requires knowing all the edge weights. To overcome this issue, an iterative sampling strategy has been proposed (Ghoshdastidar and Dukkipati (2017)). The difficulty with such a scheme is that there is no clear mechanism to rectify past mistakes in sampling. In this paper, we will discuss the possibility of sampling efficiently, from a distribution with similar properties to equation (1), using Markov-Chain-Monte-Carlo (MCMC) sampling method. This technique complements Spectral Clustering on hypergraphs and together they form a relatively accurate and fast data segmentation method. Our contributions are threefold:

- A novel MCMC sampler that samples edges from a distribution resembling the distribution in equation (1).
- Effective mode jumping mechanism for edge sampling in robust model fitting that prevents the sampler from getting trapped in one mode (a single structure in data).
- Practical method for the inclusion of structures' size in the sampling process.

The remainder of paper is organized as follows. Section 2 discusses the prior works that is most relevant to the proposed method. In section 3 the proposed sampling algorithm is introduced. In section 4 we provide the experimental results of the data segmentation method over some well-known datasets and compare those with the state of the art methods. Section 5 concludes the paper.

2. Background

In geometric constraint based data segmentation (also referred to as robust model fitting), the intention is to cluster N data points, $X = \left[x_j\right]_{j=1,\dots,N} \in \mathbb{R}^d$ into C clusters, such that points within clusters are related to each other by a set of models parametrized by $\Theta = \left\{\theta^{(i)}\right\}_{i=1}^{C}$; $\theta^{(i)} \in \mathbb{R}^\Omega$. The number of data points in each groups is $\left\{\hat{k}^{(i)}\right\}_{j=1}^{C}$; $\hat{k}^{(i)} \in \mathbb{R}$. In this section we first provide an overview of different segmentation approaches followed by how the problem can be formulated as a hypergraph clustering problem. The methods to solve hypergraph clustering are also reviewed and the importance of edge sampling is discussed.

2.1. An overview of related robust model fitting approaches

A traditional approach to perform data segmentation is the fit-and-remove strategy (such as in Sequential RANSAC (Vincent and Laganiére (2001)) in which RANSAC (Fischler and Bolles (1981)) is used to find and remove structures, sequentially. It is well known that mistakes made in early removals within most of sequential methods can affect the remaining structures.

The sequential dependency issue of the fit-and-remove approach has been tackled by energy minimization as well as data clustering methods. Energy minimization methods such as those presented in (Boykov et al. (2001)), (Yu et al. (2011)) and (Delong et al. (2012)), define and minimize an energy function that consists of a data fidelity term as well as a model complexity term. By proper tuning of the parameters used for combining these factors, the outcomes of energy minimization methods can be very accurate. However, in this paper, we chose hypergraph clustering method due to its relatively high speed and accuracy as suggested by (Balakrishnan et al. (2011)).

Many subspace learning and clustering methods also share the above issue and are sensitive to the choice of regularization parameters. For instance, Robust-PCA (Candès et al. (2011)) splits the data matrix into a low-rank representation matrix and a sparse error matrix and minimizes its cost function (which is some norm of the error matrix) regularized by the rank of representation matrix. Finding the appropriate degree of regularization is difficult. In factorization methods such as (Cabral et al. (2013)) the low-rank representation is obtained by learning a dictionary and coefficients for each data point, involving a hard to tune regularization parameter. In Sparse Subspace Clustering (SSC) method (Elhamifar and Vidal (2013)) a block-diagonal sparse matrix along with an error matrix are again combined (with the aid of a parameter) to quantify relations between data points in each cluster. To identify structures,

both the error and the L_1 norm of the sparse matrix are minimized, which involves tuning the regularization parameter. To improve the result, in Low-Rank-Representation (LRR) method (Liu et al. (2013)), nuclear norm of this sparse matrix was used as the regularization term, while tuning the regularization parameter remains a challenge. To ease the tuning process, an estimate of the regularization parameter is suggested by (Liu et al. (2016)), which includes the number of data points. Despite its high speed and accuracy for small datasets, its extension to problems involving large datasets is unclear.Recently methods such as LRSR (Wang et al. (2016)) and CLUSTEN (Kim et al. (2016)) have added more constraints to the regularization used in LRR, which makes tuning even harder. Global Dimension Minimization in (Poling and Lerman (2014)), used to estimate the fundamental matrix for the problem of two-view motion segmentation, takes a similar strategy. The method is relatively more accurate compared to LRR and SSC but remains computationally expensive.

Generally speaking, these methods require high computational power while the effect of regularization is controlled by parameters that are difficult to tune. These parameters often depend on noise scales, complexity of structures and ever-increasing number of structures and data points, which vary between data-sets and applications. Main advantage of our proposed method is that it requires minimum prior knowledge and avoids such parameters. The promising performance of the recent sampling-clustering approaches (Tennakoon et al. (2016) and Purkait et al. (2017)), provides a clear motivation for designing an effective sampling technique. This article is an extended version of the paper (Sadri et al. (2016)) and includes a novel element (i.e. structure size estimation) that significantly improves the performance of the method. Indeed, this is the most important difference between the proposed method and other papers that benefit from the optimization method of (Bab-Hadiashar and Hoseinnezhad (2008)). We have also included te results of several more experiments in this paper.

Tuning optimization parameters of these algorithms is essentially the same as sampling from the distribution of these parameters by an expert. The main intuition of our strategy is to sample from the distribution of putative model in parameter space similar to approach proposed by (Li et al. (2015)). This method uses a Mixture of Gaussians whose parameters are obtained through Expectation-Maximization steps. As the location of structures are unknown, using low number of Gaussians may lead to missing the structures and increasing the number of Gaussians is computationally expensive for EM. We propose to explore the parameter space using the distribution of hyperedges, form a hypergraph and perform segmentation using hypergraph clustering.

2.2. Guided sampling methods for robust model fitting

There are many sampling techniques developed for the purpose of data segmentation (regardless of the method of segmentation). Methods such as (Liu and Yan (2012); Jain and Govindu (2013); Chen and Lerman (2009b)) use uniform sampling from data. However it has been shown that uniform sampling generates mostly low quality hypotheses specially when size of samples is large, as the probability of choosing a pure sample decreases for larger sizes. One way to produce high quality samples is to use guided sampling methods such as Multi-GS (Tat-Jun et al. (2012)) and Swensden-Wang method (Swendsen and Wang (1987)). The former is a method designed for sampling minimal subsets. The latter was used by (Purkait et al. (2017)) which achieved higher speed compared to Multi-GS, partially due to use of large samples. In this method using an iteratively refined set of random clusters (based on work of (Pham et al. (2014)) and Swensden-Wang method), a moderate number of large pure samples were generated. However, the algorithm initially depends on spatial continuity and mistakes in early clusterings can lead to generation of impure samples later. Same problem can be seen in the sampling method given in (Ghoshdastidar and Dukkipati (2017)). In this paper, we propose a sampling method that requires no such prior knowledge.

In LBF method (Zhang et al. (2012)), it is proposed to guide samples by optimizing a cost function over parameter space. The cost function of their choice is the β-number of the residuals of a model. The suggested optimization method relates on the gradients of the cost function rather than second derivatives, which limits its speed. Moreover, the cost function derivatives are very high in areas close to structures, so prior knowledge such as spatial continuity is necessary to produce good initializations (Tran et al. (2014)).

Another guided sampling method called HMSS, using the Least k^{th} Order Statistics (LkOS) estimator was introduced in (Tennakoon et al. (2016)) facilitating a fit-and-remove segmentation strategy. In this method samples are randomly generated and guided to fit data properly and the segments are removed from the data to make recovery of other structures easier. The LkOS cost function is defined as:

$$C(\theta) = r^2_{[k],\theta} \tag{2}$$

where $r^2_{[k],\theta}$ is the k^{th} sorted squared residual with respect to model with parameters $\theta \in \mathbb{R}^{\Omega}$ and k refers to the structure size. This cost function has minima around the underlying structures in parameter space and is biased towards structures with low variance regardless of their size (Rousseeuw and Leroy (2005); (Chin et al. (2009a))). Performing local optimization on this function is challenging as it is highly non-linear due to the sorting step (it is infeasible to obtain derivatives of the target distribution with respect to parameters or state). (Bab-Hadiashar and Hoseinnezhad (2008)) presented Fast-LkOS (FLkOS) which uses approximate second derivatives (similar to Newton method) to find a local minimum which makes the method fairly fast. Unlike LBF optimization method, it does not require prior knowledge about purity of the initial sample. (Tennakoon et al. (2016)) shows that this method can easily be extended to use larger sample sizes and find more accurate estimates of the cost function minima. The method is different to that work in two different directions: 1- In HMSS method, samples are compared based on the cost of finding the best candidate and structures are removed sequentially. By choosing only one of many samples of a distribution, the method becomes vulnerable to over-fitting (to that specific sample). Then, by inaccurate removal of structures, other struc-

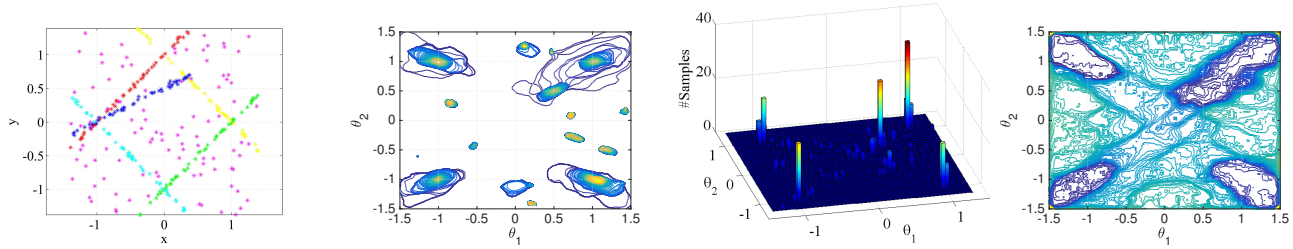

Fig. 1. (a) An example of a line fitting problem. Here segmentation is done using proposed method (b) The contours of proposed PDF $\tilde{P}_\Theta(\theta)$ (c) Unnormalized histogram of excessive number of samples produced by our MCMC method (d) The proposed cost function with infered k explained in section 3.4

tures will be affected and accuracy may be reduced. In our method, we generate samples according to the distribution of models and use Spectral Clustering which incorporates the information in all samples at once. 2- In HMSS, there is no clear way of estimating structure size k to improve optimization (because samples are discarded after selection of the candidate). In the proposed framework, the samples are used to sequentially improve their probability. Other methods such as TSSE (Wang and Suter (2004)) (which uses mean shift), MLESAC (Torr and Zisserman (2000)) (which uses EM) and IKOSE (Wang et al. (2012)) have also been proposed to guide samples, but these methods are highly dependent on the quality of initial samples and they are computationally expensive.

