Branch-and-Bound for Model Selection and Its Computational Complexity

Ninad Thakoor, *Member*, *IEEE*, and Jean Gao, *Senior Member*, *IEEE*

Abstract—Branch-and-bound methods are used in various data analysis problems, such as clustering, seriation and feature selection. Classical approaches of branch-and-bound based clustering search through combinations of various partitioning possibilities to optimize a clustering cost. However, these approaches are not practically useful for clustering of image data where the size of data is large. Additionally, the number of clusters is unknown in most of the image data analysis problems. By taking advantage of the spatial coherency of clusters, we formulate an innovative branch-and-bound approach, which solves clustering problem as a model-selection problem. In this generalized approach, cluster parameter candidates are first generated by spatially coherent sampling. A branch-and-bound search is carried out through the candidates to select an optimal subset. This paper formulates this approach and investigates its average computational complexity. Improved clustering quality and robustness to outliers compared to conventional iterative approach are demonstrated with experiments.

Index Terms—Clustering, segmentation, combinatorial optimization, branch-and-bound, model selection.

1 Introduction

Clustering is a popular unsupervised learning technique applied in areas, such as data mining [1], image processing [2], pattern recognition [3], and bioinformatics [4]. Clustering organizes the data meaningfully by grouping similar data points in a cluster and splitting dissimilar data points in different clusters. Normally, the similarity between data points is assessed with the help of a dissimilarity or distance measure, such as euclidian distance. Classical clustering method by k-means divides the data into k partitions so that the sum of square-error between cluster means and the data in the corresponding cluster is minimized. The k-means procedure falls under the category of partitioning methods for clustering. Hierarchical methods of clustering create a hierarchy of clusters. In an agglomerative hierarchy, smaller clusters are merged to construct larger clusters, starting from individual data points leading to a single cluster. Under a divisive hierarchy, larger clusters are divided to form smaller clusters. The divisive strategy starts with a single cluster, and finally, each data point corresponds to a cluster. The desired clustering can be generated by cutting the hierarchy at a predetermined depth. Density-based clustering approaches grow clusters based on density of data points in the clustering space. Unlike partitioning approaches, the density-based approaches can detect clusters of arbitrary shapes. In the model-based clustering approach, each cluster is represented by a parametric model [5]. A data point is assigned to the cluster whose model explains the data points best. A model, such as Gaussian mixture model (GMM) or hidden Markov model (HMM), is defined a priori based on the domain knowledge. Han and Kamber [1] give detailed descriptions of contemporary techniques, which follow the aforementioned clustering paradigms.

Clustering problems involving image data, such as image segmentation, motion segmentation, stereo disparity segmentation, and structure-and-motion segmentation, can be expressed as model-based clustering problems. For model-based clustering problems, to assign a data point data to an appropriate cluster, the cluster parameters should be known. On the other hand, the cluster parameters can be computed only if the cluster assignments are known. This "chicken-and-egg" dilemma leads to an iterative formulation for model-based clustering methods similar to expectation maximization (EM) algorithm [6]. Clustering aims to optimize an assignment cost to achieve a (locally) optimal solution. If the number of clusters is increased, generally the cost for the same data reduces. The degenerate case for this happens when one cluster corresponds to one data point and the corresponding clustering cost is zero. Clearly, such a scenario is undesirable. Thus the clustering cost must be penalized for additional clusters. A variety of model-selection methods exist, which incorporate this idea [7]. Note that the term "model" in model selection refers to the ensemble of the number of clusters and the parametric models for these clusters. To apply model selection to clustering, candidate models are generated sequentially by varying the number of clusters, and the best model according to a model-selection criterion is selected. For the image data, the iterative and sequential problem of model selection can be simplified to a one-step optimization by using the knowledge that the clusters formed in an image are spatially coherent. The candidates for cluster

- *N. Thakoor is with the Center for Research in Intelligent Systems, University of California, Riverside, Engineering Unit II EBU2, Room 216, Riverside, CA 92521-0425. E-mail: ninad.thakoor@ucr.edu.*
- *J. Gao is with the Computer Science and Engineering Department, University of Texas at Arlington, Box 19015, Arlington, TX 76019. E-mail: gao@uta.edu.*

Manuscript received 28 Jan. 2009; revised 22 Oct. 2009; accepted 30 Jan. 2010; published online 30 Aug. 2010.
Recommended for acceptance by X. Zhu.
For information on obtaining reprints of this article, please send e-mail to: tkde@computer.org, and reference IEEECS Log Number TKDE-2009-01-0043.
Digital Object Identifier no. 10.1109/TKDE.2010.156.

parameters can be generated by sampling spatially coherent image data points. Once the candidates are known, a subset of these candidates can be selected by optimizing a model-selection criterion. This transforms the problem into a one-step model-selection problem.

This idea is utilized in structure-and-motion segmentation approaches proposed recently [8], [9], [10]. Schindler and Suter [9] carry out multibody structure-and-motion segmentation from two camera views. After correspondences are established between the two views, the correspondences are grouped together based on the spatial coherence. From each group of correspondences, a hypothesis for underlying structure-and-motion is generated using random sample consensus (RANSAC) [11]. A geometrically robust information criterion (GRIC) [12] is optimized to select the best subset of candidates. The optimization of the criterion is carried out with Tabu search [13]. In [8], Li solves the two-view motion segmentation problem starting from a set of candidate motions generated by applying spatial coherence, prior distribution, chirality constraints etc. The segmentation problem is then formed as a facility location problem and solved with linear programming relaxation [14], [15]. In our previous work [10], we generate candidates for structure-and-motion by applying local sampling followed by nonmaximal suppression. We optimize the Bayesian information criterion (BIC) [16] with a branch-and-bound strategy.

The branch-and-bound approach [17] to global optimization splits the optimization problem into smaller subproblems and for these subproblems, upper and/or lower bounds on the cost function are estimated. These bounds are used to eliminate the subproblems that would not lead to an optimal solution. The subproblems that survive are further divided and the bound calculation is continued till all the subproblems are explored. The branch-and-bound procedure is applied in diverse areas, such as feature selection [17], [18], [19], image registration [20], rate-distortion based coding [21], job scheduling [22], [23] and clustering [17], [18], [24], where the number of clusters is known. Our previous work in [25], [26] explores branch-and-bound for greedy maximum variance clustering of stereo disparity with unknown number of clusters.

In this paper, we outline a general framework based on [10] for multihypothesis branch-and-bound model selection and analyze its average computational complexity. The average computational complexity of the branch-and-bound algorithms, which search over random trees, has been explored by a number of researchers [27], [28], [29], [30], [31], [32]. The term "random" applies to the structure of the tree and weights of the tree edges in general. However, for the multihypothesis branch-and-bound model-selection problem, the structure of the tree is deterministic, and only the weights of the tree edges are random. Thus, a separate treatment for the complexity of the problem becomes necessary.

This paper is organized as follows: Section 2 formulates a generalized multihypothesis branch-and-bound model-selection problem. Section 3 develops the framework to estimate the expected complexity of the branch-and-bound search for the problem. The computation of various quantities involved in the estimation of complexity of the algorithm is discussed in Section 4. Section 5 presents the results achieved by the model-selection process and its expected complexity. Conclusions are presented in Section 6.

2 Formulation

This section first formulates model-based clustering as a model-selection problem. Later, a branch-and-bound solution for the problem is devised, and application of the model selection to an image processing problem is discussed.

2.1 The Problem

Consider a set of M observations $\mathbf{Y}$, such as image intensity/color, video motion or stereo disparity,

$$\mathbf{Y} = \{\mathbf{y}_1, \mathbf{y}_2, \ldots, \mathbf{y}_M\}.$$

The corresponding cluster memberships for the observations can be denoted by $L = \{l_1, l_2, \ldots, l_M\}$. If an observation $\mathbf{y}_j$ belongs to a cluster k then $l_j = k$ and vice versa. Under the model-based clustering paradigm, the data can be explained with one of the K clusters with parameters $\{\Theta_1, \Theta_2, \ldots, \Theta_K\}$, respectively. A generic model for estimating observations from the cluster parameters and the memberships can be given as [33]

$$\mathbf{y}_j = g(\mathbf{x}_j; \Theta_{l_j}) + \mathbf{v}_j, \quad j = 1, 2, \ldots, M. \tag{1}$$

In this model, $\mathbf{X} = \{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_M\}$ are the independent variables on which the observations $\mathbf{Y}$ depend (these can be quantities, such as spatial locations for images or time instances for time-series data). If the data do not have spatial or temporal relationship, which is true for many clustering problems, the independent variables would not appear in the model [1]. $g(\mathbf{x}; \Theta_f)$ can be a linear or nonlinear function or any process that can compute observation $\mathbf{y}$ from $\mathbf{x}$ given parameters Θ_f. $\mathbf{V} = \{\mathbf{v}_1, \mathbf{v}_2, \ldots, \mathbf{v}_M\}$ is the noise corrupting the observation, which is generally assumed to follow a zero mean independent Gaussian distribution. The above model appears in the missing data problems as well [2]. According to the missing data formulation, the observations $\mathbf{Y}$ are available and the cluster memberships L are missing.

The model-based clustering problems have two unknown quantities, the cluster parameters $\Theta = \{\Theta_1, \Theta_2, \ldots, \Theta_K\}$ and the memberships L. Given the memberships L, the maximum likelihood estimate for the parameters Θ is given by

$$\hat{\Theta} = \arg\max_{\Theta} \Pr(L|\Theta, \mathbf{Y}). \tag{2}$$

Given the parameters Θ, the maximum likelihood estimate for the memberships L is given by

$$\hat{L} = \arg\max_{L} \Pr(\Theta|L, \mathbf{Y}). \tag{3}$$

After simplification

$$\hat{\Theta}_i = \arg\min_{\Theta} \sum_{\forall j \leftrightarrow l_j = i} \|\mathbf{y}_j - g(\mathbf{x}_j; \Theta)\|^2, \tag{4}$$

$$\hat{l}_j = \arg\min_{i} \|\mathbf{y}_j - g(\mathbf{x}_j; \Theta_i)\|^2. \tag{5}$$

Here, $i = 1, 2, \ldots, K$ and $j = 1, 2, \ldots, M$. Conventional methods for model-based clustering iterate between estimation of Θ and L till one or the other converges. They additionally require that the number of clusters K is known a priori. This requirement is unrealistic for most clustering problems. Thus the number of clusters has to be varied to select the optimal number of clusters. This process is called model-selection. The model-selection constitutes to the choice of K and corresponding Θ. Since the likelihood of the model increases as more clusters are added, a criterion which penalizes the likelihood with increasing clusters, such as Akaike Information Criterion (AIC) or Bayesian Information Criterion (BIC) is used to select the optimal number of clusters [34]. If the number of free parameters per cluster is N

$$\mathrm{AIC}(\Theta) = -2\log(\mathcal{L}_\Theta) + 2KN, \tag{6}$$

$$\mathrm{BIC}(\Theta) = -2\log(\mathcal{L}_\Theta) + KN\log(M). \tag{7}$$

Or, a generalized model-selection criterion can be given by

$$\mathcal{C}(\Theta) = -\log(\mathcal{L}_\Theta) + \alpha \cdot K, \tag{8}$$

where α is a positive constant and $\mathcal{L}_\Theta$ gives the likelihood of the data for a model Θ. The model-selection thus leads to a sequential process which follows the iterative clustering.

