

# Recognition of Shapes by Editing Their Shock Graphs

Thomas B. Sebastian, *Member, IEEE*, Philip N. Klein, and Benjamin B. Kimia

**Abstract**—This paper presents a novel framework for the recognition of objects based on their silhouettes. The main idea is to measure the distance between two shapes as the minimum extent of deformation necessary for one shape to match the other. Since the space of deformations is very high-dimensional, three steps are taken to make the search practical: 1) define an equivalence class for shapes based on shock-graph topology, 2) define an equivalence class for deformation paths based on shock-graph transitions, and 3) avoid complexity-increasing deformation paths by moving toward shock-graph degeneracy. Despite these steps, which tremendously reduce the search requirement, there still remain numerous deformation paths to consider. To that end, we employ an edit-distance algorithm for shock graphs that finds the optimal deformation path in polynomial time. The proposed approach gives intuitive correspondences for a variety of shapes and is robust in the presence of a wide range of visual transformations. The recognition rates on two distinct databases of 99 and 216 shapes each indicate highly successful within category matches (100 percent in top three matches), which render the framework potentially usable in a range of shape-based recognition applications.

**Index Terms**—Shape deformation, shock graphs, graph matching, edit distance, shape matching, object recognition, dynamic programming.

## 1 INTRODUCTION

THE recognition of objects in images is one of the central problems in computer vision. It is a challenging task mainly due to the *large degree of variations* which projections of objects in 2D images experience as a result of within-class variations, articulations and other object-based changes, as well as variations in position, viewing direction, illumination, etc. The extent by which representations of object shape capture such variations can significantly impact the effectiveness of recognition. In recognition applications, shapes have been represented as point sets [3], [4], [17], outline curves [2], [10], [36], [33], [26], [13], and by their medial axis [25], [31], [39], [42], [52], among others.

When a shape is represented using a point set (unorganized cloud of points) [18], matching is typically done using an assignment algorithm [17]. These methods have the advantage of not requiring *ordered* boundary points. However, if the similarity of two points is based on a local measure, the matching process does not necessarily capture the “coherence of shapes” in that the relationship among portions of shape may not be fully captured in the match, see Fig. 1a. This drawback is alleviated to some extent by using a global shape context [3], [4] where the relative location of all points is qualitatively captured; see Section 6.3. A drawback of such methods [4], [9] is their sensitivity to partial occlusion and articulation of parts, mainly due to the use of an extrinsic reference frame, see Fig. 1b. Also, an alteration

of fine shape-geometry of parts may significantly affect our perception, but not affect the overall distance, see Fig. 1b.

In curve-based representations, the matching is done either by aligning ordered point sets using an optimal similarity transformation [1], or by finding a mapping from one curve to another that minimizes an elastic performance functional consisting of stretching and bending terms [2], [10], [36], [33]. In the discrete domain, the minimization problem is often transformed into matching shape signatures with curvature, bending angle, or absolute orientation as attributes [13], [36]. The curve-based matching methods often suffer from one or more of the following drawbacks: asymmetric treatment of the two curves, sensitivity to sampling, lack of rotation and scaling invariance, and sensitivity to articulations, placement and deformations of parts.

Shapes have also been represented by their medial axis and variants. Zhu and Yuille model animate shapes using two deformable primitives, “worm” (a tubular structure) and “circle,” which lead to a skeletal-graph representation [52]. These skeletal graphs are matched using a branch and bound strategy, where the similarity between instances of primitives is defined in terms of the principal deformation modes in a Bayesian framework. The inherent instabilities of this skeletal representation are accounted for by defining skeleton operators and by using a user-specified model graph as an intermediary in the match process. This approach is shown to be effective under articulation, viewpoint variation, and occlusion. However, its applicability to represent inanimate objects is limited due to the choice of primitives used. Further, it is difficult to extend the number of primitives because the graph representation implicitly assumes that the edges correspond to the worm and the nodes correspond to the circle. In a related approach, Liu and Geiger [25] use the A\* algorithm to match shape-axis trees, which are defined by the locus of midpoints of optimally corresponding boundary points. The cost of matching edges in these trees is computed in terms of the cost of matching corresponding boundary curves. As in [52], graph topology changing operations are

- T.B. Sebastian is with GE Global Research Center, PO Box 8 KWC 218A, Schenectady, NY 12301. E-mail: sebastian@crd.ge.com.
- P.N. Klein is with the Department of Computer Science, Brown University, Box 1912, Providence, RI 02912. E-mail: klein@cs.brown.edu.
- B.B. Kimia is with the Division of Engineering, Brown University, Box D, Providence, RI 02912. E-mail: kimia@lems.brown.edu.

Manuscript received 12 June 2001; revised 23 July 2002; accepted 11 Apr. 2003.

Recommended for acceptance by S. Dickinson.

For information on obtaining reprints of this article, please send e-mail to: tpami@computer.org, and reference IEEECS Log Number 114344.

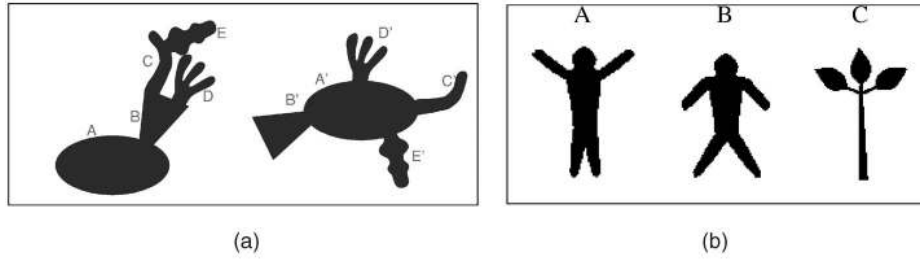


Fig. 1. (a) Shapes with matching subparts. An approach that is based on similarity of parts, but does not capture their relative spatial arrangement will have negligible cost in matching these two shapes since the cost of matching pairs  $(A, A')$ ,  $(B, B')$ ,  $(C, C')$ ,  $(D, D')$ , and  $(E, E')$  is zero. (b) Shape  $B$  results from an articulation of parts in shape  $A$ . This similarity is not captured by shape context [4] since articulation (disproportionately) affects the global signature (shape context) of each point. In addition, those changes in the geometry of parts which do significantly affect the perception of shape do not cause a large change in the shape context, resulting in  $dist(A, B) > dist(A, C)$ .

used to deal with transformations like articulation and occlusion. However, their algorithm uses unordered trees and does not necessarily preserve the ordering of edges at nodes in the final match, see Fig. 2. In addition, the topology-changing operators used in both these methods [52], [25] are graph-based, and not motivated by changes in the underlying shape. This means that the costs of these operators that are not independent of one another, may not be consistent with one another. Consequently, the cost of the match can have discrete jumps. Instead, the cost of deletion operator should be consistent with the fact that deleting an edge in the graph is the limit of matching that edge to a zero-length edge.