When a good sample is found and model parameters are derived, a robust scale estimator is often used to find a structure. Traditional robust scale estimators such as MED, MAD, KOSE, ALKS (Lee et al. (1998)) and MSSE (Bab-Hadiashar and Suter (1999)) provide the structure size as well as noise scale. The underlying assumption of these methods is that the model fits the structure with very high probability and the noise density is often assumed to be Gaussian. On one hand, the model may not fit the structure perfectly before finding a good sample and on the other hand, the success of optimization method relies on knowing the size of the structure. The major issue is that, if the sample is not ideal, robust scale estimators produce poor estimates of the structure size. Our proposed sampling framework produces a useful estimate of the structure size before optimization.

2.3. Robust model fitting by hypergraph clustering

A weighted hypergraph $\mathcal{H} = (\mathcal{V}, \mathcal{E})$ can be fully defined by a binary incident matrix H of size $|\mathcal{V}| \times |\mathcal{E}|$ and a diagonal weight matrix $W = \mathbf{diag}([w(e)]_{e=1\cdots|\mathcal{E}|})$. Here, the e-th element of the weight matrix, $w(e)$, represents the affinity between the subset of vertices, v_e, belonging to edge e and for all vertices $v \in v_e$ we have $H(v, e) = 1$. Given a ρ-tuple of data points in e, model parameters are obtained by fitting a model to the associated data x_e such that $\theta_e = \min \mathcal{F}(\theta, x_e)$. To calculate the residuals for all ρ data points, a distance measure is defined such that $r_{v,\theta_e} = \mathcal{R}(v, \theta_e)$ for all $v \in v_e$. The cost function $\mathcal{F}$ is defined as some norm over the fitting errors of data points x_e to model θ, i.e. $\mathcal{F}(\theta, x_e) = \|\mathcal{R}(x_e, \theta)\|_L$ and L is usually 2. The evaluation function $\mathcal{R}$ should be defined according to the type of the geometric model and the number of parameters. We continue without exact formulation of this function but the examples for Homography fitting, Fundamental matrix fitting and subspace fitting can be found in section 4. The weight of the hyperedge e can then be defined as:

$$w(e) = exp\left(-\frac{\sum_{v \in v_e} r^2_{v,\theta_e}}{2\sigma_e^2}\right) \tag{3}$$

where σ_e is a normalization constant. In this method, σ_e is calculated by applying a robust scale estimator (i.e. MSSE (Bab-Hadiashar and Suter (1999))) to all residuals. By generating a number of samples of the hyperedges, accurate estimates of H and W are obtained. Using matrix H the following two diagonal matrices can be constructed: $D_v = \mathbf{diag}([d(v_i)]_{i=1,\ldots,|\mathcal{V}|})$ and $D_e = \mathbf{diag}([|v_e|]_{e=1,\ldots,|\mathcal{E}|})$. Here $d(v_i) = \sum_{e \in \mathcal{E}} w_e \times H[v_i, e]$ is the sum of weights of all edges containing v_i. The matrices W, H, D_v and D_e are required to perform *NCut* on the hypergraph.

The original *NCut* was generalized to hypergraphs by (Zhou et al. (2006)). In hypergraph *NCut*, the vertices $\mathcal{V}$ are partitioned in two non overlapping segments $(S \cup S_c) = \mathcal{V}$ such that the following criterion is minimized:

$$NCut(S, S^c) = vol(S, S^c)\left(\frac{1}{vol(S)} + \frac{1}{vol(s^c)}\right) \tag{4}$$

where $vol(S, S^c) = \sum_{e \in c(S,S^c)} w(e) \times \frac{|e \cap S||e \cap S^c|}{|e|}$ with $c(S, S^c)$ being the hyper edges that need to be cut in order to partition S and S^c and $vol(S) = \sum_{e \in S} d(v)$. Both (Shi and Malik (2000)) and (Zhou et al. (2006)) solved a relaxed version of the above problem by reducing the hypergraph to a graph with the following adjacency matrix:

$$A = HWD_e^{-1}H^\top - D_v \tag{5}$$

To cut the hypergraph, the eigenvectors corresponding to the C largest eigenvalues of laplacian of A (defined as $\Delta_A = I - \frac{1}{2}D_v^{-1/2}AD_v^{-1/2}$) is found and data points are then partitioned using a clustering method such as K-Means.

2.4. Quality of the resulting cut

A high quality graph has an adjacency matrix (A) with a block-diagonal structure. Element a_{ij} of matrix A should have a high value if vertices v_i and v_j belong to same structure and low otherwise. The dominant eigenvectors of this matrix are those

that span most of the columns of A. This means that block-diagonal shape happens if chosen hypotheses are densely from underlying structures. If the density of low quality samples is high, elements of A that represent vertices of different structures can also have high values and the final cut can become inaccurate. Detailed explanation on this phenomenon is provided in (Agarwal et al. (2005); Ochs et al. (2014); Purkait et al. (2017)).

To provide an example, an instance of a simple line fitting problem is shown in figure 1(a). In this example, the aim is to segment $N = 500$, 2D data points into $C = 5$ clusters in presence of outliers. The contours of putative model distribution $\tilde{P}_\Theta(\theta)$ (based on exhaustive sampling) is shown in figure 1(b) while our approximation of this distribution is shown in figure 1(c). Comparison of these two figures shows that the sampling strategy has been successful in generating samples, densely from the high probability areas. The success of method depends on the success of the optimization step that finds local minima of the cost function in equation (2). The contours of this function for the line fitting example is given in figure 1(d).

2.5. Markov Chain Monte Carlo sampling from multi mode distributions

Markov Chain Monte Carlo (MCMC) is the most commonly used method in sampling from a distribution that can be evaluated only up to a proportional constant. Detailed description of this technique is outside the scope of this paper and interested readers are refereed to (Andrieu et al. (2003)). As the distribution of putative models in the prescribed model fitting problem may include multiple isolated modes, a mode jumping mechanism has to be included. Inspired by the mode jumping mechanism developed in (Tjelmeland and Hegstad (2001)) for continuous spaces, we designed a similar Markov chain for the discrete space. In this method, the Markov Chain is constructed by short jumps and long jumps as well as local optimization steps.

3. The proposed method

As discussed previously, the use of hypergraphs to solve a model fitting problem involves two challenges: 1) The structure of the graph is not known a priori; therefore we need to construct the full hypergraph, calculating which is very expensive; 2) The weight measure for robust model fitting is sensitive to accidental alignment of outliers.

Our aim is therefore to efficiently and effectively construct a sampled version of the hypergraph in order to solve the robust model fitting problem. The formulation of our edge sampling distribution and the method to effectively sample edges are outlined in sections 3.1 and 3.2. Afterwards we analyses the importance of k in section 3.3 and our proposed method to sample from its distribution in section 3.4. We will finish this section with providing the overall algorithm in section (3.5).

3.1. The sampling distribution

In robust model fitting, each hyperedge $e \in \mathcal{E}$ has a corresponding model instance with parameters $\theta \in \Theta$. Sampling edges from $P_\mathcal{E}(e)$ in equation (1) can be viewed as sampling θ from $P_\Theta(\theta \mid X)$. For any given set of input data X, we have:

$$P_\Theta(\theta \mid X) \propto P_\Theta(X \mid \theta) P_\Theta(\theta). \tag{6}$$

Given that all parameters θ are equally likely at the outset, the uninformative prior $P_\Theta(\theta)$ is assumed to be uniform. This is a significantly less restrictive assumption, compared to methods that impose a prior spatial continuity constraint, such as used in (Brox and Malik (2010)), (Zhang et al. (2012)) and (Purkait et al. (2017)). For the calculation of likelihood $P_\Theta(X \mid \theta)$, we use a slightly modified version of the robust k-th order cost function:

$$P_\Theta(X \mid \theta) = \frac{1}{Z} exp(\frac{-r_\theta^2[k]}{\sigma_\theta}) \tag{7}$$

where $r_\theta^2[j]$ is the j-th sorted square residual with respect to model θ and σ_θ is a normalization constant. The parameter k in this equation is the minimum number of points accepted to form a structure and is problem dependent. The value of this likelihood will be high only if the corresponding edge contains vertices that are all members of the same structure. As will be described in section 3.2, the proposed MCMC sampling strategy does not need any knowledge of the normalization factor Z.