On the other hand, if a linearly ordered set of N_c candidates $C = \{C_1, C_2, \ldots, C_{N_c}\}$ for cluster parameters Θ_i is given, we can choose a subset Θ which optimizes the model-selection criterion given in (8). The likelihood of the data is proportional to the sum of the residuals for the current model and is given by

$$\log(\mathcal{L}_\Theta) = -\frac{1}{2}M\log\left(\frac{\mathrm{SSD}(\Theta)}{M}\right) + \mathrm{Constant}, \tag{9}$$

where,

$$\mathrm{SSD}(\Theta) = \sum_{j=1}^{M} \min_{C_i \in \Theta} r_j(C_i)$$

and

$$r_j(C_i) = \|\mathbf{y}_j - g(\mathbf{x}_j; C_i)\|^2 \tag{10}$$

are the residuals for jth observation for the candidate C_i.

After subsuming the constants, the model-selection criterion becomes

$$\mathcal{C}(\Theta) = M\log\left(\frac{\mathrm{SSD}(\Theta)}{M}\right) + \alpha \cdot K, \tag{11}$$

which is to be minimized by selecting $\Theta \subset C$, where K is the number of candidates in the subset Θ.

There are 2^{N_c} possible solutions for this subset selection problem. Even for a moderate value of N_c, an exhaustive search is computationally expensive. However, the nature of the problem allows us to use a branch-and-bound approach to obtain an optimal solution for the problem in reasonable time for the practical problems.

2.2 Outliers

Due to their iterative nature, the model-based clustering problems are especially sensitive to the outliers. In addition to the candidates above, an alternative candidate C_0 can be added to account for the outliers in the data. According to this candidate, the residual for all the observations are assumed to be

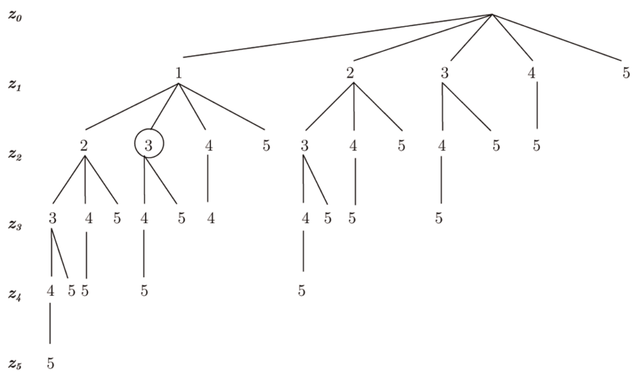

Fig. 1. Solution tree for $N_c = 5$ and an additional candidate for outliers.

$$r_j(C_0) = r_0. \tag{12}$$

With this setup, if the residuals for the jth observation are greater that r_0 according to the candidates $C_1, C_2, \ldots, C_{N_c}$, the observation is treated as an outlier.

2.3 Branch-and-Bound Algorithm

All the possible solutions of the model-selection problem can be represented by a rooted tree. It is important that every solution is listed only once in the tree to avoid unnecessary computations. This can be ensured by creating child nodes that are different than:

- left siblings,
- ancestors, and
- left siblings of ancestors.

One simple way of generating such a solution tree for five candidates is shown in Fig. 1 with an additional candidate claiming that the data point is an outlier. For ease of representation, each tree node is labeled by the index of the most recently added candidate instead of listing indices of all the candidates in the subset. The subset of candidates corresponding to a node is given by a walk from the root node to the node under consideration. The circled node includes candidates $\{C_0, C_1, C_3\}$. The candidate C_0 indicates that the data point is outlier, i.e., the point does not belong to any of the cluster parameter candidates. Note that, in the solution tree, the node label z_i increases monotonically with the tree depth and the node label is lesser than its right sibling's label. These two conditions ensure that the rule stated above to generate the child nodes is followed.

Each node of the tree represents a subset of candidates and two hypotheses, one that *the subset gives the optimal solution* of the model-selection problem and the other that *the subset is a partial solution* to the problem. A partial solution is a subset of optimal solution, i.e., adding more candidates to a partial solution would lead to an optimal solution. Note that if a partial solution hypothesis for a node is rejected, then none of the child nodes of the node can be a partial solution. A branch-and-bound algorithm aims to validate the hypotheses presented by all the tree nodes explicitly or implicitly. As the algorithm is a search strategy, at any point of search it maintains the best search result till that point. The best search result, which is nothing but lowest cost encountered in the search $\mathcal{C}^*$, can be used to validate the optimal solution hypothesis. If the modeling cost $\mathcal{C}(\Theta)$ for current node is higher than $\mathcal{C}^*$, the current node cannot be an optimal solution. The partial solution hypothesis can be validated with a lower bound on the modeling cost of all the solutions leading from current subset of candidates. All the solutions leading from current subset are represented by child nodes of the current node. If the lower bound on child nodes is higher than $\mathcal{C}^*$, then the partial solution hypotheses for the node as well as its child nodes can be safely rejected.

The lower bound on the first term of (11) corresponds to the lower bound on $\mathrm{SSD}(\Theta)$. The lower bound on $\mathrm{SSD}(\Theta)$ is reached when a lower bound on residuals of individual data points is achieved. With candidate subset Θ, the residue for jth observation is given by

$$\min_{C_i \in \Theta} r_j(C_i).$$

If the subset Θ is hypothesized as a partial solution, one can add a subset of candidates Θ^+ to existing subset Θ to form a solution Θ'. To find the lower bound on the residual, one must include all the possible candidates to build Θ^+. By observing the structure of the solution tree, it is clear that the candidates $C_k \in \Theta^+$ must have $C_k > \max(\Theta)$. Note that, here the hypotheses are compared by their indices. Thus the lower bound on the residual for jth observation is given by

$$\begin{aligned}\min_{C_i \in \Theta'} r_j(C_i) &= \min\left(\min_{C_i \in \Theta} r_j(C_i), \min_{C_k \in \Theta^+} r_j(C_k)\right)\\ &= \min\left(\min_{C_i \in \Theta} r_j(C_i), \min_{\forall C_k > \max(\Theta)} r_j(C_k)\right).\end{aligned}$$

Thus the lower bound on $\mathrm{SSD}(\Theta)$ is given by

$$\mathrm{SSD}_{\mathrm{Lower}}(\Theta) = \sum_{j=1}^{M} \min\left(\min_{C_i \in \Theta} r_j(C_i), \min_{\forall C_k > \max(\Theta)} r_j(C_k)\right). \quad (13)$$

As at least one more candidate has to added to the partial solution of size K to reach an optimal solution, the lower bound on second term of (11) is given by $(K+1)$. The lower bound on the hypotheses leading from Θ is thus given by

$$\mathcal{C}_{\mathrm{Lower}}(\Theta) = M \log\left(\frac{\mathrm{SSD}_{\mathrm{Lower}}(\Theta)}{M}\right) + \alpha \cdot (K+1). \quad (14)$$

From the cost function (11) and the bound (14), the branch-and-bound algorithm can be implemented.

We adapt a generic queue-based implementation of the branch-and-bound procedure from [28]. With the queue-based implementation, the solution tree can be explored using various search strategies. We list a few of these methods here [28]:

- Best bound first (BBF),
- Ordered depth first,
- Generation order depth first,
- Ordered breadth first, and
- Generation order breadth first.

These methods prioritize the search of nodes in different ways, as suggested by their names. As the BBF search algorithm has the least time complexity, we choose the BBF search for our implementation and the complexity analysis. Following gives an implementation of BBF search for the model-selection problem.

Best bound first branch-and-bound procedure:

```
1. Insert hypotheses for the root node in the priority queue $Q$.
2. Set the optimal cost $\mathcal{C}^* = \infty$.
3. Pop the first hypothesis from $Q$ which is nothing but the least cost hypothesis.
4. If the popped hypothesis is an optimal cost hypothesis then terminate the algorithm.
5. For the child nodes of the popped hypothesis, validate and insert hypotheses in $Q$. For all the child nodes:
   a. Validate the optimal solution hypothesis.
      • Compute cost $\mathcal{C}(\Theta)$ for the node.
      • If $\mathcal{C}(\Theta) < \mathcal{C}^*$,
         - Insert an optimal solution hypothesis in $Q$ with priority $1/\mathcal{C}(\Theta)$.
         - Delete hypotheses after the location where above hypothesis was inserted.
         - Set $\mathcal{C}^* = \mathcal{C}(\Theta)$.
   b. Validate the partial solution hypothesis, if the node is an internal node.
      • Calculate bound $\mathcal{C}_{\mathrm{Lower}}(\Theta)$.
      • If $\mathcal{C}_{\mathrm{Lower}}(\Theta) < \mathcal{C}^*$, insert a partial solution hypothesis in $Q$ with priority $1/\mathcal{C}_{\mathrm{Lower}}(\Theta)$.
6. Go to Step 3.
```

