Selecting Canonical Views for View-Based 3-D Object Recognition

Trip Denton
Drexel University

M. Fatih Demirci
Drexel University

Jeff Abrahamson
Drexel University

Ali Shokoufandeh
Drexel University

Sven Dickinson
University of Toronto

Abstract

Given a collection of sets of 2-D views of 3-D objects and a similarity measure between them, we present a method for summarizing the sets using a small subset called a bounded canonical set (BCS), whose members best represent the members of the original set. This means that members of the BCS are as dissimilar from each other as possible, while at the same time being as similar as possible to the non-BCS members. This paper will extend our earlier work on computing canonical sets [2] in several ways: by omitting the need for a multi-objective optimization, by allowing the imposition of cardinality constraints, and by introducing a total similarity function. We evaluate the applicability of BCS to view selection in a view-based object recognition environment.

1. Introduction

In this paper we will present a computational framework for a variation of the pattern summarization problem known as the *canonical set* problem. Intuitively, for a given set of patterns under a known similarity function, its canonical set is the small subset of its members that best characterizes the elements of this set. Formally, if $\mathcal{P} = \{p_1, ..., p_n\}$ is the set of patterns and $\mathcal{S} : \mathcal{P} \times \mathcal{P} \rightarrow \mathbb{R}^{\geq 0}$ denotes the similarity function, we are interested in a *small* subset $\mathcal{P}' \subseteq \mathcal{P}$ that maximizes the similarity between $\mathcal{P}'$ and $\mathcal{P} \setminus \mathcal{P}'$. This problem is known to be computationally intractable [3].

Our approach to canonical sets was motivated by the novel view expressed by Cyr and Kimia [1] that "the shape similarity metric between object outlines endows the viewing sphere with a metric which can be used to cluster views into aspects, and to represent each aspect with a prototypical view." Additionally, our work on canonical sets of a general class of patterns is in large part motivated by novel ideas that were introduced in the context of aspect graph representations [8] and their relevance in identifying regions of "equivalent views" on the viewing sphere.

Recently [2], we proposed an approximation algorithm for solving a variation of canonical set for single-pattern classes using a multi-objective optimization. In this paper, we extend our framework with the notion of a *bounded canonical set* (BCS), which imposes upper and lower bounds on the cardinality of the canonical set. Such bounds are important for applications that required a representative set of prescribed size. The possibility of imposing such bounds will preclude the need for a multi-objective optimization, which is an expensive and complex step in the original formulation of canonical set. The elements of BCS have the additional property of being maximally dissimilar from each other. This property is useful when dealing with patterns belonging to multiple objects and one desires canonical elements that best represent one object while being maximally dissimilar from representative elements of other objects. Finally, in the current formulation, we will not make any restrictive assumptions about the similarity function, i.e., we will assume that all elements are explicitly comparable.

As motivation, consider a set of patterns consisting of 19 views of a single object (chair) in Figure 1 taken along the equatorial great circle. Moreover, assume we are also given a similarity measure to compare 2-D views (see Section 3). Our goal is to identify a small canonical set (containing between 2 and 3 views) that best characterizes this set. The solution to this BCS problem computed by our algorithm is outlined in blue in Figure 1.

The BCS problem can be formally stated as follows: Given a set of views of an object $\mathcal{P} = \{p_1, ..., p_n\}$, a similarity function $\mathcal{S} : \mathcal{P} \times \mathcal{P} \rightarrow \mathbb{R}^{\geq 0}$, and two integer bounds k_{min} and k_{max}, the *bounded canonical set* for $\mathcal{P}$ is a subset $\mathcal{P}' \subseteq \mathcal{P}$ that best characterizes the elements of $\mathcal{P}$ with respect to the similarity function $\mathcal{S}$ while having cardinality k, where $k_{min} \leq k \leq k_{max}$.

Figure 1. Views of chair with its BCS outlined.

2. BCS Construction

In this section, we describe our method for constructing bounded canonical sets in polynomial time. Starting with a set of views $\mathcal{P} = \{p_1, ..., p_n\}$ of an object, and a similarity function $\mathcal{S} : \mathcal{P} \times \mathcal{P} \to \mathbb{R}^{\geq 0}$, we construct a complete edge weighted graph $G = G(\mathcal{P})$, where the patterns $\{p_1, ..., p_n\}$ are represented by vertices, and the edges between the vertices have weights corresponding to the measure of similarity between the vertices. We use V^* to denote the vertices in V, the vertex set of G, corresponding to the BCS.