3.2. The proposed Markov chain

A Markov chain consists of a transition kernel $Q_\Theta(\cdot|\theta)$ and an acceptance criterion, such as Metropolis-Hastings, which defines a probability of accepting a new sample as:

$$\alpha(\theta^*|\theta) = min\left\{1, \frac{P_\Theta(X \mid \theta^*)}{P_\Theta(X \mid \theta)} \frac{Q_\Theta(\theta|\theta^*)}{Q_\Theta(\theta^*|\theta)}\right\}. \tag{8}$$

As the distribution $P_\Theta(\theta \mid X)$ is multi-modal, we need to construct the transitions so that the Markov chain is not trapped in any local maximum. To achieve this, we compose our proposed Markov chain to include short jumps to exploit a single structure and random long jumps for exploring the space of Θ. This long jump is the key that the chain visits all structures and since their model is not a given priori it has to be random. It is noteworthy that we only use data supported states in the parameter space which means that random jump would be nothing but random sampling from data as is explained shortly.

Long jump construction as illustrated in figure 2 uses a simple 2D distribution (with two modes) as the target distribution $\tilde{P}_\Theta(.)$ and has two elements: The first is to move from $\theta \rightarrow \theta^*$ (the forward path) and to quantify $Q(\theta^*|\theta)$; The second is to quantify $Q(\theta|\theta^*)$, the probability of going from $\theta^* \rightarrow \theta$ (the reverse path). These two elements are used to find the acceptance probability for a new sample. This construction is explained in 5 following steps:

Step 1: A large random transition

The first step is to make a large random transition (φ) in parameter space. This hopefully moves the state away from the basin of attraction of the current mode, to a new state, denoted by $\theta_\varphi = T_0(\theta, \varphi)$. Constructing such a move requires two elements: 1) a method to generate φ; 2) a function that performs the translation. Examples of T_0 include $T_0(\theta, \varphi) = \theta + \varphi$ for

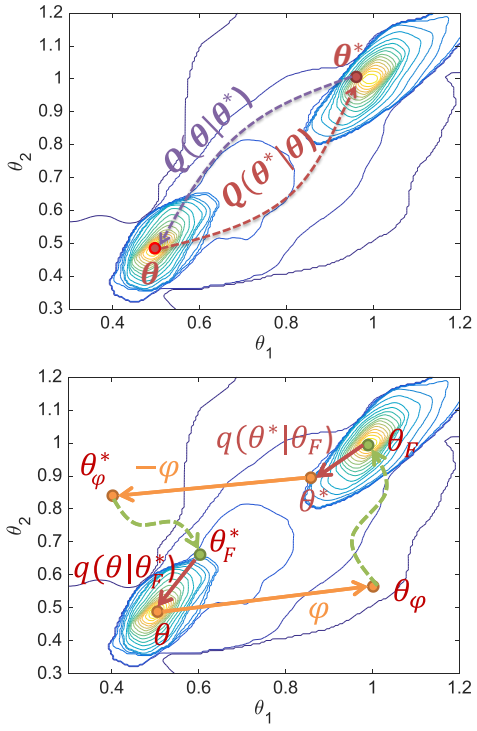

Fig. 2. A typical 2D bi-modal distribution, here the purpose is to make a jump between two modes, e.g. from θ to θ^*. Top: The overall transition which is modeled by $Q(\cdot|\cdot)$, Bottom: The proposed multi-step transition ending with a short transition according to the model $q(\cdot|\cdot)$.

translation by a vector φ or $T_0(\theta, \varphi) = \varphi\,\theta$ for rotation or scale by a properly contained matrix φ.

At the first look, the long jump may seem to be the outcome of selection of the next point in the parameter space according to some criterion. Indeed, the most widely used method of generating φ in MCMC methods is to assume a distribution with a very large variance around the current state (θ) and to use it to make the long jump. This would require some knowledge of the separation between modes (how far is far enough?), which may not be practical in data segmentation problems. In our algorithm, instead of commonly these used random jumps, we propose the selection of a new state based on an inliers or outliers dichotomy of the current hypothesis (state).

The state θ_φ is derived as follows: Sort the squared residuals $\{r^2_{j,\theta}\}_{j=1}^N$ and perform the MSSE to separate inliers from outliers. Randomly select a data point from the outliers and get the $\rho - 1$ points around it in sorted residual space, where ρ is the chosen sample size. Then a model can be fitted to these ρ points to obtain θ_φ. The rationale here is that, since there is an effective method to dichotomize inliers/outliers, a state obtained by sampling among outliers would be distant from the current mode. Selecting the ρ subset in the sorted residual space is beneficial because of the fact that data points from the same structures tend to cluster together in that space (Toldo and Fusiello (2008)). However, the algorithm does not require the ρ points to be purely from the one structure because the local optimization that follows will guide the sample to a local optimum.

Step 2: Local optimization

Next, a local optimization on $\tilde{P}_\Theta(\theta)$ is performed to transit to a state in a high probability area denoted by θ^*. This local optimization step improves the sample quality. From the Markov chain property perspective, it is a deterministic move that simply guides the sample to the closest local maximum of the distribution. However, the success of this step is sensitive to an accurate estimate of the structure size k before the optimization. Our heuristic method, which samples from the distribution of $P(k|\theta)$, is given in section 3.4.

Step 3: Short transition

A short jump to θ^* is performed using a local distribution $q(\cdot|\theta_F)$. Given that θ_F is the location of the optimum and $\Sigma(\theta_F)$ is the inverse of the Hessian of the $\tilde{P}_\Theta(\theta)$ at the optimum, i.e. $\Sigma(\theta_F) = [\nabla^2 \tilde{P}_\Theta(\theta)|_{\theta=\theta_F}]^{-1}$, one typical choice of the local distribution can be:

$$q(\theta^*|\theta_F) = \mathcal{N}_\Omega(\theta^*; \theta_F, \Sigma(\theta_F)) \tag{9}$$

which is the density of an Ω dimensional Gaussian random variable with mean θ_F and covariance $\Sigma(\theta_F)$.

Calculation of $q(\cdot|\cdot)$ using a normal distribution, as given in equation ((9)), is not practical in the data segmentation problems because of the difficulty of estimating the covariance matrix. Therefore, in our implementation, the jump from θ^F to θ^* is made by taking one more step in the optimization process. The probability value $q(\theta^*|\theta_F)$ (and $q(\theta|\theta^*_F)$ in the reverse path) is measured via a novel measurement based on the T-distance (Wang et al. (2015)). We define the probability of moving from one state to another as:

$$q(\theta^*|\theta_F) = \frac{1}{N}\langle \mathbf{w}_{\theta^*}, \mathbf{w}_{\theta_F} \rangle \tag{10}$$

$$q(\theta|\theta^*_F) = \frac{1}{N}\langle \mathbf{w}_{\theta}, \mathbf{w}_{\theta^*_F} \rangle \tag{11}$$

where $\langle\cdot,\cdot\rangle$ denotes the inner product and $\mathbf{w}_\theta$ is the vector of affinities for all data points to the model with parameters θ.

Step 4: The reverse path

In order to construct the reverse path we begin by moving from θ^* to $\theta^*_\varphi = T_1(\theta^*, \varphi)$. We rather choose T_1 based on T_0 such that $T_1(T_0(\theta, \varphi), \varphi) = \theta$. Examples include $T_1(\theta^*, \varphi) = \theta^* - \varphi$ for translation by vector φ or $T_1(\theta^*, \varphi) = \varphi^{-1}\,\theta^*$ for rotation or scale by inverse of matrix φ. The next step is to perform a local optimization to move to θ^*_F, in the hope that we get back close to the original mode. The last step is to calculate $q(\theta|\theta^*_F)$ to find the probability of jumping from θ^*_F to θ.

Step 5: Calculation of $Q(\cdot|\cdot)$

Assuming the independence of the random long transition from the short jump, $Q(\theta|\theta^*) = q(\theta|\theta^*_F)\,\Phi(\theta^*_\varphi|\theta^*)$ and $Q_\Theta(\theta^*|\theta) = q(\theta^*|\theta_F)\,\Phi(\theta_\varphi|\theta)$ where $\Phi(\cdot|\cdot)$ is the density function of the state after a random long transition from another state. The focus of this work is mainly on applications where the target structures are of relatively small sizes compared to the overall data population (i.e. the number of outliers and pseudo-outliers are far larger than inliers for any structure). With this approximation, the possible domain of the parameter values after a long transition asymptotically covers the whole parameter space, as the number of data points approaches infinity. Considering that the p-tuples selected from outliers (to select the next state for a long jump) are equally probable, we deduce that the parameter

value after a long transition is almost uniformly distributed in the parameter space, and so is its conditional density. Hence, for the two long transitions involved in our method we have $\Phi(\theta^*_\varphi|\theta^*) = \Phi(\theta_\varphi|\theta) = \text{cte.}$, and we end up with $\frac{Q_\Theta(\theta|\theta^*)}{Q_\Theta(\theta^*|\theta)} = \frac{q(\theta|\theta^*_F)}{q(\theta^*|\theta_F)}$. If the reverse path moves the state to a mode other than the original, $q(\theta|\theta^*_F)$ will be small and so will the probability of acceptance $\alpha(\theta^*|\theta)$ be.

The short jumps are performed simply by selecting a random sample h_φ (modeled by θ_φ) among the inliers of θ.

3.3. Importance of k:

An important parameter in the cost function (2) is the minimum structure size k. We have observed that k has a significant effect on the shape of the cost function in equation (2). When the value of k is increased, the peaks corresponding to smaller structures (e.g. the accidental alignment of outliers or the detection of multiple structures as a result of over-segmentation) will disappear from the sampling distribution and the cost function would become smoother. This means that the ideal value of k changes for different structures even within the same dataset. This directly affects the optimization step, as the size of the basin of attraction also depends on the value of k.