2.4 Application to Multibody Structure-and-Motion Segmentation

The multibody structure-and-motion (MSaM) segmentation problem groups image correspondences according to coherent structure and motion. The set of M image correspondences in this case be given by $\mathbf{Y} = \{\mathbf{y}_1, \mathbf{y}_2, \ldots, \mathbf{y}_M\}$ and $\mathbf{X} = \{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_M\}$, where $\mathbf{y}_j$ and $\mathbf{x}_j$ are image coordinates of jth correspondence. If the image sequence contains K moving rigid objects, jth image correspondence is related as

$$\mathbf{y}_j^T F_{l_j} \mathbf{x}_j = 0. \quad (15)$$

Here $F = \{F_1, F_2, \ldots, F_K\}$ corresponds to fundamental matrices [11] of K rigid bodies. We can rewrite (15) as a generic model shown in Section 2 as

$$\mathbf{y}_j = f(\mathbf{x}_j; F_{l_j}) + \mathbf{v}_j. \quad (16)$$

The function f here corresponds to the triangulation method [11], which can estimate $\mathbf{y}_j$ given $\mathbf{x}_j$ and the corresponding fundamental matrix. Due to the geometric nature of the problem, maximum likelihood estimation uses geometric distance measure such as reprojection error

$$\delta(\mathbf{y}_j, \mathbf{x}_j, C_i)^2 = \|\mathbf{y}_j - \hat{\mathbf{y}}_j\|^2 + \|\mathbf{x}_j - \hat{\mathbf{x}}_j\|^2, \quad (17)$$

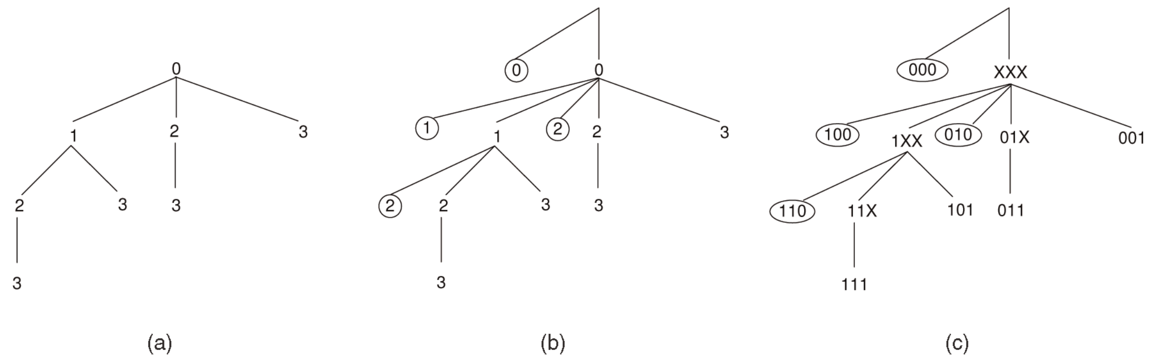

Fig. 2. (a) Original branch-and-bound tree for $N_c = 3$, (b) its edge-weighted equivalent, and (c) coding for the tree nodes.

where $\hat{\mathbf{y}}_j$ and $\hat{\mathbf{x}}_j$ are estimated correspondences by candidate C_i given by

$$\hat{\mathbf{y}}_j = f(\mathbf{x}_j; C_i),$$
$$\hat{\mathbf{x}}_j = f(\mathbf{y}_j; C_i^T).$$

The candidates for the fundamental matrix can be generated by spatially local sampling of the correspondences and estimating the fundamental matrix from the sample. If these candidates are $C = \{C_1, C_2, \ldots, C_{N_c}\}$, the residual for jth correspondence for candidate C_i is given by

$$r_j(C_i) = \delta(\mathbf{y}_j, \mathbf{x}_j, C_i)^2. \tag{18}$$

With these residuals, one can proceed to generic implementation of the branch-and-bound strategy for model selection outlined at the beginning of the section. Section 5.2 reports results for this implementation.

3 Branch-and-Bound as an Edge-Weighted Tree Search Problem

The worst case computational complexity of any branch-and-bound search algorithm is same as the complexity of the brute force search. However, a branch-and-bound approach is generally applied to an **NP** hard global optimization problem for which the worst case complexity gives a little or no insight into the performance of the approach. In such a situation, the average or expected computational complexity would give a more reasonable estimate of the performance of the approach. In this section, we formulate a framework to estimate the expected computational complexity for the branch-and-bound model-selection approach presented in previous section.

Under the current formulation, each node of the solution tree represents two hypotheses, the optimal solution hypothesis and the partial solution hypothesis. This gives rise to a binomial tree [35] of order N_c as the representation of the model-selection problem (see Fig. 2a). However, in a typical tree search problem only the leaf nodes can represent an optimal solution. To incorporate this, we modify the original tree structure and add a "twin" node to each internal node of the binomial tree. This updated tree structure is shown in Fig. 2b and will be used for computational complexity analysis. The circled nodes are the newly added twin nodes. In the updated tree, the leaf nodes (i.e., leaf nodes from the original tree and newly added twin nodes) represent the optimal solution hypotheses and the internal nodes represent the partial solution hypotheses.

To represent each hypothesis uniquely, we devise a representation for each hypothesis with symbols $\{0, 1, \mathrm{X}\}$ ("zero," "one," and "undetermined"). In this N_c elements' wide representation, if a hypothesis includes a candidate C_i then $(N_c - i)$th element of the representation is one and if the hypothesis does not include the candidate C_i then $(N_c - i)$th element is zero. For the partial solution hypothesis represented by internal nodes, an additional symbol X is used. The symbol X indicates a candidate that can be included in a solution later in the search. The $(N_c - i)$th element of the representation is set to X, if any child nodes can include the candidate C_i. One can quickly get the "twin" node of an internal node by replacing Xs with zeros.

The cost associated with a leaf node is the cost of the optimal solution hypothesis $\mathcal{C}(\Theta)$. On the other hand, the cost associated with an internal node is the cost of the partial solution hypothesis $\mathcal{C}_{\mathrm{Lower}}(\Theta)$. From (11) and (14), one can conclude that the cost of a node is lower than its child nodes. This also means that each edge of the updated tree has a nonzero positive weight associated with it. The cost of reaching a node can be computed by adding weights of all the edges along the path from the node to the root of the tree. Note that our formulation does not require explicit computation of the edge weights as the cost of reaching a node can be directly computed from (11) or (14). The least cost leaf node in the edge-weighted tree corresponds to the optimal solution for the branch-and-bound process. Thus, our branch-and-bound approach can be seen as a least cost leaf search problem for the updated edge-weighted tree.

3.1 Average Complexity

To estimate the average complexity, we concentrate on the BBF approach which explores the least number of nodes before it reaches the optimal solution [28]. In a BBF implementation, every time the least cost node is popped out of the priority queue Q. Child nodes of the currently popped node are inserted in the queue Q. The priory of a node is set inversely proportional to its cost. For the edge-weighted tree, the first leaf node popped from Q during BBF search is optimal [28]. This also means that the complexity, i.e., the number of nodes popped out before the optimal node, is same as the number of internal nodes which have their costs less than the optimal cost. Additionally, the optimal node has the cost less than all the other leaf nodes by definition.

Let T denote the set of all the leaf nodes of the tree and I denote the set of all internal nodes of the tree. The optimality probability of the node i, $\Pr_o(i)$, denotes the probability that the node i is optimal, i.e., it has the least cost among the leaf nodes

$$\Pr_o(i) = \prod_{\forall j \in T \setminus i} \Pr(\mathcal{C}(i) < \mathcal{C}(j)). \tag{19}$$

The cost probabilities $\Pr(\mathcal{C}(i) < \mathcal{C}(j))$ are probabilities of comparison of the sum of edge weights leading to nodes i and j. These can be seen as probabilities of comparison between two sums of edge weights, and thus can be given by $\Pr_T(S_m < S_n)$. Here S_m and S_n are the sums of m and n edge weights, respectively $(1 \le n \le N_c, 1 \le m \le N_c)$. Note that it is not necessarily true that $m = depth(i)$ and $n = depth(j)$. One has to remove the common edges along the path to the root node from the node depth to get values of m and n. If the number of common edges is l then $m = depth(i) - l$ and $n = depth(j) - l$. We define nodes i and j to have a relationship of order (m, n). In graph theory terms, the relationship between the two nodes can be seen as the simple path between them, and $(m + n)$ gives the length of the simple path.

Due to the recursive structure of the tree, the weight relationships repeat themselves. Thus $\Pr_o(i)$ can be written as

$$\Pr_o(i) = \prod_{m=1}^{N_c} \prod_{n=1}^{N_c} \Pr_T(S_m < S_n)^{O_i(m,n)}. \tag{20}$$

Here O_i is the optimality matrix for the node i and its (m, n)th element indicates the number of times the relationship (m, n) (and hence the term $\Pr_T(S_m < S_n)$) appears in the computation of $\Pr_o(i)$.

The complexity for node i, $N(i)$ denotes the number of internal nodes explored by BBF search if the node i is optimal. When the node i is optimal, the internal node j is explored only if its cost is less than the cost of the optimal node i. Thus the complexity, when the node i is optimal, is

$$N(i) = \sum_{\forall j \in I} \Pr(\mathcal{C}_{\text{Lower}}(j) < \mathcal{C}(i)). \tag{21}$$

Similar to the optimality probability $\Pr_o(i)$, the complexity $N(i)$ of the node can be expressed as

$$N(i) = \sum_{m=1}^{N_c} \sum_{n=1}^{N_c} \Pr_I(S_n < S_m) \cdot R_i(m, n). \tag{22}$$

Here R_i is the complexity matrix for the node i and its (m, n)th element indicates the number of times the relationship (m, n) (and hence the term $\Pr_I(S_m < S_n)$) repeats in computation of $N(i)$. Note that different subscripts are used for probabilities $\Pr_T$ and $\Pr_I$, as the sums compared by these probabilities differ slightly. For $\Pr_T$, one of the weights in both sums is for an edge from an internal node to a leaf node while all the other weights are for edges between internal nodes. For $\Pr_I$, all the weights correspond to edges between internal nodes. If we assume that this difference is negligible, then

$$\Pr(S_m < S_n) = \Pr_T(S_m < S_n) = 1 - \Pr_I(S_n < S_m). \tag{23}$$

With the optimality probability $\Pr_o(i)$ and the complexity $N(i)$, the expected complexity $\overline{N}$ can be estimated as

$$\overline{N} = \frac{\sum_{\forall i \in T} \Pr_o(i) N(i)}{\sum_{\forall i \in T} \Pr_o(i)}. \tag{24}$$

The next section describes the computation of quantities involved in the estimation of the expected complexity.