Examining the canonical set shown in Figure 2, we categorize the edge set of G into three groups: *intra* edges, where both endpoints are within the canonical set V^*, *cut* edges, where one of the endpoints is in the canonical set and the other is not, and *extra* edges, which are the rest. Our goal is to minimize the sum of the weights of the intra edges, while at the same time maximizing the sum of the weights of the cut edges. In doing so, we attempt to place the vertices that are most representative of the others in the BCS, while keeping the BCS members as dissimilar as possible.
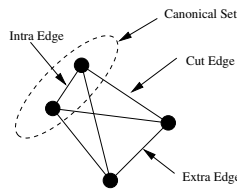

Figure 2. BCS edges

The problem of maximizing the weight of the cut edges is known to be NP hard. Goemans and Williamson [5] explored the problem MAX-CUT in graphs, and used semidefinite programming (SDP) relaxations to provide good approximate solutions. See Goemans [4] and Mahajan and Ramesh [9] for a survey of recent results and applications of SDP. Following their lead, we formulate our problem of BCS as an integer programming problem, and then use SDP to give us a good approximate solution. Recently, Shi and Malik [10] proposed an optimization formulation for segmentation problems in terms of special cuts, known as nor-

malized cuts. They provided an approximation solution to the segmentation problem in terms of eigen-values of Laplacian matrices associated with similarity graphs.

For each pattern p_i, $1 \leq i \leq n$, we introduce a binary indicator variable y_i. The variable y_i can have a value of $+1$ or -1, indicating whether the corresponding pattern belongs to V^* or $V \setminus V^*$, respectively. Let $y \in \{-1, +1\}^n$ be the vector $[y_1, \ldots, y_n]^t$. The problem of maximizing the sum of the weights of the cut edges can then be formulated as

$$\text{Maximize } \frac{1}{2} \sum_{i,j} \mathcal{W}_{ij}(1 - y_i y_j). \tag{1}$$

Similarly, minimizing the sum of the weights of the intra edges becomes

$$\text{Minimize } \frac{1}{4} \sum_{i,j} \mathcal{W}_{ij}(1 + y_i)(1 + y_j). \tag{2}$$

Due to the intractability of these two integer programming formulations [3], and in order to prepare for our approximation, we will reformulate the problem as a quadratic optimization problem. To this end, we introduce a *set indicator* variable $y_{n+1} \in \{-1, +1\}$, that is, $p_i \in V^*$, $1 \leq i \leq n$, if and only if $y_i = y_{n+1}$. It is easy to see that equations 1 and 2 can be combined into the minimization objective:

$$\frac{3}{4} \sum_{i,j} \mathcal{W}_{ij} y_i y_j + \frac{1}{2} \sum_{i=1}^{n} y_{n+1} y_i \sum_{j=1}^{n} \mathcal{W}_{ij} - \frac{1}{4} \sum_{ij} \mathcal{W}_{ij} \tag{3}$$

subject to the following constraints on the size of the canonical set:

$$\frac{1}{2} \sum_{i=1}^{n} (1 + y_{n+1} y_i) - k_{min} \geq 0 \tag{4}$$

$$k_{max} - \frac{1}{2} \sum_{i=1}^{n} (1 + y_{n+1} y_i) \geq 0 \tag{5}$$

Let the vector d be a vector in $\mathbb{R}^n$ whose i^{th} entry has value $d_i = \frac{1}{2} \sum_{j=1}^{n} \mathcal{W}_{i,j}$. Define the scalar $w_{all} = \sum_{i,j}^{n} \mathcal{W}_{i,j}$, and $\vec{0}$ as an all zero vector in $\mathbb{R}^n$. Using the vector coloring of integer variables [4, 2], the SDP formulation of BCS can be stated as follows:

$$\begin{array}{ll} \text{Minimize} & \mathcal{C} \bullet \mathcal{X} \\ \text{Subject to} & \mathcal{D}_i \bullet \mathcal{X} \geq 0, \; \forall i = 1, \ldots, n + 3, \\ & diag(\mathcal{X}) = \mathbf{e}, \\ & \mathcal{X} \succeq 0. \end{array}$$

where $\mathcal{A} \bullet \mathcal{B}$ denotes the Frobenius inner product of matrices $\mathcal{A}$ and $\mathcal{B}$, i.e. $\mathcal{A} \bullet \mathcal{B} = \mathbf{Trace}(\mathcal{A}^t \mathcal{B})$, and $\mathcal{X} \succeq 0$ means that $\mathcal{X}$ is positive semidefinite. The coefficient matrix of the

objective function is

$$
\mathcal{C} = \begin{bmatrix} \frac{3}{4}\mathcal{W} & d & \vec{\mathbf{0}} & \vec{\mathbf{0}} \\ d^t & -\frac{1}{4}w_{all} & 0 & 0 \\ \vec{\mathbf{0}}^t & 0 & 1 & 0 \\ \vec{\mathbf{0}}^t & 0 & 0 & 1 \end{bmatrix}
$$