For example, in the earlier line fitting problem, a set of models are generated uniformly, with intercepts and slopes from the range $[-1.5,\ \ 1.5]$ with resolution 0.01. The optimization is then performed for each model to reach the optimum state. If the Euclidean distance of the optimum state from any of the structures is less than 0.1, it is assumed to have come from the basin of attraction of that structure. The color-coded basins of attractions, for $k = 12$, are shown in Figure 3(a) and for $k = 35$ in Figure 3(b) (structure sizes are 50). Figure 3 shows that the basins of attractions become larger as k increases, which leaves us with fewer local optima. In section 4, we show by simulation that choosing a small k results in a high error rate for segmentation. To reduce the segmentation error, we devise a method in section 3.4 to infer the ideal value of k before the optimization step.

3.4. Calculation of k:

We propose a novel method for using the previously sampled hyperedges to calculate the probability that the closest k data points to the current model (θ_φ) are from the same structure. In other words, the aim is to estimate the hyperedge weight containing the closest k data points to the model, using weights of previously sampled hyperedges. Since hyperedge samples are theoretically independent (except for the identical structures), the weights of the hyperedges corresponding to the closest k data points are proportional to discrete distribution $P_K(k|\theta)$. Assuming that we have an estimate of the model parameters before the optimization step, we state that k is also a parameter of the model and we have $P_{\Theta,K}(k,\theta) = P_K(k|\theta)P_\Theta(\theta)$.

One way to use the hypergraph information as a priori is to use iterative clustering such as that used in (Purkait et al. (2017)). We however infer the value of k from the hypergraph information using the definition of the *NCut* criterion in equation (4). To estimate the values of $P_K(k|\theta)$ for all k, the value of the *NCut* criterion for separating the closest k data points to the model is found, and we choose the likelihood:

$$P_K(k|\theta) \propto NCut(\{v_{I(1)}, \cdots, v_{I(k)}\}, \{v_{I(k+1)}, \cdots, v_{I(N)}\}) \qquad (12)$$

where I is the indices of sorted residuals in an ascending order with respect to the current model θ_φ.

The first local minimum of this function gives the closest k data points that can be segmented out with the lowest cost, denoted by $\hat{k}_\varphi$. The $P_K(k|\theta_\phi)$ for k larger than this $\hat{k}_\varphi$ is set to zero. This procedure is illustrated better by algorithm 1.

Algorithm 1 Proposed likelihood $P_K(k|\theta)$ up to a scale

Require: Hyperedges weights vector $\mathcal{W} \in \mathbb{R}^{NM}$, the incidence matrix $H \in \mathbb{R}^{N\times NM}$)
Ensure: The likelihood $P_K(k|\theta)$ up to a scale
for $k = k_m, \cdots, N$ **do**
  calculate $P_K(k|\theta)$ up to a scale by Eq. (12)
  if $k > k_m$ **and** $P_K(k|\theta) < P_K(k-1|\theta)$ **then**
    break
  end if
end for

3.5. Overall Sampling Algorithm

Algorithm 2 Proposed sampling method

Require: Data points X, number of samples M, a random state θ
Ensure: Hyperedges weights vector $\mathcal{W} \in \mathbb{R}^{NM}$, the incidence matrix $H \in \mathbb{R}^{N\times NM}$)
$A = \mathbf{0} \in \mathbb{R}^{N\times N}$
while $M > 0$ **do**
  Evaluate θ to find λ (probability of long jump)
  if $u < \lambda$ where $u \sim \mathcal{U}(0,1)$ **then**
    sample e_φ (a ρ-tuple from outliers)
    Find $\theta_\varphi = \min \mathcal{F}(\theta, e_\varphi)$
    Sample from $P(k|\theta_\phi)$ using algorithm 1
    Perform FLKOS to find e_F and $\theta_F = \min \mathcal{F}(\theta, e_F)$ s
    One more FLKOS step to find e^* and $\theta^* = \min \mathcal{F}(\theta, e^*)$
    Calculate $Q(\theta^*|\theta)$ and $Q(\theta|\theta^*)$
    Calculate $P_\Theta(X|\theta^*)$ and $P_\Theta(X|\theta)$ using Eq. (7)
    Calculate Metropolis-Hastings $\alpha = \alpha(\theta^*|\theta)$
  else
    sample e^* (a ρ-tuple from inliers)
    Find $\theta^* = \min \mathcal{F}(\theta, e^*)$
    Calculate $P_\Theta(X|\theta^*)$ and $P_\Theta(X|\theta)$ using Eq. (7)
    Calculate Metropolis $\alpha = \alpha(\theta^*|\theta)$
  end if
  if $\alpha > v$ (where $v \sim \mathcal{U}(0,1)$) **then**
    $\theta \leftarrow \theta^*$, and $M \leftarrow M - 1$
    Store $[w_{j,\theta^*}]_{j=1,\cdots,N}$ in $\mathcal{W}$
    $H_{v_i, M+\tilde{M}} \leftarrow 1$ for $v_i \in v_{e^*}$ and $\tilde{M} = 1, \cdots, N$
  end if
end while

The MCMC chain is initialized at a random state. A mixture of long and short jumps are performed in order to generate samples from the target distribution $\tilde{P}_\Theta(\cdot)$, as described in Algorithm 2. The probability of performing a long jump λ is set to $\lambda = \frac{1}{N}\sum_{j=1}^{N} w_{j,\theta}$, which means that if the number of inliers is

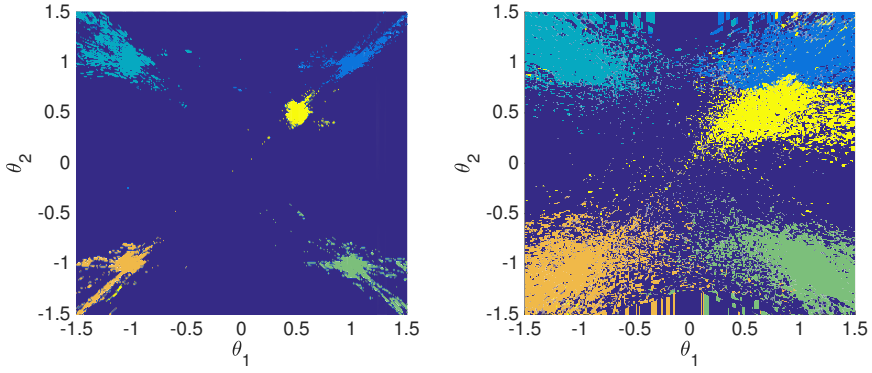

Fig. 3. (a) Basin of attraction of each structure for the optimization step for the line fitting problem (where all structures sizes are 50**) with** $k = k_m = 12$**. (b) Same for** $k = 35$**.**

small, it would rather exploit the local vicinity by making short jumps. Otherwise it is preferred to explore the data by a long jump. M samples and their affinity vectors are generated using this algorithm. After the generation of each sample, the incidence matrix H and the weights vector W are updated. If $\mathcal{H}$ is not updated, then k remain k_m (similar to our previous work in (Sadri et al. (2016))). When M samples are provided, the hypergraph is used to segment the data using *NCut*. In this algorithm the $\mathcal{U}(0, 1)$ denotes uniform distribution from 0 to 1.

4. Experimental Results

In this section we discuss our choices of parameters for the different methods outlined above. We have evaluated our method for a few well-known datasets. One is the Adelaide-RMF dataset introduced in Wong et al. (2011) which has two parts. One part is for two-view motion segmentation by fundamental matrices for motions of different objects and the other is for planar homography estimation. The second dataset is the 155-Hopkins dataset introduced in Tron and Vidal (2007) for multiple-view motion segmentation using subspace analysis.

We have evaluated our method called MCNC (Monte Carlo & *NCut*) 100 times for each problem in these datasets. The threshold parameter of the MSSE robust estimator is set to 2.35 in all problems proposed by (Wang et al. (2012)). The parameter k_m should be high enough to avoid generating a large finite sample bias for the MSSE, which is set to 12, proposed by (Hoseinnezhad et al. (2006)). The parameter ρ, the size of a sample, is set to $\rho_m + 4$ as proposed by (Tennakoon et al. (2016)), where ρ_m is the number of parameters of the model. The total number of samples is intended to be $M = 100\ C\ \log(N)$, where C is number of clusters. The results are given for two cases: once when k is not updated and is set to k_m (called MCNC(k_m)) and the other when inference is used (called MCNC). It is observed that the results are generally better in the latter case. It is noteworthy that our method does not explicitly include any other sensitive parameter that requires tuning.

4.1. Results on Adelaide-RMF dataset

We have tested how our method performs on the Adelaide-RMF dataset (Wong et al. (2011)) which includes pairs of matched feature-points between two views together with the ground truth labels. The task is to segment the pairs, belonging to each structure with different motions. For each model fitting, the minimization step, $\theta_e = \min \mathcal{F}(\theta, e)$, will fit a fundamental matrix F_{θ_e} or a Homography matrix H_{θ_e} to subset v_e of pair-data points $X_e = [X_{1,e}, X_{2,e}]$. Fundamental matrix fitting is done by solving $X_{1,e}^\top F_{\theta_e} X_{2,e} = 0$, where $F_{\theta_e} \in \mathbb{R}^{3\times3}$, $X_{1,e} = (x_1, y_1, 1)^\top$ and $X_{2,e} = (x_2, y_2, 1)^\top$ are the coordinates of the sampled subset in each view (Torr and Murray (1997)). Homography estimation is done via solving $X_{1,e}^\top H_{\theta_e} = X_{2,e}$. The Evaluation function $\mathcal{R}(\theta_e)$ uses Sampson distance (Hartley and Zisserman (2003)) to find residuals of each pair to the model.

Following (Wang et al. (2018)), we have evaluated our method for all sequences in the Adelaide-RMF dataset for two-view motion segmentation as shown in Table 1. We have also examined the performance of our method for homography based segmentation on the second part of the Adelaide-RMF dataset, with the results shown in Table 2.