A representation derived from viewing the medial axis as singularities formed during a propagation from boundaries, i.e., Blum's grassfire, is the *shock tree* or *shock graph* [41]. In this approach, isolated points like branch points, end-points, and those medial axis segments (links), where shocks are monotonically flowing give rise to a tree or a graph. The approach in this paper [37], uses a graph structure where shocks with isolated point topology are nodes and those with curve topology are links; we refer to this as the Kimia et al.'s *shock graph*. See Section 2 for details. This representation (tree versus graph) was used in a similar form in Sharvit et al. [39], Kimia et al. [20]. In an alternate representation [42], [31], the oldest shock is selected as the root of a tree and connected shocks of the same type are mapped as nodes. Edges in this tree are placed between adjacent shock types such that the parent node has a higher time of formation than all the child nodes. We refer to this as the Siddiqi et al.'s *rooted shock tree* to facilitate a discussion on the two types of shock representations. While this representation captures the shock hierarchy

well, it maps shock segments with directed flow to a point (node), where the dynamic information is implicit. For example, the planar order at nodes is not represented, see Fig. 2. In addition, the selection of the oldest shock as the root of the tree, leads to an instability in the shock-tree representation, as relatively small change in the shapes can lead to a shock tree of very different topology, see Fig. 3. This may result in erroneous matches unless explicitly accounted for by additional measures.

We now present shape matching methods that use these shock-based representations. Sharvit et al. [39] and Kimia et al. [20] found the optimal pairwise assignment of shock graph nodes using the graduated assignment approach [17], where the similarity between each pair of nodes is defined based on the geometry of the shock segments linking them. This approach is fast, handles missing/extra nodes using slack variables, and gives good results for a database of 25 shapes. However, the errors in the matching process also point out a fundamental flaw in that the coherence of a shape is not necessarily preserved in the match process, as illustrated in Figs. 1 and 2.

Siddiqi et al.'s rooted shock tree representation has been used in conjunction with several different matching schemes to index into shape databases. First, an approach based on subgraph isomorphism [42], characterizes the topology of a shock subtree by the sum of the eigenvalues of its adjacency matrix. The geometric similarity between nodes is measured as the Hausdorff distance between shock branches that are aligned by an optimal affine transformation, see Fig. 4. These two similarity measures are combined in the tree matching procedure, which starts at the roots of the trees and proceeds through the subtrees in a depth-first fashion. This matching strategy is thus sensitive to the choice of roots in the shock trees, see Fig. 3.

Another approach based on matching rooted shock trees relies on finding maximal cliques of an association graph [31]. The main idea behind this approach is that maximal subtree isomorphism between rooted trees induces a maximal clique in the corresponding tree association graph which can be found as the global maximum of a quadratic function. The geometric similarity between nodes in this approach is defined in terms of differences in attributes of the shocks, namely, location, time of formation, orientation, and velocity. This approach has been extended to handle many-to-many correspondences [32] and to match free trees [30]. A third approach [47] uses approximate edit distance between shock trees, where the edit costs are restricted to be uniform. This approach relies on the computational equivalence between maximum common subgraph and tree edit distance, where

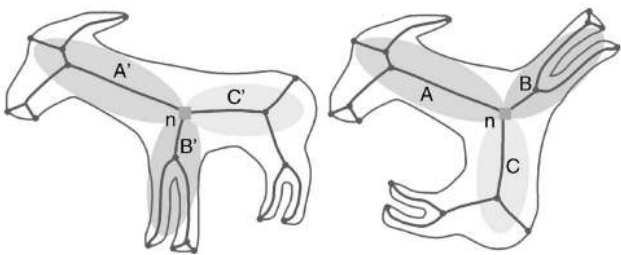


Fig. 2. This figure illustrates the importance of capturing the *planar order* at nodes of skeletal graphs in the match process. The shape on the left is reconstructed on the right after changing the ordering of branches from  $(A', B', C')$  to  $(A, C, B)$ . This reordering of branches results in two clearly different shapes. However, an approach that does not explicitly represent order at graph nodes cannot distinguish between these shapes because there is a zero-cost match between portions of the two shapes (highlighted by ellipses of the same shade of gray for purposes of illustration).