4 Computing Cost Probabilities, Optimality Matrix and Complexity Matrix

The optimality matrix O_i and the complexity matrix R_i are different for nodes which do not have relationship $(1, 1)$ and the matrices change with the order of the tree as well. However, the cost probabilities $\Pr_T(S_m < S_n)$ and $\Pr_I(S_n < S_m)$ are only determined by the distribution of edge weights. This section first describes how to estimate these probabilities.

4.1 Cost Probabilities for Uniformly Distributed Edge Weights

This section estimates cost probabilities for an edge-weighted tree with edge weights uniformly distributed between $[0, 1]$. We start with the sum of m independently and uniformly distributed edge weights between $[0, 1]$, given by [36]

$$f_m(S_m) = \frac{1}{(m-1)!} \sum_{j=0}^{m} (-1)^j \binom{m}{j} [(S_m - j)_+]^{m-1}. \tag{25}$$

Here, $(\cdot)_+$ means positive part of $(\cdot)$. This can be written as

$$(\cdot)_+ = \frac{(\cdot) + |(\cdot)|}{2}.$$

$\Pr(S_m < S_n)$ when $m > n$ can be derived from $f_m(S_m)$ to be

$$\Pr(S_m < S_n) = \sum_{q=1}^{n} \sum_{k=0}^{q-1} \sum_{j=0}^{q-1} (-1)^{(k+j)} \binom{n}{k} \binom{m}{j} \left[\sum_{p=1}^{n} (-1)^{(p-1)} \frac{(x-k)^{(n-p)}}{(n-p)!} \frac{(x-j)^{(m+p)}}{(m+p)!} \right]_{(q-1)}^{q}. \tag{26}$$

Here,

$$[f(x)]_{(q-1)}^{q} = f(q) - f(q-1).$$

Refer to the appendix for the detailed derivation. $\Pr(S_m < S_n)$ when $m < n$ can simply computed as

$$\Pr(S_m < S_n) = 1 - \Pr(S_n < S_m).$$

Fig. 3 shows the plot of these probabilities.

4.2 Cost Probabilities by Sampling

For a typical model-selection problem, the distribution for the sums does not have a closed form solution, or it is

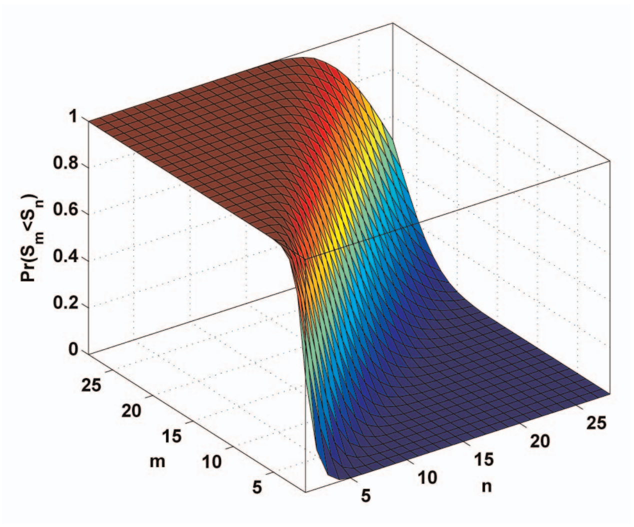

Fig. 3. $\Pr(S_m < S_n)$ for uniform iid random variables.

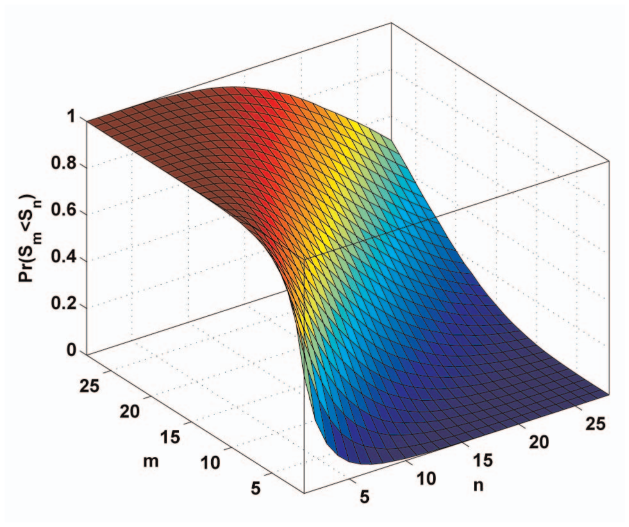

Fig. 4. $\Pr(S_m < S_n)$ for squared Gaussian iid random variables generated by sampling.

unknown. In such a case, a close approximation of $\Pr_T(S_n < S_m)$ and $\Pr_I(S_n < S_m)$ or $\Pr(S_n < S_m)$ can be generated with sampling. The sampling can be implemented as a simple process listed below.

Sampling:

- For all the possible combinations of m and n repeat following N_s times.
  - Generate a hypothesis Θ_1 of size m and compute its cost $\mathcal{C}(\Theta_1)$.
  - Generate a hypothesis Θ_2 of size n such that $\Theta_1 \cap \Theta_2 = \emptyset$ and compute its cost $\mathcal{C}(\Theta_2)$.
  - Compare the costs of the hypotheses, if $\mathcal{C}(\Theta_1) < \mathcal{C}(\Theta_1)$, $\Pr(S_m < S_n) = \Pr(S_m < S_n) + 1$.
- For all the possible combinations of m and n, normalize the probabilities $\Pr(S_m < S_n) = \Pr(S_m < S_n)/N_s$.

Fig. 4 shows $\Pr(S_m < S_n)$ generated with sampling when the edge weights are independent and identically distributed (iid) as squared zero mean Gaussian with unit standard deviation.

4.3 Computing Optimality Matrix

The optimality matrix O_i is computed by comparing each leaf node i with all the other leaf nodes. The edge-weighted tree can be seen as a binomial tree with an added "twin" node for all the internal nodes. Thus, recursive properties of the binomial tree can be used in computation of O_i. To compute O_i, we transform the edge-weighted tree back to the binomial tree by merging the twin nodes with the internal nodes and retaining the representation of the twin nodes after merging. Thus, the node representation is now binary.

Computation of O_i relies on the property that *a simple path between two nodes of a binomial tree includes the root node of only the smallest subtree, including both the nodes*. Note the following important properties before proceeding to compute O_i:

- Depth of a node, $d(i)$ is equal to the number of ones in the binary representation.
- Each node i belongs to a unique combination of binomial subtrees $T_0, T_1, \ldots, T_{d(i)}$ and location of ones in the representation indicates the order of binomial subtrees, e.g., the most significant bit indicates a binary subtree of order $N_c - 1$ and the least significant bit indicates a binomial subtree of order 0. Note that all the nodes belong to a subtree T_0 of order N_c.
- The number of nodes belonging to a subtree T_t at depth k is given by

$$N_t(k) = \begin{cases} \binom{T_t}{k-t}, & \text{if } 0 \leq k - t \leq T_t; \\ 0, & \text{otherwise}, \end{cases} \tag{27}$$

  where $k = 0, 1, 2, \ldots, N_c$ and $t = 0, 1, 2, \ldots, d(i)$.

Since a node i only belongs to subtrees $T_0, T_1, \ldots, T_{d(i)}$, to compute O_i, we have to analyze these subtrees alone. The binomial tree can be split into these subtrees and can be analyzed subtree by subtree, starting with the largest subtree T_0. For each subtree, we select the nodes which exclusively belong to the subtree under consideration. This can be done by removing the nodes belonging to the next largest subtree from the subtree under consideration. Finally, one has to offset the result of merging of the "twin" nodes. The merging leads to the relationships of the order $(m, 0)$ and $(0, n)$ which would have been of the order $(m + 1, 1)$ and $(1, n + 1)$ otherwise. Also, we have to remove the relationship $(0, 0)$ which corresponds to comparison of the node i with itself. The algorithm to compute the optimality matrix O_i follows:

1. Initialize $T = \{T_0, T_1, \ldots, T_d\} =$ the subtree membership of the node i, $d(i) =$ depth of the node i, set $O_i(1, 1) = -1$ and all the other elements of O_i equal to zero.
2. Set $t = 0$ such that the current subtree $T_t = T_0$.
3. For the subtree T_t, at each depth $k = 0, 1, \ldots, N_c$ compute $M_t(k)$ the number of nodes which belong exclusively to the subtree tree T_t.

$$M_t(k) = \begin{cases} N_t(k) - N_{t+1}(k), & \text{if } t < d(i); \\ N_t(k), & \text{Otherwise.} \end{cases}$$

4. If $k > 0$ and $d(i) - t > 0$, set $O_i(d(i) - t, k) = O_i(d(i) - t, k) + M_t(k)$ else set $O_i(d(i) - t + 1, k + 1) = O_i(d(i) - t + 1, k + 1) + M_t(k)$.
5. Set $t = t + 1$. If $t \le d(i)$, then go to step (3), else terminate the algorithm.

4.4 Computing Complexity Matrix

To compute the complexity matrix R_i, each leaf node i has to be compared with internal nodes of the cost-weighted tree. After the "twin" node merging, one has to compare each node i of the merged tree with all the internal nodes of the tree. Note that the internal nodes of a binomial tree of order N_c form a binomial tree of order $(N_c - 1)$. Thus, similar to (27), the number of internal nodes belonging to subtree T_t at depth k is given by

$$L_t(k) = \begin{cases} \binom{T_t - 1}{k - t}, & \text{if } 0 \le k - t \le T_t - 1; \\ 0, & \text{otherwise.} \end{cases} \tag{28}$$

The algorithm to compute the complexity matrix R_i is slight variation of the one that calculates O_i.

1. Initialize $T = \{T_0, T_1, \ldots, T_d\} =$ the subtree membership of the node i, $d(i) =$ depth of the node i, set all the other elements of R_i equal to zero.
2. Set $t = 0$ such that the current subtree $T_t = T_0$.
3. For the subtree T_t, at each depth $k = 0, 1, \ldots, N_c$ compute $M_t(k)$ the number of nodes which belong exclusively to the subtree tree T_t.

$$M_t(k) = \begin{cases} L_t(k) - L_{t+1}(k), & \text{if } t < d(i); \\ L_t(k), & \text{Otherwise.} \end{cases}$$

4. If $m > 0$ and $d(i) - t > 0$, set $R_i(d(i) - t, k) = R_i(d(i) - t, m) + M_t(k)$ else set $R_i(d(i) - t + 1, k + 1) = R_i(d(i) - t + 1, k + 1) + M_t(k)$.
5. Set $t = t + 1$. If $t \le d(i)$, then go to step (3), else terminate the algorithm.