In these two tables (Table 1 and Table 2), M1 represents the method called KF (Chin et al. (2009b)) which is a data clustering method based on random sampling and outlier removal (which occasionally removes some inliers), M2 represents the method HMSS (Tennakoon et al. (2016)) which is a sequential fit-and-remove method that uses random sampling to initialize the optimization method used in our work, M3 represents the method called Multi-GS (Tat-Jun et al. (2012)) which is a hypergraph clustering method similar to our method, with the only difference being in the sampling part which is minimal subset sampling (the number of samples for Multi-GS is set to same as ours). In these two tables, the results of running methods M1, M2 and M3 on our computer system (with a Core-i7-2.2GHz-4970k Mobile CPU and 16GB Memory, running MS-Windows 10) are shown. In order to see the effect of sampling method, we have incorporated their sampling strategy and used our clustering method to find final labels, which occasionally resulted in different accuracies compared to the ones presented in their papers. Also we have used similar parameters wherever possible. To compare our method with the recent state-of-the-art, we also included the results for MSHF (Wang et al. (2018)), which is M5 in tables. We borrowed the results reported therein, and acknowledged the fact that the computation time would not be

Table 1. Results for two-view motion segmentation on Adelaide-RMF dataset

Sequence	(#)		M1	M2	M3	M4	MCNC (k_m)	MCNC
Biscuit	1	Std.	0.21	5.68	23.06	0.55	5.2	5.47
		Avg.	0.48	14.71	25.84	1.3	12.8	13.91
		Time	7.86	0.50	79.34	5.27	1.86	1.68
Book	1	Std.	0.38	5.66	18.58	0.42	7.27	1.55
		Avg.	4.22	19.78	24.54	0.64	11.66	21.02
		Time	6.11	0.24	24.97	4.81	3.06	2.47
Cube	1	Std.	0.17	5.56	22.34	0.66	0.49	2.63
		Avg.	8.11	11.60	23.07	2.08	1.32	3.07
		Time	6.98	0.34	79.48	5.11	1.34	1.44
Game	1	Std.	0.1	4.05	32.65	0.74	11.27	1.55
		Avg.	30.45	11.37	38.15	2.44	5.11	2.58
		Time	5.95	0.24	42.7	4.95	0.89	0.91
Cubechips	2	Std.	0.2	6.06	0.85	0.98	17.63	7.76
		Avg.	1.41	14.60	5.63	3.55	19.07	4.31
		Time	12.29	0.76	64.87	5.18	2.56	2.57
Cubetoy	2	Std.	0.18	6.13	0.8	0.79	2.54	0.51
		Avg.	1.71	18.18	5.62	2.16	1.97	2.3
		Time	12.66	0.86	51.65	4.89	2.32	2.44
Breadcube	2	Std.	0.21	4.16	1.32	0.78	19	4.19
		Avg.	3.88	25.74	4.96	2.31	10.58	0.37
		Time	12.05	1.14	46.17	4.82	2.33	2.52
Gamebiscuit	2	Std.	0.39	1.58	1.85	0.74	1.39	3.6
		Avg.	1.72	20.57	7.32	1.95	4.85	7.33
		Time	12.98	0.68	91.49	5.81	13.37	7.82
Breadtoy	2	Std.	0.15	6.12	1.5	7.76	17.36	7.88
		Avg.	3.73	17.27	7.33	4.86	10.73	3.33
		Time	13.26	1.40	68.62	5.87	6.8	3.96
Breadtoycar	3	Std.	0.19	5.71	1.43	1.96	3.24	1.56
		Avg.	9.58	15.56	4.42	5.42	4.61	3.95
		Time	17.14	0.82	24.15	5.06	2.09	5.75
Biscuitbook	2	Std.	0.22	5.74	1.16	1.82	13.69	4.95
		Avg.	3.21	17.90	2.55	2.4	16.11	4.71
		Time	14.75	0.78	129.47	6.57	4.94	4.55
Biscuitbookbox	3	Std.	0.26	3.49	1.6	0.9	9.95	1.91
		Avg.	4.65	24.82	1.93	1.54	23.24	3.13
		Time	19.78	0.92	53.44	5.44	12	5.95
Breadcubechips	3	Std.	0.19	4.07	7.03	1.75	8.51	0.92
		Avg.	7.07	19.48	1.06	1.74	19.2	5.86
		Time	17.87	1.14	57.11	4.35	3.41	4.85
Cubebreadtoychips	4	Std.	0.11	5.27	7.72	6.62	7.81	2.95
		Avg.	11.36	22.87	3.11	4.25	22.16	7.19
		Time	27.6	2.62	91.05	4.7	5.96	4.13
Breadcartoychips	4	Std.	4.65	5.41	9.45	6.14	5.41	3.83
		Avg.	16.69	18.47	16.96	25.06	11.98	5.13
		Time	24.31	1.40	40.76	4.02	4.52	5.93
Carchipscube	3	Std.	0.1	3.32	0.44	7.56	5.97	2.61
		Avg.	10.91	19.76	17.51	25.51	30.42	4.72
		Time	17.07	0.80	18.52	3.81	4.46	5.56
Toycubecar	3	Std.	0.41	4.03	1.2	6.57	6.61	4.14
		Avg.	17.98	19.37	16.2	14	24.77	13.1
		Time	16.68	0.74	25.35	4.73	3.91	9.26
Boardgame	3	Std.	0.27	4.87	1.91	7.77	8.77	5.11
		Avg.	10.16	19.21	28.6	21.57	21.54	10.1
		Time	19.83	1.14	58.07	4.63	14.51	4.99
Dinobooks	3	Std.	4.24	4.89	1.85	2.08	10.11	10.1
		Avg.	14.71	20.40	19.52	18.05	26.43	15.9
		Time	20.43	1.92	118.96	4.89	13.78	5.25
Total Average		Std	0.64	4.83	7.19	2.97	8.53	4.22
		Avg	8.52	18.51	13.38	7.41	14.66	7.01
		Time	15.03	0.97	61.37	4.99	3.8	4.67

Table 2. Results for Homography based segmentation on Adelaide-RMF dataset

Sequence	(#)		M1	M2	M3	M4	MCNC (k_m)	MCNC
Bonython	1	Std.	2.5	1.08	16.54	0.01	0.76	1.85
		Avg.	29.04	3.70	28.28	0	0.16	0.32
		Time	1.19	0.28	11.65	0.96	0.21	0.22
Physics	1	Std.	1.33	9.28	14.87	0.01	11.81	10.54
		Avg.	13.21	19.96	39.43	0	5.57	4.97
		Time	0.96	0.20	12.68	1.75	0.16	0.16
Unionhouse	1	Std.	0.64	0.43	26.66	0.01	18.55	18.41
		Avg.	17.02	1.78	24.81	0.3	12.08	11.15
		Time	1.46	0.32	38.05	1.02	0.26	0.26
Elderhalla	1	Std.	0.33	6.53	0.45	0.15	0.5	2.73
		Avg.	6.78	7.46	1.17	0.93	1.02	1.43
		Time	2.28	0.44	15.28	2.16	0.31	0.3
Elderhallb	2	Std.	0.83	2.02	0.58	1.1	7.62	7.82
		Avg.	10.39	6.06	12.63	2.94	14.72	13.94
		Time	3.36	0.68	30.47	2.18	0.59	0.58
Hartley	2	Std.	0.22	5.70	0.32	0.31	0.84	0.82
		Avg.	6.09	5.02	2.5	1.9	1.96	1.9
		Time	2.61	0.52	62.16	1.63	0.43	0.44
Library	2	Std.	0.66	2.70	3.38	0.82	1.84	1.61
		Avg.	20	4.50	4.65	2.37	2.46	2.16
		Time	2.21	0.52	16.04	1.8	0.36	0.37
Sene	2	Std.	0.28	8.80	0.37	0.13	0.22	0.25
		Avg.	15	5.86	0.44	0.24	0.66	0.67
		Time	2.43	0.48	22.78	1.8	0.57	0.57
Nese	2	Std.	22.55	10.47	0.51	0.27	0.17	0.11
		Avg.	15.94	6.96	1.88	0.2	0.84	0.82
		Time	2.31	0.48	24.15	2.32	0.59	0.6
Ladysymon	3	Std.	10.14	9.34	2.58	0.86	1.07	1.48
		Avg.	22.78	10.51	5.06	2.62	7.67	7.87
		Time	2.3	0.44	20.86	2.39	0.85	0.84
Oldclassicswing	2	Std.	0.75	0.43	0.25	0.33	0.86	0.77
		Avg.	11.08	2.92	1.27	1.08	2.16	2.2
		Time	2.89	0.56	74.89	1.81	0.84	0.86
Neem	3	Std.	1.17	7.81	4.96	0.48	0.57	0.7
		Avg.	7.05	10.32	3.82	1.78	6.46	6.34
		Time	3.33	0.72	21.4	2.13	0.56	0.57
Johnsona	3	Std.	0.38	7.18	4.73	1.43	7.01	7.03
		Avg.	35.92	12.62	4.03	3.02	9.16	9.28
		Time	5.43	0.88	57.11	2.24	1.13	1.15
Johnsonb	4	Std.	0.33	5.34	10.51	4.96	4.77	5.18
		Avg.	64.64	18.06	18.39	16.61	20.69	21.79
		Time	13.42	1.72	261.62	5	4.83	4.75
Napiera	4	Std.	0.47	5.33	4.54	3.26	9.16	7.86
		Avg.	31.13	13.18	23.37	27.78	12.19	10.33
		Time	2.53	0.72	29.88	2.18	0.38	0.38
Napierb	3	Std.	0.82	5.50	5.14	4.12	1.09	0.86
		Avg.	30.69	11.83	19.92	13.12	8.38	8.27
		Time	3.47	0.76	21.93	1.87	0.97	0.97
Barrsmith	3	Std.	0	5.99	6.65	9.5	8.61	6.21
		Avg.	11.62	5.99	29.33	24.48	4.17	3.15
		Time	2.32	0.92	18.91	1.42	0.32	0.32
Unihouse	3	Std.	0.07	6.00	4.98	0.38	3.09	2.93
		Avg.	46.93	10.47	14.04	9.29	9.29	9.34
		Time	50.83	3.04	2908.61	10.5	8.25	8.22
Bonhall	3	Std.	1.59	11.14	0.13	8.69	4.76	4.16
		Avg.	64.04	18.05	29.06	31.65	24.01	23.23
		Time	20.65	4.68	835.38	7.87	3.59	3.57
Total Average		Std.	2.3716	5.85	13.89	7.38	5.38	3.9
		Avg.	24.177	9.22	12.32	10.3	8.56	7.3
		Time	6.6317	0.96	12.63	2.37	1.4	1.5