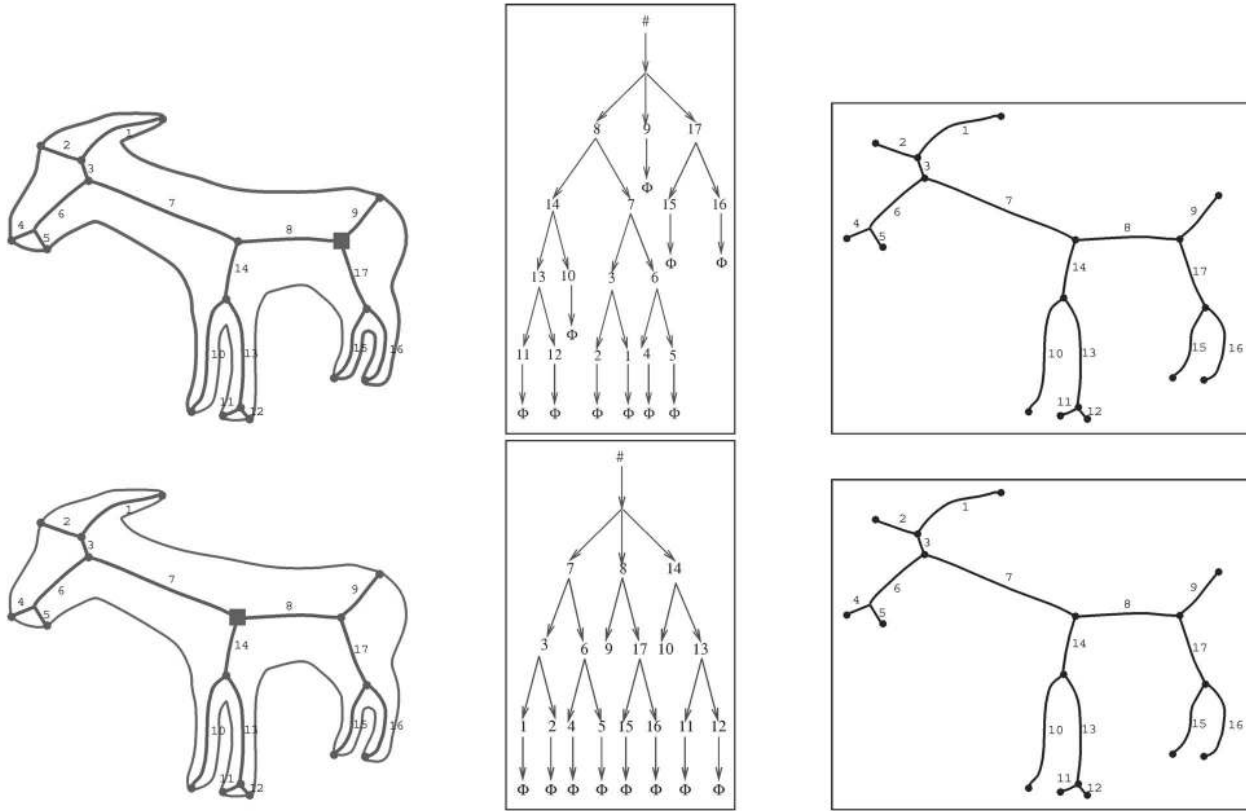


Fig. 3. This figure compares the shock graph representations of Siddiqi et al. [42], [31], and Kimia et al. [39], [20]. The original shapes and shocks are shown in the left column. The oldest shock is indicated by a square (black). The shock graphs of Siddiqi et al. and Kimia et al. are shown in the middle and right columns, respectively. Note that a relatively small change in the shape causes the oldest shock to change, leading to a significant change in the topology of Siddiqi et al.'s shock representation, which may result in erroneous matches. In contrast, the graph topology in the right column remains stable.

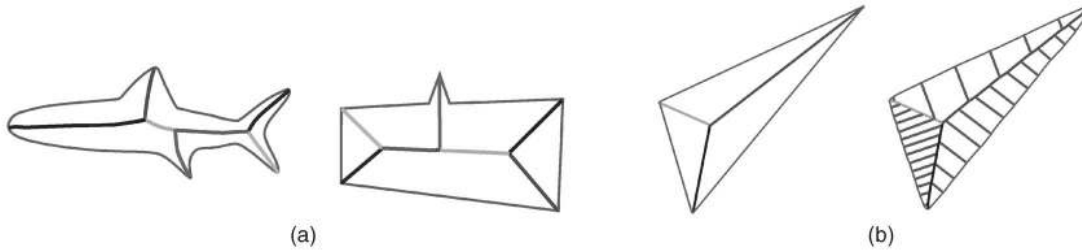


Fig. 4. (a) Shapes that are very dissimilar can have the same shock graph topology and (b) vice versa. Hence, graph-matching approaches based on topology without using geometrical information [8], [50] are not applicable for shock graph matching.

the cost of relabeling an element is assumed to be less than the cost of deleting it and reinserting one with a different label [7]. The similarity of shock edges is measured in terms of the length of the corresponding boundary segments [48].

The above shock-based matching approaches [20], [39], [31], [42], [47] have been shown to perform well in indexing into small databases of shapes (less than 30 shapes), but their performance for larger databases has not yet been examined. We expect that, for a recognition method to be effective for a much larger database, i.e., one depicting an extensive range of visual transformations including occlusion, viewpoint variation, and articulation, the matching method should be able to handle the inherent *instabilities of the symmetry-based representations*, as formally classified in [15], [53]. In other words, the matching strategy has to explicitly relate shapes on either side of an instability, which have different shock graph topologies but similar shape, with negligible matching cost.

We now present a brief overview of the proposed approach for comparing 2D shapes that addresses some of these issues. The main idea is to treat each shape as a point in a shape space and define the distance between two shapes in terms of the minimum-cost deformation path connecting them. Since there are infinitely many ways to deform a shape, the space of shapes and of deformations must be “discretized” to reduce the dimensionality of the search to practical limits. We address this issue by first defining an equivalence class on shapes, where all shapes with the same shock graph topology are considered equivalent. We then define as equivalent all the deformation paths that have the same set of transition shapes (instabilities of the shock graph representation that form the boundaries between shape equivalence classes), see Fig. 5. The representation of the deformation paths explicitly in terms of the transition shapes allows us to relate shapes on either side of the instability with negligible cost. The shock transitions can be effectively mapped to an “edit” operation



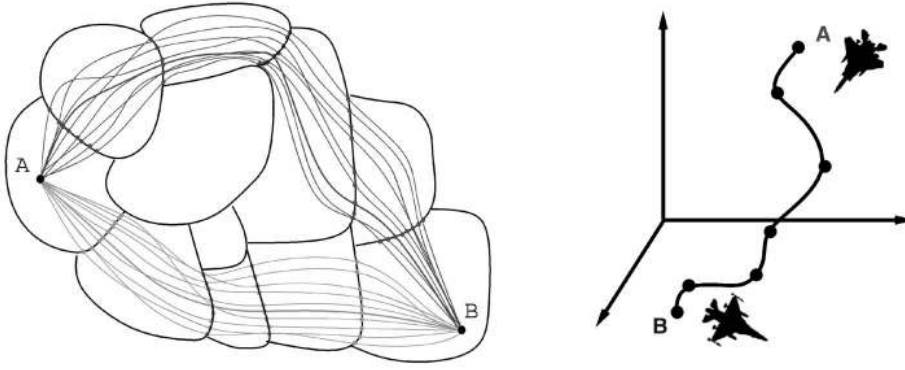


Fig. 5. Every pair of shapes in the shape space are related by an infinite number of deformation paths, few of which are shown here. Each deformation path can be effectively characterized by the sequence of transitions (represented by dots) experienced by the shape in the morphing process.



Fig. 6. A preview of the correspondence between two fish based on shock graph edit distance. Same colors indicate matching shock branches, and gray colored branches in the shock graphs have been pruned. This color scheme is used throughout the paper.

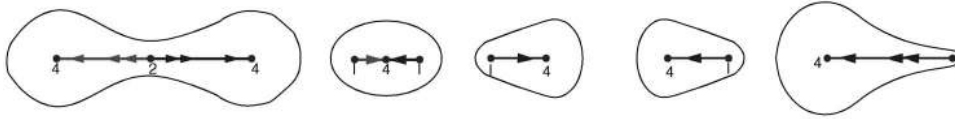


Fig. 7. From [15], the geometric locus and graph topology of the medial axis cannot tell these shapes apart but the shock graph can.

on the shock graph in the discrete domain. The cost for a deformation path is defined by summing the costs of individual edit operations that are, in turn, computed in a consistent fashion by summing local shape differences. We employ a graph edit-distance algorithm which was developed in [22], [23] to find the globally optimal path in polynomial time, see Fig. 6. The proposed approach is implemented and is shown to be robust in the presence of a variety of visual transformations such as articulation, partial occlusion, boundary perturbations, viewpoint variation, etc., and gives intuitive results for indexing into shape databases.