Note that when the internal node j is an ancestor of the leaf node i the probability $\Pr(S_n < S_m)$ is 1. We have to correct the complexity matrix for these cases by setting $R_i(k, 1) = R_i(k, 1) - 1$ where $0 \le k < d(i)$. With the corrected complexity matrix R_i, the complexity $N(i)$ becomes

$$N(i) = d(i) + \sum_{m=1}^{N_c} \sum_{n=1}^{N_c} \Pr(S_n < S_m) \cdot R_i(m, n). \tag{29}$$

Since the complexity for computing the optimality matrix and the complexity matrix increases exponentially with N_c, the computational complexity analysis for the model selection can only be done for moderate number of candidates. In the following section, computational complexity of branch-and-bound model selection is analyzed for the multibody structure-and-motion segmentation.

5 Experimental Results

The performance of the proposed model-selection framework was studied for the multibody structure-and-motion (MSaM) segmentation problem. The MSaM segmentation is carried out on a pair of images. The goal of MSaM segmentation is to group parts of image which have similar 3D motion. However, the number of motions in the image is unknown. Initially, patches in the pair of images are matched to find correspondence between them. As the objects in the image move, there appearance changes and the matching becomes challenging. Thus, some of the matches can be incorrect. For all the motions, the incorrectly matched patches are outliers. For a motion, the matches from other motions can be seen as outliers as well. Thus, the MSaM segmentation problem requires a technique which is immune to outliers and can detect the number of clusters automatically. The proposed method meets both requirements.

To generate candidates for the proposed model-selection approach, for each image correspondence, a fundamental matrix candidate was computed from circular spatial neighborhood of the correspondences using the "Structure-and-Motion Toolkit" from [37]. Outlier and inlier correspondences were selected for each fundamental matrix candidate by applying a threshold δ_T to the reprojection error of the correspondences for the candidate. If the reprojection error of a correspondence for a candidate is less than δ_T, the candidate explains the correspondence, and the correspondence can be treated as an inlier for the candidate. On the other hand, if the reprojection error of a correspondence for a candidate is greater than δ_T, the candidate fails to explain the correspondence. In this scenario, the correspondence has to be treated as an outlier. The number of inlier correspondences for each candidate indicates the support for the candidate. To avoid repeated similar candidates, candidates, with smaller support sharing substantial (>80 percent) inlier correspondences with a candidate with larger support, were suppressed. Finally, the surviving candidates were arranged in decreasing order of their support. An alternative candidate for outliers C_0 was added with $r_j(C_0) = \delta_T$. Note that for all the experiments δ_T was chose to be 3. The Bayesian information criterion (BIC) was optimized for these candidates to select the optimal hypothesis.

The first two experiments were targeted toward quality of segmentation while the last experiment was carried out to verify the proposed computational complexity framework.

5.1 Clustering Quality

The clustering quality of the branch-and-bound model-selection approach was estimated with synthetic data and was compared with iterative MSaM segmentation. For the experiments, 100 different fundamental matrices were randomly generated. For each fundamental matrix, 50 correspondences were generated with the model given by (16) adding iid Gaussian noise with $\sigma = 1$. At a time, correspondences for different fundamental matrices were combined together to form an experimental data set.

First, iterative clustering was carried out with the experimental data set with 2, 3 and 4 clusters. For iterative clustering, the number of clusters was assumed to be known. Initially, each correspondence was assigned to one of the clusters randomly. From the initial assignment, fundamental matrix for each cluster was computed with the "Structure-and-Motion Toolkit" from [37]. This step is equivalent to parameter estimation step in (4). In the next step each correspondence was assigned to the cluster for which the reprojection error is minimized. This corresponds

TABLE 1
Clustering Accuracy in Percent

Clusters	Iterative No outliers	Iterative	Proposed
1	-	97.60	98.10
2	95.77	74.90	96.96
3	85.35	54.54	96.09
4	80.17	44.83	95.60

to membership estimation step in (5). These two steps were repeated till the cluster memberships converged. Note that to compute the fundamental matrix, a robust MLESAC estimator [38] was used.

For rest of the experiments, 50 randomly selected correspondences from the remaining motions were added to the data as outliers. Iterative segmentation was repeated with the updated data. The correspondences for which the reprojection errors were larger that a threshold δ_T were labeled as outliers after each iteration.

Finally, the data was segmented with the proposed approach. Average of clustering accuracy for 100 runs is shown is Table 1. For the iterative approach, the clustering accuracy starts at 95 percent with two clusters and drops as number of clusters increases. As the number of clusters and data points increases, the search space for the clustering solution expands. As the iterative clustering yields a local optimum, the quality of the clustering deteriorates with expansion of the search space. However, the drop in accuracy for the iterative approach is much steeper when outliers are added to the data. Note that for the data with outliers, the accuracy reported corresponds to correctly clustering inliers in to individual cluster and correctly labeling all the outliers. Thus with outliers, the accuracy is meaningful for one cluster unlike the iterative clustering without outliers. The proposed approach outperforms iterative clustering method even in presence of outliers. The drop in the clustering accuracy is minimal (98 percent to 95 percent) as the number of clusters is increased. Unlike the iterative clustering, the number of clusters is assumed to be unknown and automatically detected by the proposed method.

5.2 Real Data

Publicly available image data sets were chosen for the clustering quality experiments with real data. The chosen data sets provide images as well as correspondences.

For the first experiment, “Box-book-mag” and “Desk” image pairs from [9] are used. The “Box-book-mag” pair has three independently moving objects captured with a stationary camera. For these image pairs, in addition to correct correspondences, some incorrect matches are provided. In Fig. 5a, the detected motions between the image pair are indicated by different colored lines. Each line connects a point in an image to the position of its matched correspondence in the second image. This provides “track” of the points as the visual indication of the motion overlaid on one of the images. A group of lines with same color indicates that they follow the same motion. The red colored lines are the detected outlier motions. Figs. 5b and 5c show segmentation results where correspondences marked with same color are determined to have same motion and belong to same cluster. The outlier matches corresponding to the red lines are removed in Figs. 5b and 5c.

For the “Desk” image pair shown in Fig. 6, there are three moving objects, namely a pile of books, a computer screen and a journal. Although the camera has also moved, there are no matches for the background. Thus the background motion is not detected. The result of segmentation can be seen in Figs. 6b and 6c.

In the next experiment, our method is applied to the “car-truck-box” sequence used by Vidal et al. [39], [40]. The motion between frame 1 and frame 10 of the sequence was analyzed. In this sequence, there are three different motions. The box lies on a rotating desk, while the car and the truck are moved away from each other with hand. As seen in Fig. 7, three moving objects are correctly identified; however, some of the motion vectors are incorrectly assigned. This is due to the sampling scheme rather than the cost function being optimized.

Finally, the proposed approach was tested with JHU155 database sequences [41], which include traffic sequences with two or three motions. The “cars2” sequence shown in Fig. 8 has two moving cars captured by a moving camera. Fig. 9b gives segmentation results for the “cars3” sequence,

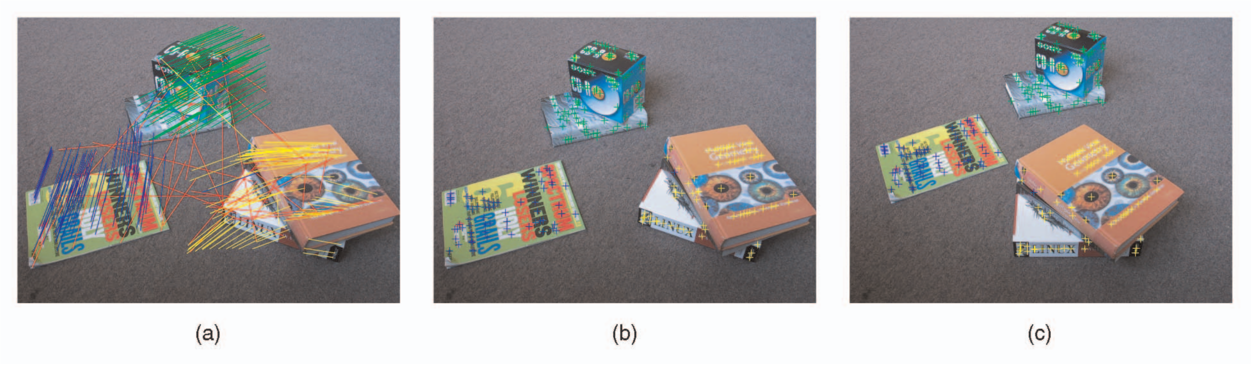

Fig. 5. Box-book-mag: (a) Motion between two views, each cluster is denoted by different color, motions marked by red are outliers, (b) segmentation result for the first view, and (c) segmentation result for the second view.

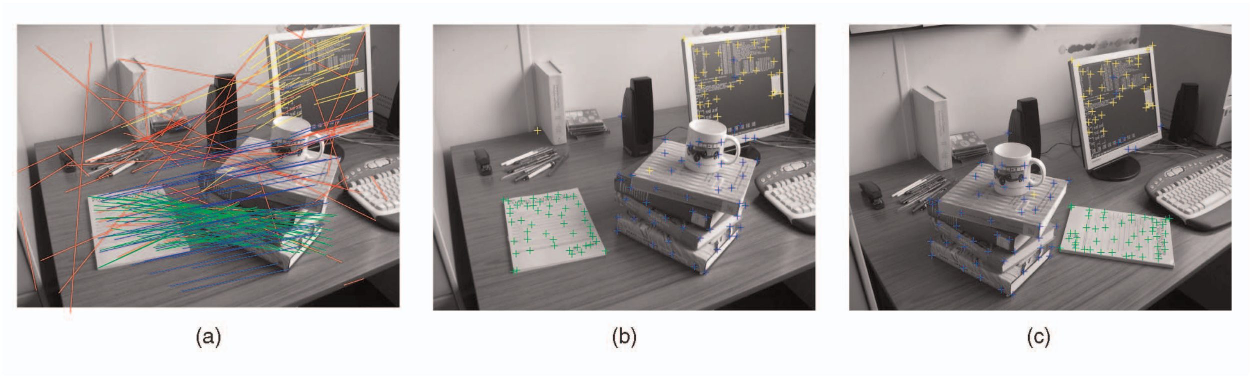

Fig. 6. Desk: (a) Motion between two views, each cluster is denoted by different color, matches marked by red are outliers, (b) segmentation result for the first view, and (c) segmentation result for the second view.