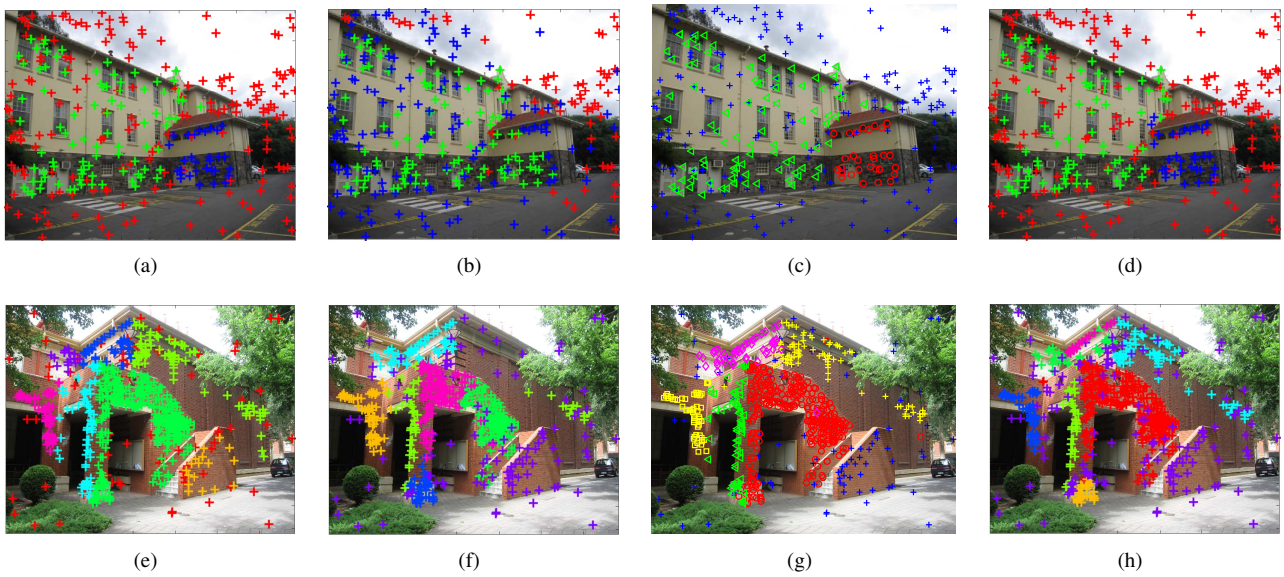

Fig. 4. Examples of Homography based segmentation, first row is for scene "Hartley" and second row is for "johnsonb". (a) groud truth (b) Multi-GS (c) MSHF (d) MCNC

the same for both computer systems. However, since in their experiments, they used a 2.4Ghz-core-i7-3960 CPU which has similar or better performance compared to our system, the time comparison is still reasonable. Examples of running MSHF, Multi-GS and our method on two scenes from each dataset are shown in figures 4 and 5.

From the results in Table 1, we observe that in total, for fundamental matrix estimation, our method returns a lower mean error than all the other methods, except for Multi-GS which is well-known to be a computationally expensive solution. In addition, the total average run times reported in Table 1 demonstrate that our method is faster than the others. The results reported in Table 2 demonstrate that in total, for homography-based segmentation, our method is both more accurate (with smaller mean estimation error) and faster (with smaller mean computation time). In particular, with the Multi-GS method, we observe that although it can recover fundamental matrices more accurately than our method (at the expense of heavy computations), it struggles with detection of planar homographies.

The main reason why our method generally outperforms KF (Chin et al. (2009b)) and HMSS is that these methods perform random sampling at their cores (blindly or guided), but ours samples from the distribution of hyperedges. In MSHF (Wang et al. (2018)), the vertices in hypergraph are hypotheses and the hyperedges are data points. Using random minimal samples (different from ours), a set of models is produced and a weight is given to each vertex (similar to our likelihood but with a different kernel). Then, an information theocratic approach is suggested to reduce the hypergraph to keep significant samples (where we use probabilities on Markov chain to decide about importance of samples). In our method, we explicitly treat vertices as data points and hyperedges as models and sample from distribution of hyperedges by their probability. However, as we use an acceptance criterion we are making much more effort to improve our samples sequentially (which seems to be more successful than the information theocratic approach). Also our design of Markov chain with optimization steps, assists our samples to represent structures more accurately. The long jumps ensure exploration of the space and short jumps tend to exploit each structure efficiently. As the tabulated results indicate, performance of our proposed method surpasses theirs in terms of average accuracy and speed, while standard deviations of errors of their method are generally lower, which suggests more stability.

4.2. Rigid body motion segmentation by subspace fitting

4.2.1. Real data

The task in this set of 155 video sequences is motion segmentation based on multiple-view trajectories tracked on each object in each video sequence. In this dataset, there are either two or three objects with different motions; this information is used as prior knowledge in the *NCut* implementation. If the affine camera projections of N points to the image plane within F frames are available, the trajectories belonging to a rigid object span a subspace of $rank \leq 4$ (Vidal (2011)). Therefore, subspace clustering can be used to solve the problem. The data will have the form $X = \left[x_{f\alpha}\right]_{\alpha=1,\ldots,N}^{f=1,\ldots,F}, x \in \mathbb{R}^2$. The model fitting step, $\theta_e = \min \mathcal{F}(\theta, e)$, performs two steps on the sampled subset in e: first, to remove its translation $t_e = \sum_{j=1}^{|e|} x_j$ ($x_j \in \mathbb{R}^{2F}$), which means $\tilde{X}_e = X_e - \mathbf{t}_e$ ($X_e = [x_j]_{j=1,\ldots,|e|}$ and $\mathbf{t}_e = [t_e\ t_e\ \ldots\ t_e] \in \mathbb{R}^{2F\times|e|}$), and second to take the first r eigenvectors of $\tilde{X}_e$, $U_{(e,r)} \in \mathbb{R}^{2F\times r}$ where r is chosen based on the eigenvalues of $\tilde{X}_e$ (Note that the sampled subset may live on a lower dimensional subspace than 4 dimensions (Vidal (2011))). It should be noted that proper model selection can be very effective in recovery of model instances. Even though in our current setting, we use the simple method explained above, effective methods such as Transfer Cost based model selection (Frank et al. (2011)) can be very useful and would help with estimat-

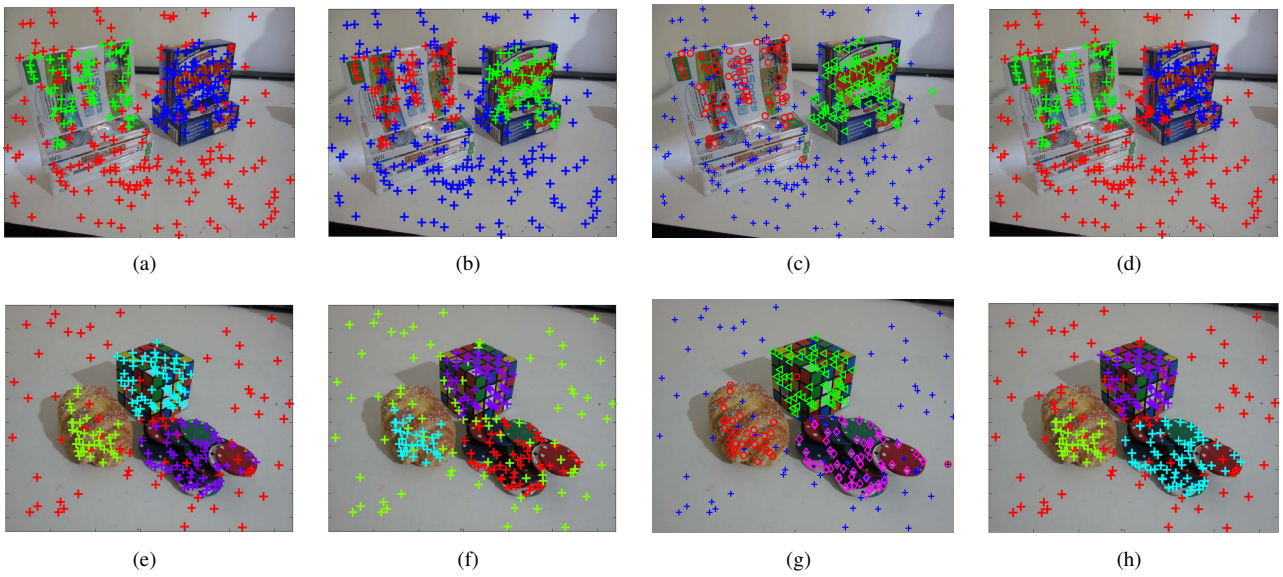

Fig. 5. Examples of two-view motion segmentation, first row is for scene "gamebiscuit" and second row is for "breadcubechips". (a) groud truth (b) Multi-GS (c) MSHF (d) MCNC

ing both r and C. In this work however, we chose the simplest method to achieve higher speeds.