The constraint matrix corresponding to the lower bound in (4) has the following matrix form:

$$
\mathcal{D}_{n+2} = \begin{bmatrix} \tilde{\mathbf{0}} & \mathbf{e} & \vec{\mathbf{0}} & \vec{\mathbf{0}} \\ \mathbf{e}^t & 0 & 0 & 0 \\ \vec{\mathbf{0}}^t & 0 & 2n - 4k_{min} & 0 \\ \vec{\mathbf{0}}^t & 0 & 0 & 0 \end{bmatrix}
$$

where $\tilde{\mathbf{0}}$ is an $n \times n$ matrix of zeroes and $\mathbf{e}$ is an all-ones vector in $\mathbb{R}^n$. Also, the upper bound constraint in (5) has the following matrix form:

$$
\mathcal{D}_{n+3} = \begin{bmatrix} \tilde{\mathbf{0}} & -\mathbf{e} & \vec{\mathbf{0}} & \vec{\mathbf{0}} \\ -\mathbf{e}^t & 0 & 0 & 0 \\ \vec{\mathbf{0}}^t & 0 & 0 & 0 \\ \vec{\mathbf{0}}^t & 0 & 0 & 4k_{max} - 2n \end{bmatrix}
$$

Finally, it is easy to see that the matrices $\mathcal{D}_1$ to $\mathcal{D}_{n+1}$ are all zeros with a single "one" that moves along the main diagonal.

The final step in the construction of the BCS is to construct a feasible integer solution from the matrix $\mathcal{X}$. This rounding process identifies the set of values for indicator variables $y_1, ..., y_n$ and the set indicator variable y_{n+1}. In our experiments, we have used a rounding scheme based on a Cholesky decomposition of the matrix $\mathcal{X}$ [6] and a multivariate normal hyper-plane method that can be effectively derandomized. (See [9] and [13] for details on derandomization and rounding.) We perform the rounding step numerous times, each time checking to see if the result creates a canonical set. If it does, we then check to see if the cardinality $k \leq k_{max}$. Performing the check on max cardinality was done in the rounding step for technical reasons.


Algorithm 1 Approximation of BCS

1: Construct the graph $G(\mathcal{P})$ according to Section 3.
2: Form the semidefinite program from equation 3 and constraints 4 and 5 (See [2]).
3: Solve the semidefinite program using the algorithm in [12], obtaining PSD matrix $\mathcal{X}$.
4: Compute the Cholesky decomposition $X = \mathcal{V}^t\mathcal{V}$ [6].
5: Construct the indicator variables $y_1, ..., y_{n+1}$ and form the BCS V^* according to [13].

---

3. Similarity Measure

In this section, we present an overview of the many-to-many matching of abstract representations used in comput-

ing the similarity measure, previously studied in [7]. The matching algorithm is based on the metric-tree representation of labeled graphs and their low-distortion embeddings into normed vector spaces via spherical coding. This two-step transformation reduces the many-to-many matching problem to that of computing a distribution based distance measure between two such embeddings. To compute the distance between two sets of weighted vectors, we use the Earth Mover's Distance under transformation. For two given 2-D views, the algorithm provides an overall measure of similarity.

An overview of the approach is presented in Figure 3. For a given view, an object's silhouette is first represented by an undirected, rooted, weighted graph, in which nodes represent *shocks* [11] (or, equivalently, skeleton points) and edges connect adjacent shock points (Transition 1). The shock graphs will, in turn, be represented in terms of shock trees using a minimum spanning tree of the weighted shock graph (Transition 2). (For details on the construction of these trees, see [7].) Finally (Transition 3), we compute the distance between distributions using the Earth Mover's Distance under transformation.
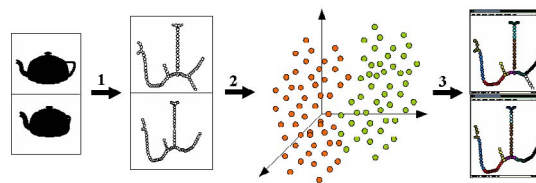
Figure 3. Computing similarity between two given views.

For the experiments, we compute the shock tree representation of every silhouette and use the many-to-many matching algorithm of [7] to compute the distance values among 2-D silhouettes corresponding to each 3-D object (pattern class). The outcome of this procedure is a distance matrix. To form the similarity matrix, and thus the graph $G(\mathcal{P})$, we use the reciprocal of the distance.