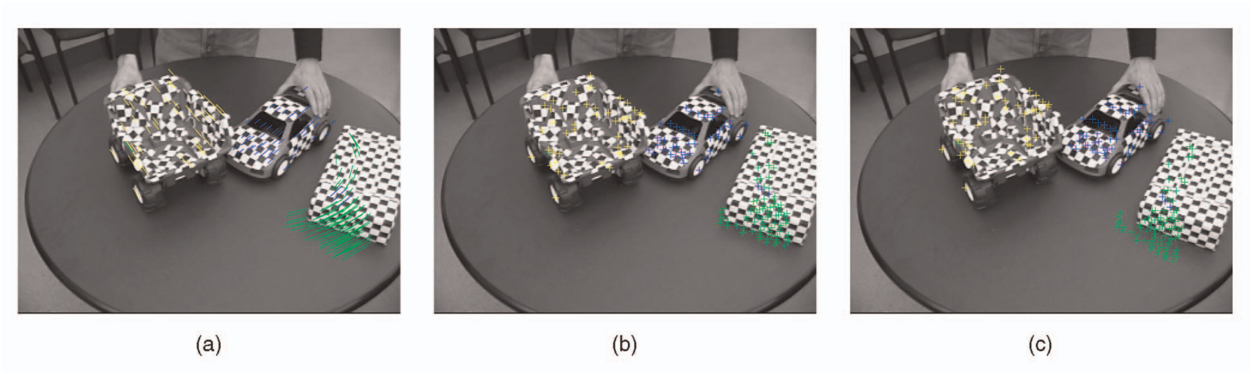

Fig. 7. Car-truck-box: (a) Motion between two views, each cluster is denoted by different color, matches marked by red are outliers, (b) segmentation result for the first view, and (c) segmentation result for the second view.

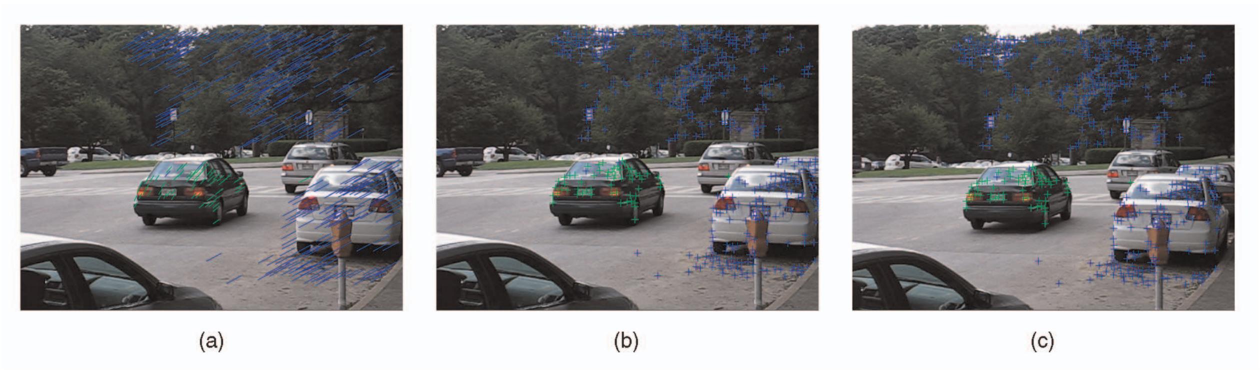

Fig. 8. JHU-Cars2: (a) Motion between two views, each cluster is denoted by different color, (b) segmentation result for the first view, and (c) segmentation result for the second view.

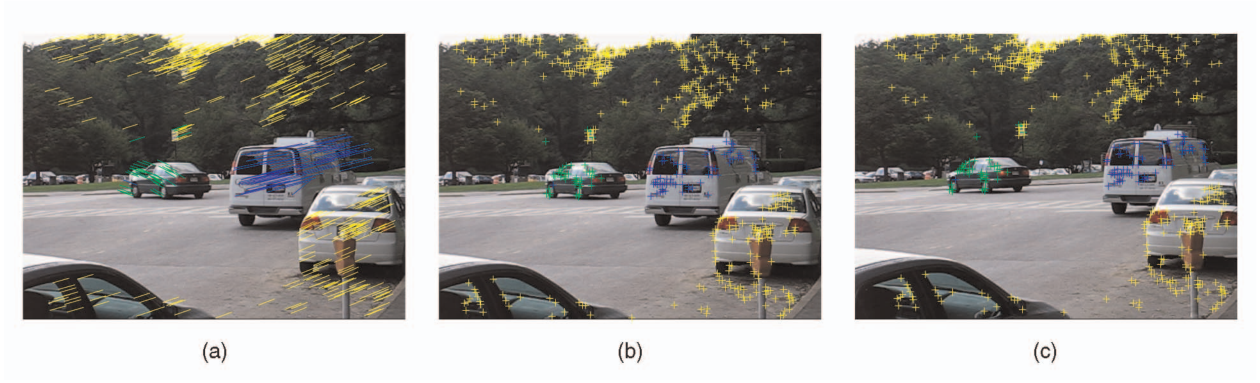

Fig. 9. JHU-Cars3: (a) Motion between two views, each cluster is denoted by different color, (b) segmentation result for the first view, and (c) segmentation result for the second view.

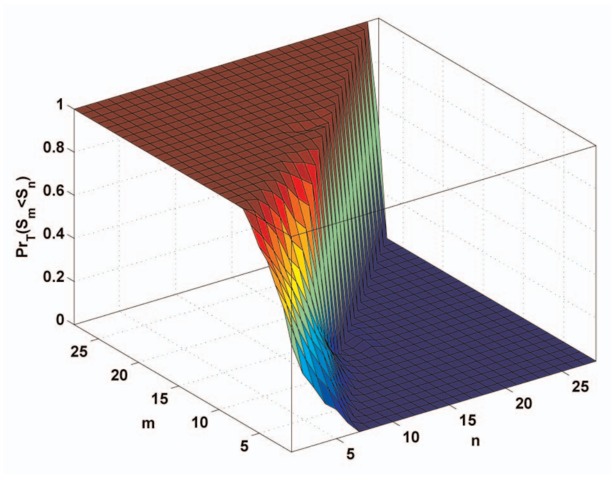

Fig. 10. $\mathrm{Pr}_T(S_m < S_n)$ for the MSaM segmentation problem.

which depicts two cars captured with moving camera. In all the experiments, the number of motions is correctly identified and the segmentation achieved is accurate.

5.3 Computational Complexity

The complexity of the proposed branch-and-bound model-selection approach was estimated with synthetic data. The synthetic data were generated similar to the first experiment and has 4 clusters, each of size 50, and 50 outliers. For the approach, the number of candidates N_c cannot be explicitly controlled, and it varies with the number of motions and their spatial configuration. For the generated synthetic data N_c was close to 20 to 30. To estimate the probabilities $\mathrm{Pr}_T(S_m < S_n)$ for the MSaM segmentation problem, we randomly generated pairs of hypotheses and compared their BIC values. To calculate the probabilities $\mathrm{Pr}_I(S_m < S_n)$, BIC value of a randomly generated hypothesis was compared to the lower bound on BIC value of another randomly generated hypothesis. Figs. 10 and 11 show the probability matrices $\mathrm{Pr}_T(S_m < S_n)$ and $\mathrm{Pr}_I(S_n < S_m)$ respectively.

Once the probability matrices $\mathrm{Pr}_T(S_m < S_n)$ and $\mathrm{Pr}_I(S_n < S_m)$ are known from sampling, the complexity for the branch-and-bound search can be estimated by evaluating (24). Fig. 12 shows the estimated expected complexity for the MSaM segmentation problem along with other tree search problems when edge weights are uniform iids and squared Gaussian iids. The worst case complexity, which is equivalent to a brute force search, is also shown for comparison. Clearly, the expected complexity of the branch-and-bound search depends on the distribution of the edge weights. This distribution is captured by probabilities $\mathrm{Pr}_T(S_m < S_n)$ and $\mathrm{Pr}_I(S_n < S_m)$.

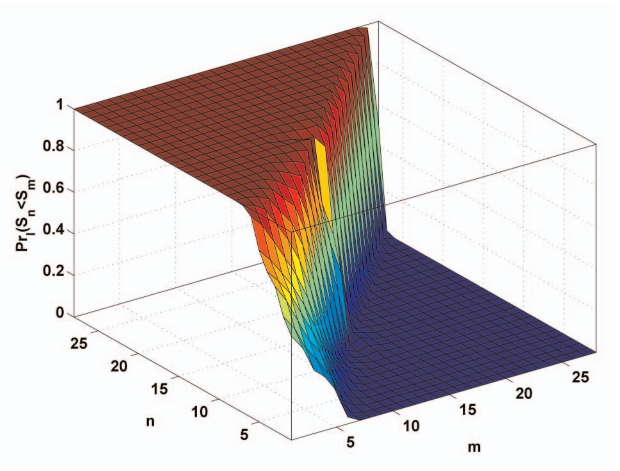

Fig. 11. $\mathrm{Pr}_I(S_n < S_m)$ for the MSaM segmentation problem.

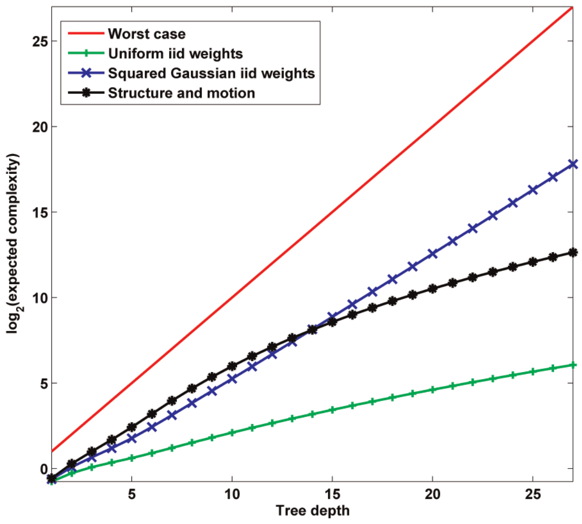

Fig. 12. Expected complexity.