The main contributions of this paper are as follows:

1. Adopt and use a *planar ordered shock graph* representation where nodes and edges are distinct types of shocks as a basic representation for matching shapes.
2. Discretize the space of shapes and of deformation paths using shock transitions that makes the problem of finding the optimal deformation path practical.
3. Connect transitions to edits and assign costs for shock-graph edit operations such that they are consistent with each other.
4. Employ an edit-distance algorithm to find the optimal solution of the discretized version of the problem.
5. Illustrate the effectiveness of the approach in the presence of various visual transformations, and for indexing into shape databases.

The paper is organized as follows: Section 2 reviews the basic concepts and develops the notion of a shock graph based on a formal classification of shock points, its instabilities under deformations, and its use in reconstructing the shape. In Section 3, the idea of discretizing the shape space based on defining equivalence classes on shapes and

their deformations, and a restriction of deformation paths to pairs of simplifying paths is described. The edit-distance algorithm for finding the optimal path is reviewed in Section 4. Section 5 describes how the cost of each edit operation is assigned in a consistent fashion. Section 6 examines matching results under a variety of transformations, shows indexing results for two databases, and finally, compares the proposed approach to two recent methods, Pelillo et al.'s rooted shock tree matching [31] and shape-context matching [4]. Section 7 concludes the paper.

## 2 SHAPE REPRESENTATION USING SHOCK GRAPHS

### Definition (Symmetry Set, Medial Axis, Shock Structure).

The symmetry set (SS) of a shape is the closure of the locus of centers of bitangent circles [6]. The medial axis (MA) of a shape is defined as the closure of the locus centers of maximal (bitangent) circles [5]. The shock structure (SH) of a shape arises from a “dynamic” interpretation of the medial axis, as the locus of singularities (shocks) formed in the course of wave propagation (as in Blum’s grass-fire) from boundaries, together with an associated direction and speed of flow.

Shock segments are curve segments of the medial axis with monotonic flow, and give a more refined partition of the medial axis segments [40], see Fig. 7. These shocks are obtained either by detecting the singularities of the evolving curve in a curve evolution/PDE approach [19], [41] or in an Eulerian-Lagrangian propagation which combines wave propagation and computational geometry concepts in a unified framework [46], [45], [54]. All shock graphs used in this paper are from the latter approach.

Medial axis and shock points were formally classified in [15], based on the order of contact of circles with the shape



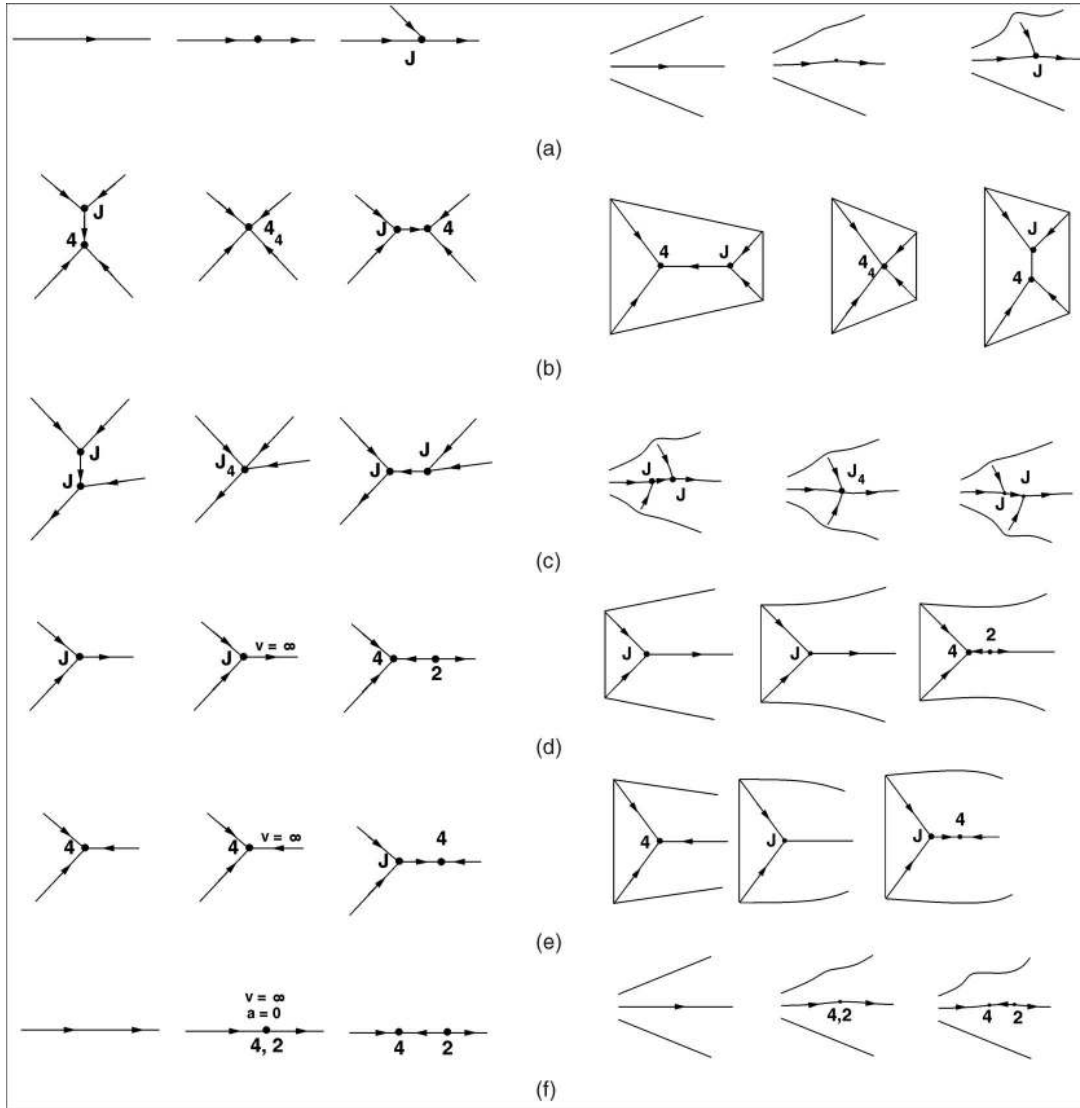


Fig. 10. From [15], the first three columns show a schematic description of the six possible transitions of the shock structure of a closed curve, and the last three columns show examples of corresponding shape deformations. The central column in each group represents the transition point in the deformation from left to right, or right to left columns. In the notation of [15], (a) is the  $A_1A_3$  transition, (b) and (c) are the two types of  $A_1^4$ , (d) and (e) are the two types of  $A_1^3$  with infinite velocity, and (f) is the  $A_1^2$  point with infinite velocity and zero acceleration. The graph operations to make the right and left columns equivalent are: splice, contract (two types), and merge (three types).