4. Experiments

In this section, we present an overview of the experiments we have performed to evaluate the method for computing BCS and its applicability to view selection for view-based 3-D object recognition. Each pattern class in our experiment corresponds to a set of 2-D views acquired from a 3-D synthetic object. Our database consists of 9 objects with 19 views each, for a total of 171 views. The 2-D views are acquired by sampling the surface of a view sphere centered on the object, with a representative view of each object is shown in Figure 4. In a view-based 3-D recognition

framework, our goal is to select a small number of views for each object in order to minimize search complexity at recognition time (assuming, for example, that recognition is performed using a linear search of the resulting selection).

For each of the 9 objects, we first create a BCS, with $4 \leq k_{min} \leq 6$, $6 \leq k_{max} \leq 9$, and compute, for each view on an object's view sphere, the identity of its closest canonical view in the object's canonical set. The resulting BCS's are combined to form a *summary set*, while the remaining views of each object are used as query views. Using our matching algorithm (see Section 3), we compared each query view to each element in the summary set. For each query view, we ranked the elements of the summary set in decreasing order of similarity and, in this ranking, noted the position of the query's nearest canonical view (in the query's object's BCS).

The correct canonical view was found to be among the top 6 elements in the ranking 90.6% of the time. The correct canonical view for a nearby query will not be top-ranked if there is another element of the summary set which is closer. This may happen when different objects share similar views which, in turn, may yield a summary set with similar elements. There is also an important trade-off between search complexity (size of the BCS bounds) and recognition accuracy (rank of the correct response). If, for example, the bounds are set too low given the complexity of the object, then there will be whole classes of object views that are not represented in the object's BCS. Thus, even though each query view on the object's view sphere will have a closest view in the BCS, that closest view may not be "nearby", thereby increasing the probability that an arbitrary member of another object's BCS (in the summary set) will be closer to the query. The larger an object's canonical set, the greater the model coverage, the better the recognition accuracy, and the greater the search complexity. These results are very preliminary, and a quantitative evaluation is in the works to study this trade-off.

Figure 4. Sample views of the 3-D objects.

5. Summary and Future Work

We have shown how to construct a bounded canonical set (BCS) to summarize a set. Through a series of experiments with a database of 2-D views of 3-D objects, we have demonstrated the applicability of BCS. Although we have applied our approach using a many-to-many matching algorithm, our approach could, in general, be used with any matching methodology. Our work is in its preliminary stages, and we plan to establish concrete performance bounds for our algorithm and derandomize it, as well as extend its application to other problem domains. With respect to modifying the formulation, we plan to study ways to effectively exclude ambiguous views from the BCS.

6. Acknowledgements

References

[1] C. M. Cyr and B. Kimia. 3d object recognition using shape similarity-based aspect graph. In *Eighth International Conference On Computer Vision (ICCV-01)*, pages 254–261, 2001.

[2] T. Denton, J. Abrahamson, and A. Shokoufandeh. Approximation of canonical sets and their application to 2d view simplification. In *CVPR*, June 2004.

[3] M. R. Garey and D. S. Johnson. *Computers and intractability: A guide to the theory of NP-completeness*. W.H. Freeman and Co., Baltimore, 1979.

[4] M. X. Goemans. Semidefinite programming in combinatorial optimization. *Mathematical Programming*, 79:143–161, 1997.

[5] M. X. Goemans and D. P. Williamson. .878-approximation algorithms for max cut and max 2sat. In *Twenty-sixth Annual ACM Symposium on Theory of Computing*, pages 422–431, New York, 1994.

[6] G. Golub and C. Loan. *Matrix Computations*. The Johns Hopkins University Press, Baltimore, 1996.

[7] Y. Keselman, A. Shokoufandeh, M. Demirci, and S. Dickinson. Many-to-many graph matching via metric embedding. In *CVPR)*, pages I–850– I–857 vol.1, June 2003.

[8] J. Koenderink and A. van Doorn. The internal representation of solid shape with respect to vision. *Biological Cybernetics*, 32:211–216, 1979.

[9] S. Mahajan and H. Ramesh. Derandomizing approximation algorithms based on semidefinite programming. *SIAM Journal on Computing*, 28(5):1641–1663, 1999.

[10] J. Shi and J. Malik. Normalized cuts and image segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22(8):888–905, 2000.

[11] K. Siddiqi, A. Shokoufandeh, S. Dickinson, and S. Zucker. Shock graphs and shape matching. *International Journal of Computer Vision*, 30:1–24, 1999.

[12] K. C. Toh, M. J. Todd, and R. Tutuncu. SDPT3 — a Matlab software package for semidefinite programming. *Optimization Methods and Software*, 11:545–581, 1999.

[13] V. V. Vazirani. *Approximation Algorithms*. Springer-Verlag, Berlin, Germany, second edition, 2003.

IEEE
COMPUTER
SOCIETY