As seen from the plots, although the expected complexity is much lesser than the worst case complexity for the branch-and-bound, it remains exponential for the most part. The rate of exponential depends on how quickly the off diagonal values of probability matrices $\mathrm{Pr}_T(S_m < S_n)$ and $\mathrm{Pr}_I(S_n < S_m)$ drop to near zero/rise close to one. On the other hand, for the MSaM segmentation problem, the increase in the complexity as $N_c > 15$ is not as drastic as $N_c < 15$. This again is a result of the off diagonal values of probability matrices $\mathrm{Pr}_T(S_m < S_n)$ and $\mathrm{Pr}_I(S_n < S_m)$ almost all of which drop to near zero/rise close to one for $N_c > 15$.

Fig. 13 compares the estimated expected complexity of the problem with the experimentally observed complexity of the problem for various values of N_c. We ran 400 experiments with different data sets to find the number of nodes explored before the optimal solution was found. These experiments were then separated based on the value of N_c, which was anywhere between 20 and 30, as mentioned before. Then the experiments corresponding to each value of N_c were sorted according to increasing complexity. Note that the figure shows only $N_c = 24$ to 27, which were the four most frequent values of N_c. As the number of experiments for each value of N_c were different, the length of the plots were normalized to 100 before plotting for easy comparison of complexity for various values of N_c. As seen in Fig. 13, although the expected complexity is slightly overestimated, it still provides a satisfactory estimate for the observed complexity.

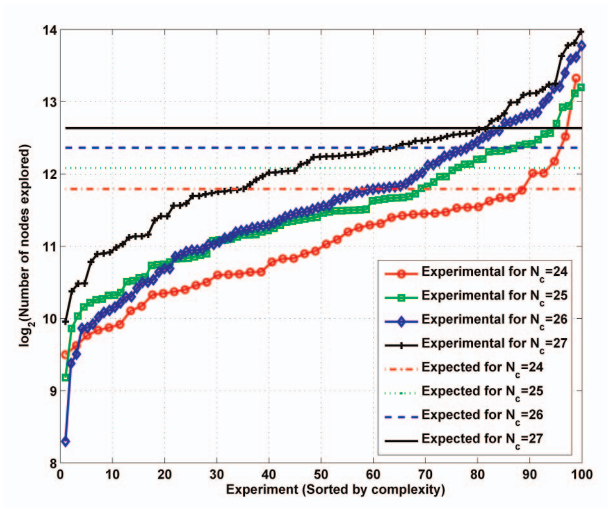

Fig. 13. Comparison of expected and actual complexity.

6 Conclusion

In this paper, we proposed a branch-and-bound model-selection algorithm for model-based clustering of image data and analyzed its expected complexity. The proposed model-selection-based approach detects the number of clusters automatically, and it is robust to outliers. When compared to conventional iterative algorithms for model-based clustering, the proposed algorithm shows marked improvement in the clustering quality. It can also be seen from the experiments that the average complexity of the algorithm is much lower than the worst case complexity. Thus the proposed algorithm is practical for model-based clustering of image data, which has moderate number of candidates, and when the size of optimal candidate subset is small. With problem-specific bounds and/or added heuristics, the complexity of the branch-and-bound algorithm can be further reduced.

Appendix

Cost Probabilities for Uniform IID

The sum of m uniform iids is distributed as

$$f_m(S_m) = \frac{1}{(m-1)!}\sum_{j=0}^{m}(-1)^j\binom{m}{j}[(S_m-j)_+]^{m-1}. \tag{30}$$

Similarly, for the sum of n uniform iids

$$f_n(S_n) = \frac{1}{(n-1)!}\sum_{k=0}^{n}(-1)^k\binom{n}{k}[(S_n-k)_+]^{n-1}. \tag{31}$$

The corresponding cumulative distributions are

$$F_m(S_m) = \frac{1}{m!}\sum_{j=0}^{m}(-1)^j\binom{m}{j}[(S_m-j)_+]^{m} \tag{32}$$

and

$$F_n(S_n) = \frac{1}{n!}\sum_{k=0}^{n}(-1)^k\binom{n}{k}[(S_n-k)_+]^{n}. \tag{33}$$

For the summation on right-hand side of (32), in interval $j-1$ and j only j terms are nonzero, and for (32), in interval $k-1$ and k only k terms are nonzero.

Assuming S_m and S_n are independent, the joint distribution of S_m and S_n is same as their product. From the joint distribution of S_m and S_n

$$\Pr(S_m < S_n) = \int_{S_n=0}^{n}\int_{S_m=0}^{S_n} f_m(S_m)f_n(S_n)dS_m dS_n. \tag{34}$$

For $n < m$,

$$\Pr(S_m < S_n) = \frac{1}{(n-1)!}\int_{S_n=0}^{n}\sum_{k=0}^{n}(-1)^k\binom{n}{k}[(S_n-k)_+]^{n-1} \left(\frac{1}{(m-1)!}\int_{S_m=0}^{S_n}\sum_{j=0}^{m}(-1)^j\binom{m}{j}[(S_m-j)_+]^{m-1}dS_m\right)dS_n.$$

The bracketed expression is nothing but cumulative distribution of S_m

$$\Pr(S_m < S_n) = \frac{1}{(n-1)!}\int_{S_n=0}^{n}\sum_{k=0}^{n}(-1)^k\binom{n}{k}[(S_n-k)_+]^{n-1} \left(\frac{1}{m!}\sum_{j=0}^{m}(-1)^j\binom{m}{j}[(S_n-j)_+]^{m}\right)dS_n. \tag{35}$$

Since $S_n \le n$, for $j > n$ the bracketed expression is 0, we change the upper limit of the summation over j to n

$$\Pr(S_m < S_n) = \frac{1}{(n-1)!}\frac{1}{m!} \underbrace{\int_{S_n=0}^{n}\left(\sum_{k=0}^{n}(-1)^k\binom{n}{k}[(S_n-k)_+]^{n-1}\right)\left(\sum_{j=0}^{n}(-1)^j\binom{m}{j}[(S_n-j)_+]^{m}\right)dS_n}_{I}. \tag{36}$$

We split integration I in to n segments each of length 1. For $q = \{1, 2, \ldots, n\}$

$$\begin{aligned} I_q &= \int_{S_n=q-1}^{q}\left(\sum_{k=0}^{q-1}(-1)^k\binom{n}{k}(S_n-k)^{n-1}\right)\left(\sum_{j=0}^{q-1}(-1)^j\binom{m}{j}(S_n-j)^{m}\right)dS_n \\ &= \sum_{k=0}^{q-1}\sum_{j=0}^{q-1}(-1)^{(k+j)}\binom{n}{k}\binom{m}{j}\underbrace{\int_{S_n=q-1}^{q}(S_n-k)^{n-1}(S_n-j)^m dS_n}_{I_q(j,k)}. \end{aligned} \tag{37}$$

Now we solve for integration $I_q(j,k)$

$$I_q(j,k) = \int_{S_n=q-1}^{q}(S_n-k)^{n-1}(S_n-j)^m dS_n. \tag{38}$$

Let $t = (S_n - j)$, then $dt = dS_n$ and $(S_n - k) = (t + j - k)$. When $S_n = q-1$, $t = q-1+j$, and when $S_n = q$, $t = q+j$

$$I_q(j,k) = \int_{t=q-1+j}^{q+j}(t+j-k)^{n-1}t^m dt. \tag{39}$$

TABLE 2
Repeated Differential Table for $(t+j-k)^{n-1}$

p	Repeated differential
1	$(t+j-k)^{n-1}$
2	$(n-1)(t+j-k)^{n-2}$
3	$(n-1)\cdot(n-2)(t+j-k)^{n-3}$
p	$(n-1)\cdot(n-2)\dots(n-(p-1))(t+j-k)^{(n-p)}$
$p+1$	$(n-1)\cdot(n-2)\dots(n-(p-1))(n-p)(t+j-k)^{(n-(p+1))}$
$(n-1)$	$(n-1)\cdot(n-2)\dots 3\cdot 2(t+j-k)$
n	$(n-1)\cdot(n-2)\dots 3\cdot 2\cdot 1$
$n+1$	0

We apply repeated integration by parts to (39) with the help of Tables 2 and 3. After resubstituting $S_n - j$ for t, we get

$$\begin{aligned} &I_q(j,k) \\ &= \Bigg[\sum_{p=1}^{n}(-1)^{(p-1)}(n-1)\dots(n-(p-1))(S_n-k)^{(n-p)} \\ &\quad \frac{(S_n-j)^{(m+p)}}{(m+1)\dots(m+p-1)(m+p)}\Bigg]^{q}_{(q-1)} \qquad (40) \\ &= \Bigg[\sum_{p=1}^{n}(-1)^{(p-1)}\frac{(n-1)!}{(n-p)!}(S_n-k)^{(n-p)} \\ &\quad \frac{m!}{(m+p)!}(S_n-j)^{(m+p)}\Bigg]^{q}_{(q-1)}. \end{aligned}$$

Combining (36), (37), (39), and (40)

$$\begin{aligned} &\Pr(S_m < S_n) \\ &= \frac{1}{(n-1)!}\frac{1}{m!}\sum_{q=1}^{n}\sum_{k=0}^{q-1}\sum_{j=0}^{q-1}(-1)^{(k+j)}\binom{n}{k}\binom{m}{j} \\ &\quad \Bigg[\sum_{p=1}^{n}(-1)^{(p-1)}\frac{(n-1)!}{(n-p)!}(S_n-k)^{(n-p)} \\ &\quad \frac{m!}{(m+p)!}(S_n-j)^{(m+p)}\Bigg]^{q}_{(q-1)} \qquad (41) \\ &= \sum_{q=1}^{n}\sum_{k=0}^{q-1}\sum_{j=0}^{q-1}(-1)^{(k+j)}\binom{n}{k}\binom{m}{j} \\ &\quad \left[\sum_{p=1}^{n}(-1)^{(p-1)}\frac{(S_n-k)^{(n-p)}}{(n-p)!}\frac{(S_n-j)^{(m+p)}}{(m+p)!}\right]^{q}_{(q-1)}. \end{aligned}$$