for certain shapes), an infinitesimal change in the shape can cause a large (abrupt) change in the shock graph topology, see Fig. 9b. The shapes where the graph topology is altered by a small perturbation are precisely the points of *instability* of the medial axis/shock graph. These instabilities, namely, all generic *shock transitions* along a one-parameter family of deformations have been formally enumerated and classified [15], see Fig. 10. A similar situation exists in 3D, as recently studied in [16], where generic transitions are classified and enumerated in terms of the order of contact of spheres with the surface.

In the next section, we discuss how these transitions, or instabilities, are used to our advantage to partition the shape space into shape cells and to group deformation paths into groups of deformation bundles, in effect, discretizing the search in the shape space for the optimal path between two shapes.

### 3 PARTITIONING THE SHAPE SPACE

This section describes our approach to defining the distance between shapes. Each shape is viewed as a point in the *shape space*, at this point, simply the collection of all “shapes,” where a neighbor of a shape is the set of infinitesimal deformation of its outline. A deformation sequence between two shapes is a path (sequence of points) in the shape space, see Fig. 5. We define the distance between two shapes in an infinitesimal neighborhood and then define the distance between two shapes  $A$  and  $B$  as the cost of the least-cost deformation path between them and denote it by  $D(A, B)$ . However, there are infinitely many deformation paths between two shapes and a key bottleneck in this deformation-based approach to matching shapes is the impracticality of the computational representation of the high-dimensional space of deformations. We propose three steps in “discretizing” this high-dimensional space such that the entire shape space is effectively and densely represented. First, we define an equivalence class on shapes based on shock graph topology.

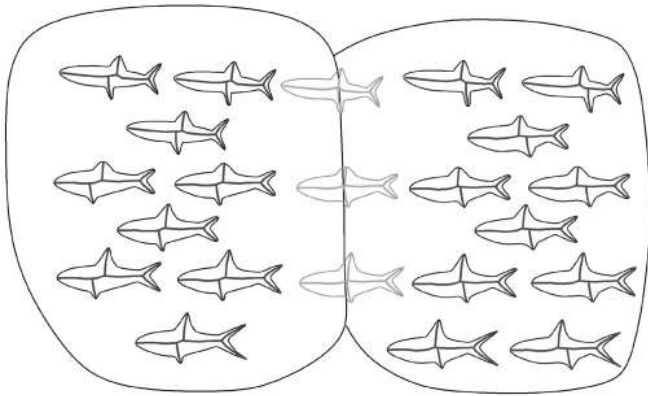


Fig. 11. Two shape cells and the transition shapes that occupy the boundary between them.

Specifically, we use the constancy of shock graph topology under deformations to define a local neighborhood for most shapes.

**Definition 2.** A shape cell is a collection of shapes which have identical shock graph topology.

At shock transitions, this neighborhood topology breaks down, i.e., two shapes on either side of a transition shape are not connected using this notion of a neighborhood, while the perceptual distance between them is negligible, see Fig. 11. Note that this is primarily the reason why recognition based on a direct subgraph isomorphism of this representation would break down. This motivates an explicit embedding of the transitions in the definition of a shape neighborhood as “seams” between the cells of the shape space. Specifically, the remedy for arriving at a broader notion of a neighborhood requires discretizing the space of all deformation paths by

considering all deformation paths that encounter the same set of transition shapes to be equivalent.

**Definition 3.** Two one-parameter shape deformation sequences from shape  $A$  to shape  $B$  are equivalent if they pass through an identical sequence of shock transitions. A shape deformation bundle from shape  $A$  to shape  $B$  is the set of equivalent one-parameter families of deformations from  $A$  to  $B$ .

In other words, two deformation paths are equivalent if they traverse through the same shape cells. Further, we represent each deformation bundle using the set of transition shapes it passes through, see Fig. 12.

We note further that those deformations paths where features are added and then removed are clearly not optimal and their considerations is wasteful, see Fig. 13. Hence, as a third measure, we avoid candidate deformation paths which unnecessarily venture into more complicated shapes. In order to avoid complexity-increasing deformation paths, we represent each deformation path by a pair of “simplifying” deformation paths from  $A$  and  $B$  to a “simpler” common shape  $C$ , see Fig. 14a. The notion of simplicity is derived from the transitions themselves, see Fig. 10: the first transition splices off branches, the second and third transitions “symmetrize” the shape, the fourth transition removes parts at “necks,” the fifth transition moves the shape to a rounder shape, and the last transition removes wavy patterns on the boundary. The deformation of a shape to the adjacent simpler transition shape restricts the shape in that a degree of freedom is removed and leads to a shape cell of lower dimensionality, see Fig. 12. In the discrete domain, the deformation of a shape to the adjacent transition shape is represented by an *edit operation* on its shock graph.