The evaluation function $\mathcal{R}(\theta_e)$ also takes two steps: it first removes sample's translation from all points $\tilde{X} = X - \hat{\mathbf{t}}_e$ where $\hat{\mathbf{t}}_e \in \mathbb{R}^{2F\times N}$, and then uses the subspace projection error $\mathbf{r}_{\theta_e} = \left\|\tilde{X} - U_{(e,r)}U_{(e,r)}^T\tilde{X}\right\|_2$ to find the fitting residuals of all data points. As can be seen from the results in Table 3, our method runs at relatively high speed and outperforms selected methods in terms of average and median error accuracy.

In Table 3 The selected methods include Spectral Curvature Clustering (SCC (Chen and Lerman (2009b)) and HOSC (Purkait et al. (2017)) which use iterative clustering to guide samples. The results for HOSC is obtained by testing the method with parameters given in the paper, and the results for SCC is borrowed from (Chen and Lerman (2009a)). The table also includes results for the method called QP-MF (Yu et al. (2011))) (The results are borrowed from the paper), since it provides an accurate and stable optimization for the task of energy minimization. Other methods included here are (HMSS (Tennakoon et al. (2016)) (The results are borrowed from the paper) and Multi-GS which we tested using the clustering method we used for evaluating our own sampling technique without changing parameters such as number of samples.

The reason that the accuracy of our method surpasses that of SCC and HOSC can be due to the fact that iterative clustering can easily fail if the clustering is inaccurate in early stages. The speed of our method is less than some methods because our method rejects some samples during the MCMC process. The performance of our method however, succeeds that of the energy minimization based method *QP-MF* mainly because it uses random sampling.

4.2.2. Synthetic data

In order to show the importance of using proper k, we devised a synthetic scene with a set of feature points on three boxes (see figure 6). These are similar to the objects used in the Hopkins dataset. The boxes, which have 50 data points on each face, rotate and translate in 3D space for 25 frames. The Locations of feature points are also perturbed by small, normally distributed random values. The final adjacency matrix A in equation (5) is given for three values of k in this figure. If the value of k is small (e.g. $k = 15$), each face will have a block in the matrix and the over-segmentation becomes inevitable. This is demonstrated in figure 6(b). Increasing the value of k improves the chance of generating a sample that includes data points from multiple faces. While each face has 50 data points, the result for $k = 65$ is shown in figure 6(c). On the other hand, when the value of k is accurately inferred, the adjacency matrix finds the desired three blocks as is shown in Figure 6(d). The eigenvalues of the Laplacian of these matrices for different values of k are also shown in figure 6(e). It shows that, with small value of k, the largest 9 eigenvalues are close to 1 where 6 of these eigenvalues are supposed to be significantly smaller that the largest three. The significance of finding the right value of k can be seen in the difference between the three largest and the six smaller eigenvalues. This difference, relied upon for successful segmentation, is significantly increased using the proposed method of inferring k.

4.3. Motion segmentation using dense trajectories

The Berkeley motion segmentation dataset includes video sequences provided for the purpose of motion segmentation. The task is very similar to the previous one, except that in this dataset, the trajectories are not hand-trimmed. The data includes structures with very small or large sizes with strongly varying noise scales along with outliers. The data may be incomplete, which means that each trajectory may be available for only a few frames in the sequence. We use the same data, provided by (Ochs et al. (2014)), and the same clustering method. However, we compare the accuracy and speed of our method

Table 3. Comparative performance in terms of accuracy and speed using the Hopkings 155 dataset.

	Multi-GS	QP-MF	HMSS	SSC	SCC	HOSC	MCNC(k_m)	MCNC
2 Motions								
Mean	1.93	4.16	3.98	2.23	1.40	5.28	6.9	**1.36**
Median	0.78	0.46	0.00	0.00	0.10	0.02	1.20	**0.00**
Time	5.6	-	-	0.65	0.66	1.27	0.75	0.60
3 Motions								
Mean	6.11	7.25	11.06	5.77	5.90	7.38	8.9	**5.71**
Median	2.63	4.9	1.20	0.95	1.99	1.53	3.2	**0.18**
Time	6.1	-	-	1.47	1.29	2.00	2.00	1.94

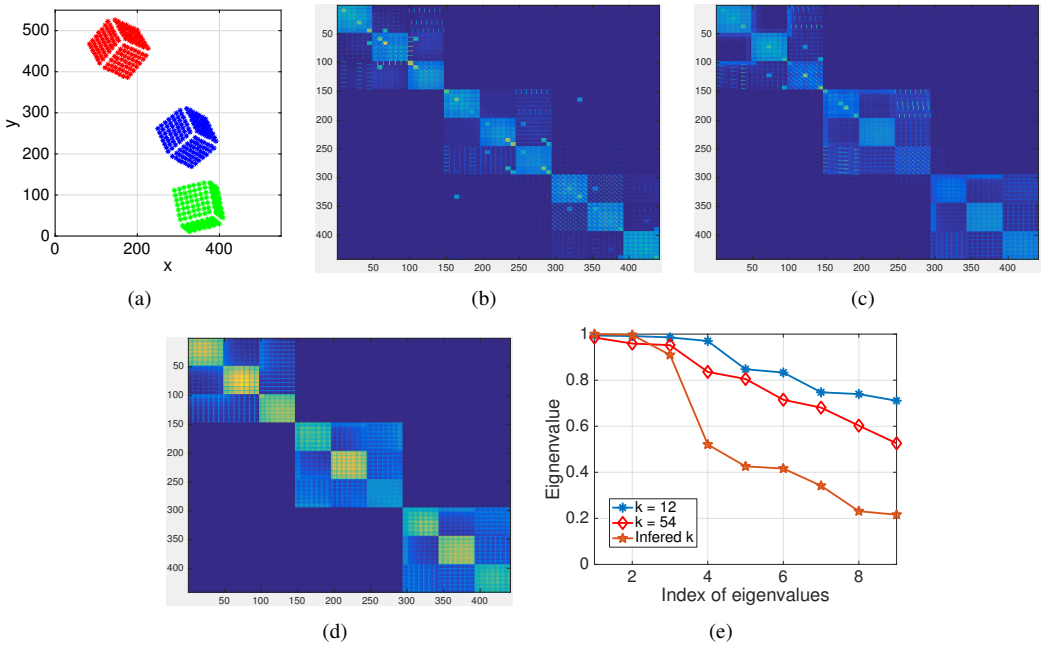

Fig. 6. (a) An example of 3D motion segmentation problem with synthetically generated feature points on three boxes; (b) The adjacency matrix when $k = 12$; (c) When $k = 65$; (d) When k is inferred through iterations; (e) The largest 9 eigenvalues of the normalized adjacency matrices for comparison

as a result of the new sampling technique. The accuracy of segmentation is calculated using manually created and labeled masks for different objects in each video. The reader is referred to the original article in (Ochs et al. (2014)) for more details. We have tested our method on all frames of the 26 video sequences provided in the dataset and the results are given in Table 4. In this table, We have included the result from the method called BM (Brox and Malik (2010)) which uses spatial continuity to find pair-wise distances between data points and form a graph. The other method included here is called OB (Ochs et al. (2014))) which forms the complete 3-uniform hypergraph. Even though this method is very accurate, it is extremely expensive in terms of computation (see the substantially large computation time listed in Table 4). Also we have mentioned the results for HOSC method Purkait et al. (2017)). As can be seen, our MCNC method is comparable in terms of speed and accuracy to the state of the art.

5. Conclusion

This paper presents a sampling framework for choosing appropriate hyperedges of a hypergraph that will enable effective data segmentation using the *NCut* method. We argued that sampling from the distribution of hyperedges is similar to sampling from the distribution of putative models in parameter space. An elaborate Markov Chain was designed to effectively sample from this distribution. The chain includes greedy long jumps for exploration and random short jumps for exploitation of structures. The design of long jumps exploits the accumulated knowledge of the (sensitive) structure size parameter and uses that to produce informative samples. Compared to other methods, our method performs reliably and efficiently in solving data segmentation problems using geometric constraints. The improved performance of our method is mainly due to the better quality of samples generated through the proposed sequence of transitions in the parameter space. An extension of this work is to investigate how various segmentation algorithms can benefit from the high-quality samples generated by this method. In particular, in conjunction with energy minimization methods such as (Delong et al. (2012)), if method's parameters are properly tuned, we can achieve significantly improved performance.

Table 4. Results for Berkeley motion segmentation benchmark

	Density	overall error	average error	over-segmentation rate	extracted objects	Total Time(s)
BM	1.03%	7.86%	28.76%	**1.35**	**30**	**9323**
OB	1.03%	5.68%	24.74%	1.48	**30**	434545
HOSC	1.03%	5.46%	22.57%	2.1	29	18339
MCNC(k_m)	1.03%	14.9%	29.02%	1.7	25	15971
MCNC	1.03%	**5.21**%	**20.02**%	1.75	29	16115

6. Acknowledgment

This work was supported by the Australian Research Council grants **DP130102524** and **LP160100662**.

References

Agarwal, S., Branson, K., Belongie, S., 2006. Higher order learning with graphs, in: International Conference on Machine Learning.

Agarwal, S., Lim, J., Zelnik-Manor, L., Perona, P., Kriegman, D., Belongie, S., 2005. Beyond pairwise clustering, in: Computer Vision and Pattern Recognition, IEEE.

Andrieu, C., de Freitas, N., Doucet, A., Jordan, M., 2003. An introduction to MCMC for machine learning. Machine Learning 50, 5–43.

Bab-Hadiashar, A., Gheissari, N., 2006. Range image segmentation using surface selection criterion. Transactions on Image Processing 15.

Bab-Hadiashar, A., Hoseinnezhad, R., 2008. Bridging parameter and data spaces for fast robust estimation in computer vision, in: International Conference on Digital Image Computing: Techniques and Applications, IEEE. pp. 1–8.