TABLE 3
Repeated Integration Table for t^m

p	Repeated integration
1	t^m
2	$\frac{t^{(m+1)}}{(m+1)}$
3	$\frac{t^{(m+2)}}{(m+1)(m+2)}$
p	$\frac{t^{(m+p-1)}}{(m+1)(m+2)\dots(m+p-1)}$
$p+1$	$\frac{t^{(m+p)}}{(m+1)(m+2)\dots(m+p-1)(m+p)}$
$(n-1)$	$\frac{t^{(m+n-2)}}{(m+1)(m+2)\dots(m+n-2)}$
n	$\frac{t^{(m+n-1)}}{(m+1)(m+2)\dots(m+n-2)(m+n-1)}$
$n+1$	$\frac{t^{(m+n)}}{(m+1)(m+2)\dots(m+n-1)(m+n)}$

References

[1] J. Han and M. Kamber, *Data Mining: Concepts and Techniques.* Morgan Kaufmann Publishers Inc., 2006.
[2] D.A. Forsyth and J. Ponce, *Computer Vision: A Modern Approach.* Prentice Hall, 2002.
[3] S. Theodoridis and K. Koutroumbas, *Pattern Recognition.* Academic Press, 2008.
[4] S. Mitra and T. Acharya, *Data Mining: Multimedia, Soft Computing, and Bioinformatics.* Wiley-Interscience, 2003.
[5] S. Zhong and J. Ghosh, "A Unified Framework for Model-Based Clustering," *J. Machine Learning Research,* vol. 4, pp. 1001-1037, 2003.
[6] A.P. Dempster, N.M. Laird, and D.B. Rubin, "Maximum Likelihood from Incomplete Data Via the Em Algorithm," *J. Royal Statistical Soc. B,* vol. 39, no. 1, pp. 1-38, http://dx.doi.org/10.2307/2984875, 1977.
[7] K.P. Burnham and D.A. Anderson, *Model Selection and Multi-Model Inference.* Springer, 2003.
[8] H. Li, "Two-view Motion Segmentation from Linear Programming Relaxation," *Proc. IEEE Int'l Conf. Computer Vision and Pattern Recognition,* pp. 1-8, June 2007.
[9] K. Schindler and D. Suter, "Two-view Multibody Structure-and-Motion with Outliers through Model Selection," *IEEE Trans. Pattern Analysis and Machine Intelligence,* vol. 28, no. 6, pp. 983-995, June 2006.
[10] N. Thakoor and J. Gao, "Branch-and-Bound Hypothesis Selection for Two-View Multiple Structure and Motion Segmentation," *Proc. IEEE Int'l Conf. Computer Vision and Pattern Recognition,* pp. 1-6, June 2008.
[11] R.I. Hartley and A. Zisserman, *Multiple View Geometry in Computer Vision,* second ed. Cambridge Univ. Press, 2004.
[12] P.H.S. Torr, "Geometric Motion Segmentation and Model Selection," *Philosophical Trans. Royal Soc. London A,* vol. 356, no. 1740, pp. 1321-1340, http://dx.doi.org/10.1098/rsta.1998.0224, May 1998.
[13] F. Glover and M. Laguna, *Modern Heuristic Techniques for Combinatorial Problems.* John Wiley & Sons, Inc., pp. 70-150, 1993.
[14] F.A. Chudak and D.B. Shmoys, "Improved Approximation Algorithms for the Uncapacitated Facility Location Problem," *SIAM J. Computation,* vol. 33, no. 1, pp. 1-25, 2004.
[15] D.S. Hochbaum, Ed., *Approximation Algorithms for NP-hard Problems.* PWS Publishing Co., 1997.
[16] G. Schwarz, "Estimating the Dimension of a Model," *Annals of Statistics,* vol. 6, no. 2, pp. 461-464, http://www.jstor.org/stable/2958889, 1978.
[17] M. Brusco and S. Stahl, *Branch-and-Bound Applications in Combinatorial Data Analysis.* Springer, 2005.
[18] K. Fukunaga, *Introduction to Statistical Pattern Recognition.* Academic Press Professional, Inc., 1990.
[19] P. Somol, P. Pudil, and J. Kittler, "Fast Branch and Bound Algorithms for Optimal Feature Selection," *IEEE Trans. Pattern Analysis and Machine Intelligence,* vol. 26, no. 7, pp. 900-912, July 2004.
[20] L.-K. Shark, A.A. Kurekin, and B.J. Matuszewski, "Development and Evaluation of Fast Branch-and-Bound Algorithm for Feature Matching Based on Line Segments," *Pattern Recognition,* vol. 40, no. 5, pp. 1432-1450, 2007.
[21] M. Roder, J. Cardinal, and R. Hamzaoui, "Branch and Bound Algorithms for Rate-Distortion Optimized Media Streaming," *IEEE Trans. Multimedia,* vol. 8, no. 1, pp. 170-178, Feb. 2006.
[22] J. Jonsson and K. Shin, "A Parametrized Branch-and-Bound Strategy for Scheduling Precedence-Constrained Tasks on a Multiprocessor System," *Proc. Int'l Conf. Parallel Processing,* pp. 158-165, 1997.

[23] S. Fujita, M. Masukawa, and S. Tagashira, "A Fast Branch-and-Bound Algorithm with an Improved Lower Bound for Solving the Multiprocessor Scheduling Problem," *Proc. Int'l Conf. Parallel and Distributed Systems*, pp. 611-616, 2002.

[24] W.L.G. Koontz, P.M. Narendra, and K. Fukunaga, "A Branch and Bound Clustering Algorithm," *IEEE Trans. Computation,* vol. 24, no. 9, pp. 908-915, 1975.

[25] N. Thakoor, V. Devarajan, and J. Gao, "Multi-Stage Branch-and-Bound for Maximum Variance Disparity Clustering," *Proc. Int'l Conf. Pattern Recognition*, pp. 1-4, Dec. 2008.

[26] N. Thakoor, J. Gao, and V. Devarajan, "Multistage Branch-and-Bound Merging for Planar Surface Segmentation in Disparity Space," *IEEE Trans. Image Processing,* vol. 17, no. 11, pp. 2217-2226, Nov. 2008.

[27] L. Devroye and C. Zamora-Cura, "On the Complexity of Branch-and Bound Search for Random Trees," *Random Structures and Algorithms,* vol. 14, no. 4, pp. 309-327, July 1999.

[28] D.R. Smith, "On the Computational Complexity of Branch and Bound Search Strategies," PhD dissertation, Duke Univ., 1979.

[29] D.R. Smith, "Random Trees and the Analysis of Branch and Bound Procedures," *J. ACM,* vol. 31, no. 1, pp. 163-188, 1984.

[30] H.S. Stone and P. Sipala, "The Average Complexity of Depth-First Search with Backtracking and Cutoff," *IBM J. Research and Development,* vol. 30, no. 3, pp. 242-258, 1986.

[31] W. Zhang, "Branch-and-Bound Search Algorithms and Their Computational Complexity," Univ. of southern California/Information Sciences Inst., Tech. Rep. ISI/RR-96-443, May 1996.

[32] W. Zhang and R.E. Korf, "Performance of Linear-Space Search Algorithms," *Artificial Intelligence,* vol. 79, pp. 241-292, 1995.

[33] J.L. Crassidis and J.L. Junkins, *Optimal Estimation of Dynamic Systems.* Chapman & Hall/CRC, 2004.

[34] A.D.R. McQuarrie and C.-L. Tsai, *Regression and Time Series Model Selection.* World Scientific, 1998.

[35] T.H. Cormen, C.E. Leiserson, R.L. Rivest, and C. Stein, *Introduction to Algorithms.* The MIT Press, 2001.

[36] W. Feller, *An Introduction to Probability Theory and Its Applications,* second ed. vol. 2, John Wiley and Sons, 1966.

[37] P. Torr, *A Structure and Motion Toolkit in Matlab,* http://cms.brookes.ac.uk/staff/PhilipTorr/Beta/torrsam.zip, 2010.

[38] P.H.S. Torr and A. Zisserman, "MLESAC: A New Robust Estimator with Application to Estimating Image Geometry," *Computer Vision and Image Understanding,* vol. 78, no. 1, pp. 138-156, 2000.

[39] R. Vidal and S. Sastry, "Optimal Segmentation of Dynamic Scenes from Two Perspective Views," *Proc. IEEE Int'l Conf. Computer Vision and Pattern Recognition,* vol. 2, pp. 281-286, June 2003.

[40] R. Vidal, S. Soatto, Y. Ma, and S. Sastry, "Segmentation of Dynamic Scenes from the Multibody Fundamental Matrix," *Proc. European Conf. Computer Vision Workshop Visual Modeling of Dynamic Scenes,* 2002.

[41] *The Hopkins 155 Dataset,* http://www.vision.jhu.edu/data/hopkins155/, 2010.

Ninad Thakoor (S'04-M'10) received the BE degree in electronics and telecommunication engineering from the University of Mumbai, India, in 2001, and the MS and PhD degrees in electrical engineering from the University of Texas at Arlington, in 2004 and 2009, respectively. His research interests include visual object recognition, stereo disparity segmentation, and structure-and-motion segmentation. He is a member of the IEEE.

Jean Gao (S'96-M'03) received the BS degree in biomedical engineering from Shanghai Medical University, Shanghai, China, in 1990, the MS degree in biomedical engineering from the Rose-Hulman Institute of Technology, Terre Haute, Indiana, in 1996, and the PhD degree in electrical engineering from Purdue University, West Lafayette, Indiana, in 2002. She is currently an associate professor with the Computer Science and Engineering Department, and the director of Biocomputing and Vision Lab, University of Texas at Arlington. Her research interests include object tracking, motion estimation, stereo object segmentation, pattern recognition, medical imaging, applications in computational biology, and clinical medical informatics. She is the recipient of the prestigious CAREER Award from the US National Science Foundation (NSF) and the Outstanding Young Faculty Award from the University of Texas at Arlington. She is a senior member of the IEEE.