With the above discretizations the deformation paths between two shapes  $A$  and  $B$  can be represented by a finite

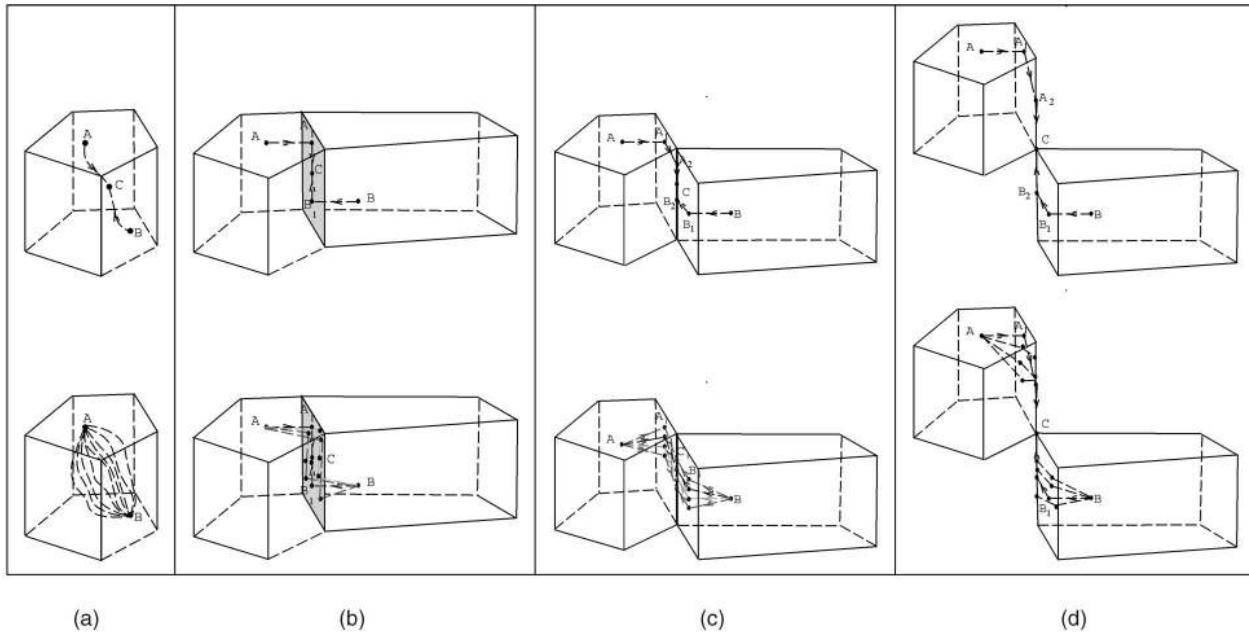


Fig. 12. A highly schematic description of a *deformation bundle* by resorting to the analogy of connecting points embedded in polyhedral cells (a very coarse analogy for shape cells). Two neighboring shape cells can meet at interfaces which upon arrival at the interface remove one, two, three, or more degrees of freedom, as compared to that of the shape cell. Of course, the shape space is a much higher dimensional space than the 3D Euclidean polyhedral space. The path illustrates a projection of  $A$  and  $B$  to the transition interface at  $A_1$  and  $B_1$ , respectively; from  $A_1$  and  $B_1$  to  $A_2$  and  $B_2$ , etc., until both  $A_n$  and  $B_n$  belong to the same shape cell. Bottom row: each path is a representative of a collection of paths or a deformation bundle which share the same sequence of transitions. In the shape  $C$ , a circle serves the role of the corner vertex  $C$ .



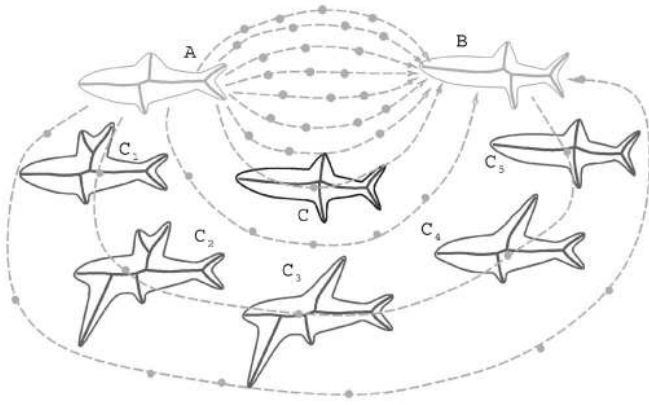


Fig. 13. Two deformation paths among many are highlighted. The sequence  $(A, C_1, C_2, C_3, C_4, C_5, B)$  is not optimal since features are first added and then removed. In order to avoid such paths, we represent each deformation path by a pair of “simplifying” deformation paths starting from  $A$  and  $B$ , respectively, and leading to a common shape  $C$ .

number of transition shapes. For example, in Fig. 14b, seven transition shapes result from applying one simplifying transition on shape  $A$  (fish). In the second step of the transition sequence, each of these seven shapes undergoes a similar deformations towards the next applicable transition. In a few steps, the shock graph reaches highly simplified shapes, e.g., elongated approximations of the object (only three steps are illustrated; additional steps are not shown). A complete path consists of a pair of simplifying deformation paths which end up in shapes belonging to a common shape cell. These two deformed shapes are then related by a continuous transformation of the shock graph attributes using a “deform” edit operation.

In summary, the three steps of 1) partitioning of the shape space into *shape cells*, 2) definition of a *deformation bundle* as an equivalence class of deformations based on the sequence of transitions along each deformation, and 3) restriction of the search to paths consisting of two simplifying subpaths from shapes  $A$  and  $B$  to a common shape cell, constitute a

discretization of the search space, and render the search of optimal deformation path practical.

#### 4 EDIT-DISTANCE ALGORITHM

Despite the above discretization and the tremendous reduction in dimensionality, numerous paths remain to be considered, as illustrated in Fig. 14. Thus, we need an efficient algorithm to find the optimal path among all possibilities. We employ a polynomial-time edit-distance algorithm developed in [23], [22] for comparing ordered, unrooted planar trees, which we review in this section.

The above notion of edit-distance was originally proposed to compare character strings [49], where there are three kinds of edit operations: 1) *deleting* a character, 2) *inserting* a character, and 3) *changing* one character into another; consider matching the word “HELLO” to “HELO,” “HELLOW” and “JELLO,” respectively. Once costs are assigned to each edit operation, edit-distance is defined to be the minimum cost of the sequence of operations required to convert one string to another and is typically computed using dynamic programming. Note that the *delete* operation and the *insert* operations are inverses of one another. An equivalent way to describe the same measure is to omit the *insert* operation and ask for the minimum cost set of operations applied to the two strings that will succeed in transforming them into a common string.

The notion of edit-distance has been generalized to comparing ordered, rooted trees [44], [51] and ordered, unrooted trees [21], where analogous edit operations are defined as: 1) *contract* an edge, 2) *uncontract* an edge, and 3) *change* the label of an edge. As before, one can omit the *uncontract* operation, and ask for the minimum-cost set of operations taking the two trees to a common tree. Edit distance has been used in many computer-vision applications such as handwritten character and word recognition [12] and stereo matching [27].

In applying the edit-distance approach to comparing shock graphs, traditional tree edit operations do not suffice.