Bab-Hadiashar, A., Suter, D., 1999. Robust segmentation of visual data using ranked unbiased scale estimate. Robotica 17, 649–660.

Balakrishnan, S., Xu, M., Krishnamurthy, A., Singh, A., 2011. Noise thresholds for spectral clustering, in: Neural Information Processing Systems, IEEE. pp. 954–962.

Boykov, Y., Veksler, O., Zabih, R., 2001. Fast approximate energy minimization via graph cuts. Transactions on Pattern Analysis and Machine Intelligence 23, 1222–1239.

Brox, T., Malik, J., 2010. Object segmentation by long term analysis of point trajectories, in: Europian Conference on Computer Vision, IEEE.

Cabral, R., De La Torre, F., Costeira, J.P., Bernardino, A., 2013. Unifying nuclear norm and bilinear factorization approaches for low-rank matrix decomposition, in: International Conference on Computer Vision, pp. 2488–2495.

Candès, E.J., Li, X., Ma, Y., Wright, J., 2011. Robust principal component analysis? Journal of the ACM (JACM) 58, 11.

Chen, G., Lerman, G., 2009a. Motion segmentation by scc on the hopkins 155 database, in: Computer Vision Workshops (ICCV Workshops), 2009 IEEE 12th International Conference on, IEEE. pp. 759–764.

Chen, G., Lerman, G., 2009b. Spectral curvature clustering (SCC). International Journal of Computer Vision 81, 317–330.

Chin, T.J., Wang, H., Suter, D., 2009a. The ordered residual kernel for robust motion subspace clustering, in: Neural Information Processing Systems, pp. 333–341.

Chin, T.J., Wang, H., Suter, D., 2009b. Robust fitting of multiple structures: The statistical learning approach, in: Computer Vision, 2009 IEEE 12th International Conference on, IEEE. pp. 413–420.

Delong, A., Osokin, A., Isack, H.N., Boykov, Y., 2012. Fast approximate energy minimization with label costs. International journal of computer vision 96, 1–27.

Elhamifar, E., Vidal, R., 2013. Sparse subspace clustering: Algorithm, theory, and applications. Transactions on Pattern Analysis and Machine Intelligence 35, 2765–2781.

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

Florescu, L., Perkins, W., 2016. Spectral thresholds in the bipartite stochastic block model, in: Conference on Learning Theory, pp. 943–959.

Frank, M., Chehreghani, M.H., Buhmann, J.M., 2011. The minimum transfer cost principle for model-order selection, in: Gunopulos, D., Hofmann, T., Malerba, D., Vazirgiannis, M. (Eds.), Machine Learning and Knowledge Discovery in Databases, Springer Berlin Heidelberg, Berlin, Heidelberg. pp. 423–438.

Georghiades, A., Belhumeur, P., Kriegman, D., 2001. From few to many: illumination cone models for face recognition under variable lighting and pose. Transactions on Pattern Analysis and Machine Intelligence 23, 643–660.

Ghoshdastidar, D., Dukkipati, A., 2017. Uniform hypergraph partitioning: provable tensor methods and sampling techniques. Journal of Machine Learning Research 18, 1–41.

Hartley, R., Zisserman, A., 2003. Multiple view geometry in computer vision. Cambridge university press.

Hoseinnezhad, R., Bab-Hadiashar, A., Suter, D., 2006. Finite sample bias of robust scale estimators in computer vision problems. Advances in Visual Computing , 445–454.

Jain, S., Govindu, V.M., 2013. Efficient higher-order clustering on the grassmann manifold, in: International Conference on Computer Vision, IEEE.

Kim, E., Lee, M., Oh, S., 2016. Robust elastic-net subspace representation. Transactions on Image Processing .

Lee, K.M., Meer, P., Park, R.H., 1998. Robust adaptive segmentation of range images. Transactions on Pattern Analysis and Machine Intelligence 20, 200–205.

Li, B., Zhang, Y., Lin, Z., Lu, H., 2015. Subspace clustering by mixture of gaussian regression, in: Computer Vision and Pattern Recognition.

Liu, G., Lin, Z., Yan, S., Sun, J., Yu, Y., Ma, Y., 2013. Robust recovery of subspace structures by low-rank representation. Transactions on Pattern Analysis and Machine Intelligence 35, 171–184.

Liu, G., Xu, H., Tang, J., Liu, Q., Yan, S., 2016. A deterministic analysis for lrr. Transactions on Pattern Analysis and Machine Intelligence 38, 417–430.

Liu, H., Yan, S., 2012. Efficient structure detection via random consensus graph, in: Computer Vision and Pattern Recognition, IEEE.

Ochs, P., Malik, J., Brox, T., 2014. Segmentation of moving objects by long term video analysis. Transactions on Pattern Analysis and Machine Intelligence 36, 1187–1200.

Pham, T.T., Tat-Jun, C., Jin, Y., Suter, D., 2014. The random cluster model for robust geometric fitting. Transactions on Pattern Analysis and Machine Intelligence 36, 1658–1671.

Poling, B., Lerman, G., 2014. A new approach to two-view motion segmentation using global dimension minimization. International Journal of Computer Vision 108, 165–185.

Purkait, P., Chin, T.J., Sadri, A., Suter, D., 2017. Clustering with hypergraphs: the case for large hyperedges. Transactions on Pattern Analysis and Machine Intelligence 39, 1697–1711.

Rao, S., Yang, A., Sastry, S.S., Ma, Y., 2010. Robust algebraic segmentation of mixed rigid-body and planar motions from two views. International Journal of Computer Vision 88, 425–446.

Rousseeuw, P.J., Leroy, A.M., 2005. Robust regression and outlier detection. volume 589. John Wiley & Sons.

Sadri, A., Tennakoon, R., Hoseinnezhad, R., Bab-Hadiashar, A., 2016. Mcmc based sampling technique for robust multi-model fitting and visual data segmentation, in: International Conference on Image Processing Theory Tools and Applications (IPTA), IEEE. pp. 1–6.

Shi, J., Malik, J., 2000. Normalized cuts and image segmentation. Transactions on Pattern Analysis and Machine Intelligence 22, 888–905.

Swendsen, R.H., Wang, J.S., 1987. Nonuniversal critical dynamics in monte carlo simulations. Physical review letters 58, 86.

Tat-Jun, C., Jin, Y., Suter, D., 2012. Accelerated hypothesis generation for multistructure data via preference analysis. Transactions on Pattern Analysis and Machine Intelligence 34, 625–638.

Tennakoon, R.B., Bab-Hadiashar, A., Cao, Z., Hoseinnezhad, R., Suter, D., 2016. Robust model fitting using higher than minimal subset sampling.

Transactions on Pattern Analysis and Machine Intelligence 38, 350–362.
Tjelmeland, H., Hegstad, B.K., 2001. Mode jumping proposals in mcmc. Scandinavian Journal of Statistics 28, 205–223.
Toldo, R., Fusiello, A., 2008. Robust Multiple Structures Estimation with J-Linkage. Springer Berlin Heidelberg. volume 5302 of *Lecture Notes in Computer Science*. book section 41. pp. 537–547.
Torr, P.H., Zisserman, A., 2000. Mlesac: A new robust estimator with application to estimating image geometry. Computer Vision and Image Understanding 78, 138–156.
Torr, P.H.S., Murray, D.W., 1997. The development and comparison of robust methods for estimating the fundamental matrix. International Journal of Computer Vision 24, 271–300.
Tran, Q.H., Chin, T.J., Chojnacki, W., Suter, D., 2014. Sampling minimal subsets with large spans for robust estimation. International Journal of Computer Vision 106, 93–112.
Tron, R., Vidal, R., 2007. A benchmark for the comparison of 3-D motion segmentation algorithms, in: Computer Vision and Pattern Recognition, IEEE.
Vidal, R., 2011. Subspace clustering. Signal Processing Magazine 28, 52–68.
Vincent, E., Laganiére, R., 2001. Detecting planar homographies in an image pair, in: Proceedings of the 2nd International Symposium on Image and Signal Processing and Analysis, pp. 182–187.
Wang, H., Chin, T.J., Suter, D., 2012. Simultaneously fitting and segmenting multiple-structure data with outliers. Transactions on Pattern Analysis and Machine Intelligence 34, 1177–1192.
Wang, H., Suter, D., 2004. Robust adaptive-scale parametric model estimation for computer vision. Transactions on Pattern Analysis and Machine Intelligence 26, 1459–1474.
Wang, H., Xiao, G., Yan, Y., Suter, D., 2015. Mode-seeking on hypergraphs for robust geometric model fitting, in: International Conference on Computer Vision, IEEE. pp. 2902–2910.
Wang, H., Yan, Y., Suter, D., et al., 2018. Searching for representative modes on hypergraphs for robust geometric model fitting. IEEE Transactions on Pattern Analysis and Machine Intelligence .
Wang, J., Shi, D., Cheng, D., Zhang, Y., Gao, J., 2016. Lrsr: Low-rank-sparse representation for subspace clustering. Neurocomputing .
Wong, H.S., Chin, T.J., Yu, J., Suter, D., 2011. Dynamic and hierarchical multi-structure geometric model fitting, in: International Conference on Computer Vision, IEEE. pp. 1044–1051.
Yu, J., Chin, T.J., Suter, D., 2011. A global optimization approach to robust multi-model fitting, in: Computer Vision and Pattern Recognition, IEEE. pp. 2041–2048.
Zhang, T., Szlam, A., Wang, Y., Lerman, G., 2012. Hybrid linear modeling via local best-fit flats. International Journal of Computer Vision 100, 217–240.
Zhou, D., Huang, J., Schölkopf, B., 2006. Learning with hypergraphs: Clustering, classification, and embedding, in: Neural Information Processing Systems, IEEE.