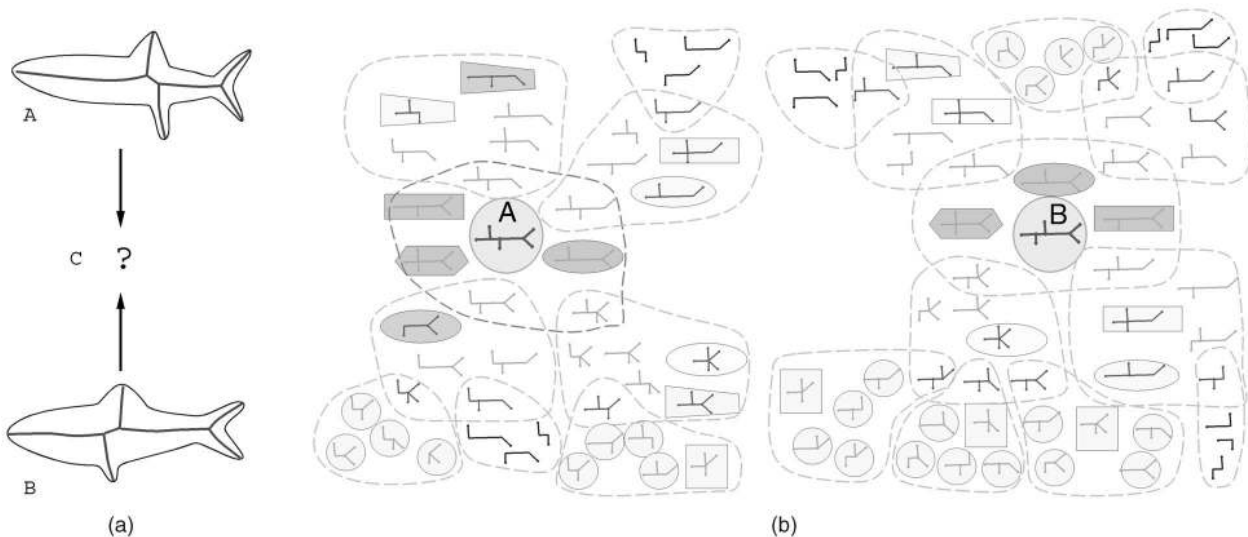


Fig. 14. (a) The optimal path of deformation between two shapes is obtained by searching all pairs of simplifying deformation paths leading to a common shape cell represented by  $C$ . (b) A hand-drawn sketch of how the space of one-parameter family of deformations for each shape can be discretized using transition shapes that result from simplifying edits. Shapes with equivalent shock graph topology are marked by common icons with common shape/coloring. The optimal path in this case goes through the gray-shaded hexagonal surround.



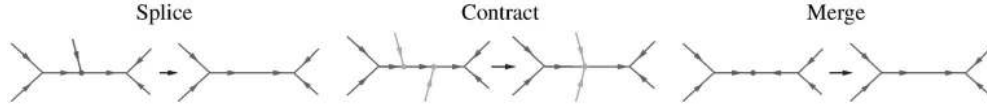


Fig. 15. Three shock-graph edits that change the graph topology.

Rather, four groups of edit operations which are derived from the shock transitions are necessary, see Fig. 15:

1. The *splice* operation deletes a shock edge that is a leaf and merges the remaining two.
2. The *contract* operations deletes a shock edge connecting two degree-three nodes.
3. The *merge* operations combines two edges at a degree-two node.
4. We also define a *deform* edit to match two shock edges with different attributes.

As in traditional edit-distance, once the costs for these edits are determined, the edit-distance is defined to be the minimum-cost of a set of operations that transforms the two graphs into a common graph.<sup>1</sup> The minimum-cost set of operations can be found using dynamic programming, as sketched in Appendix B. The definition and method of assigning costs to the edit operations is discussed in Section 5.

The computational complexity of the algorithm is  $O(n_1^3 n_2^3)$ , where  $n_1$  and  $n_2$  are the number of nodes in  $T_1$  and  $T_2$ , respectively. However, the use of a heuristic allows for a reduction to  $O(n^3 k^3)$  where  $n = \max(n_1, n_2)$  and  $k = \max(k_1, k_2)$ , where  $k_i$  are the diameter of  $T_i$ . See Appendix B for details.

## 5 COST OF EDITS

We now discuss how costs are assigned to each edit operation. Observe that edit costs have to be consistent with each other, and also with our perceptual metrics of similarity. While the former can be developed mathematically, the latter can only be judged subjectively. Our basic approach is to first derive the cost of the *deform* edit, i.e., the distance between two shapes within the same shape cell (identical shock topology), and then derive the cost of other edits as the limit of the deform cost as the shape moves to the boundary of the shape cell (transition shape). This will ensure that edit costs are consistent with each other.

We derive the deform edit cost by summing over local shape differences which are computed as differences between corresponding shock attributes. This approach is based on extending a previously developed metric for aligning curves [33], [34], [36], which is reviewed below in Section 5.1. First, however, we observe that in many applications, such as clustering and indexing into large databases, the metric properties of the distance are critical for database organization. The following proposition shows that the deformation-based notion of distance between two shapes  $A$  and  $B$  denoted by  $D(A, B)$  is a metric.

**Proposition 1.** *The distance  $D(A, B)$  is a metric, i.e., it satisfies the following properties for shapes  $A$ ,  $B$ , and  $C$ :*

1. For the traditional formulation, one also includes operations that are inverses to *splice*, *contract*, and *merge*, and defines edit-distance to be the minimum cost of a sequence of edits to morph one graph to the other.

- 1)  $D(A, A) = 0$ , 2)  $D(A, B) = D(B, A)$ , and 3)  $D(A, B) \leq D(A, C) + D(C, B)$ .

**Proof.** Properties 1 and 2 follow directly from the definition. To prove property 3, let  $P_{AB}^*$ ,  $P_{AC}^*$ , and  $P_{CB}^*$  be the minimum-cost deformation paths between  $(A, B)$ ,  $(A, C)$ , and  $(C, B)$  with costs  $D(A, B)$ ,  $D(A, C)$ , and  $D(C, B)$ , respectively. Consider the deformation path between  $A$  and  $B$  that passes through  $C$  (Fig. 16), consisting of the minimum-cost deformation paths  $P_{AC}^*$  and  $P_{CB}^*$ . The cost of this path is  $D(A, C) + D(C, B)$  by definition. From the optimality of the  $P_{AB}^*$ , it follows that  $D(A, B) \leq D(A, C) + D(C, B)$ .  $\square$

### 5.1 Matching Curves by Optimal Alignment

The approach in [33] to comparing two curves  $\mathcal{C}$  and  $\hat{\mathcal{C}}$  consists of finding the minimum-cost deformation of one curve to another, where the cost is defined as the sum of “stretching” and “bending” energies [10], [43], [2]. Let  $\mathcal{C}$  and  $\hat{\mathcal{C}}$  be parameterized by arclength  $s$  and  $\hat{s}$  with tangent orientations  $\theta(s)$  and  $\hat{\theta}(\hat{s})$ , respectively, see Fig. 17a. The basic premise of the approach is that the cost of matching entire curves can be expressed as the sum of the costs of matching infinitesimal subsegments, which is a combination of length and curvature differences, see Fig. 17c. Specifically, the cost of matching infinitesimal segments  $d\mu$  is defined as  $|d\hat{s} - ds| + R|d\hat{\theta} - d\theta|$ , where  $R$  is a constant related to the average sample length. The problem is then cast as minimizing a functional over all possible *alignments* between the two curves. To ensure symmetric treatment of the curves, the alignment is represented not as a function but as a pairing of points on the two curves leading to the notion of an *alignment curve*  $\alpha$ ,  $\alpha(\xi) = (s(\xi), \hat{s}(\xi))$ , where  $\mathcal{C}(s(\xi))$  and  $\hat{\mathcal{C}}(\hat{s}(\xi))$  denote the points on the two curves that correspond, Fig. 17b. The optimal alignment curve  $\alpha^*$  is found by minimizing the functional

$$\mu[\alpha] = \int_0^{\tilde{L}} \left[ \left| \frac{d\hat{s}}{d\xi} - \frac{ds}{d\xi} \right| + R \left| \frac{d\hat{\theta}(\hat{s})}{d\xi} - \frac{d\theta(s)}{d\xi} \right| \right] d\xi, \quad (3)$$

where  $\tilde{L}$  is the length of the alignment curve. A key advantage of using the alignment curve is that the alignment can be expressed in terms of a single function. We choose the angle between the tangent of alignment

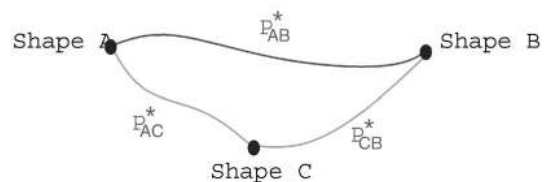


Fig. 16. This figure illustrates that the distance satisfies the triangle inequality. Let  $P_{AB}^*$  be the minimum-cost deformation path between shapes  $A$  and  $B$ . The deformation path between shapes  $A$  and  $B$  consisting of minimum-cost deformation paths  $P_{AC}^*$  and  $P_{CB}^*$  is typically not optimal.

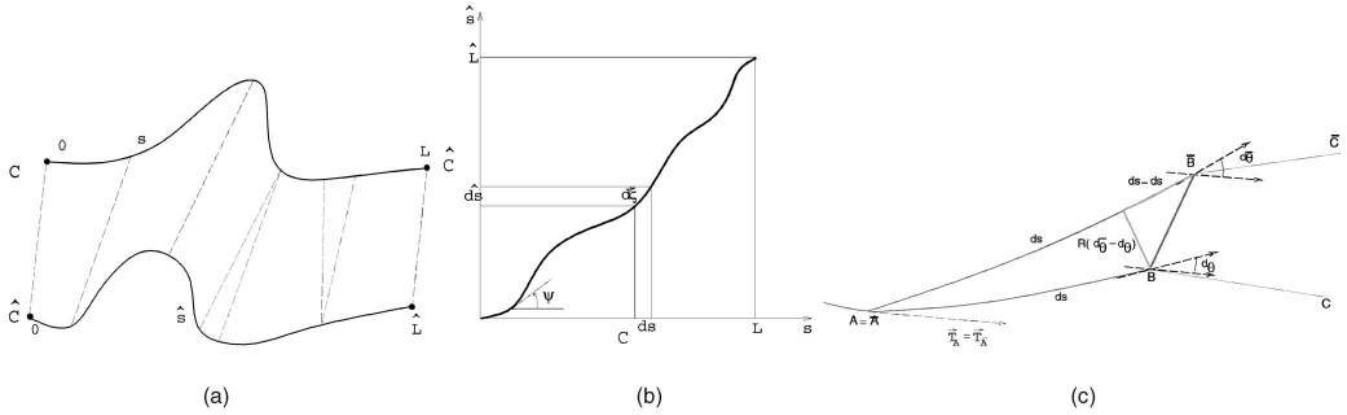


Fig. 17. (a) An alignment of two curves  $C$  and  $\hat{C}$  is represented by a pairing of points. (b) A symmetric treatment of this pairing leads to a notion of an alignment curve. (c) The cost of deforming an infinitesimal segment  $AB$  to segment  $\hat{A}\hat{B}$ , when the initial points and the initial tangents are aligned ( $A = \hat{A}$ ,  $\vec{T}_A = \vec{T}_{\hat{A}}$ ), is related to the distance  $B\hat{B}$ , and is defined by  $|d\hat{s} - ds| + R|d\hat{\theta} - d\theta|$  [34], [36].

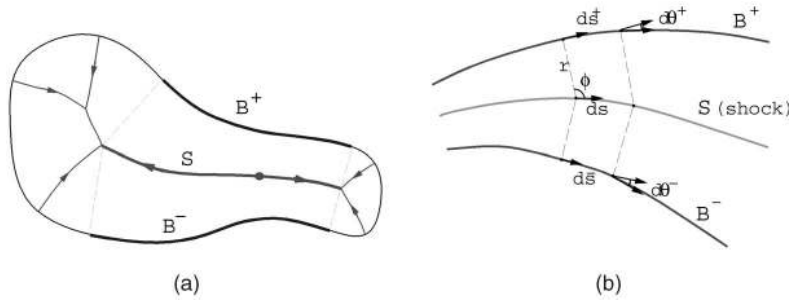


Fig. 18. The shock  $S$  and the corresponding shape boundary segments  $B^+$  and  $B^-$ .

curve and the  $x$ -axis,  $\psi(\xi)$ , see Fig. 17b. Then, (3) can be rewritten in terms of  $\psi$  as

$$\mu[\alpha(\psi)] = \int_0^{\hat{L}} [|\cos(\psi) - \sin(\psi)| + R|\kappa \cos(\psi) - \hat{\kappa} \sin(\psi)|] d\xi. \quad (4)$$

The optimal alignment curve is computed efficiently using dynamic programming [34], [36], and is reviewed in Appendix A. The distance  $d(C, \hat{C})$  between the curves  $C$  and  $\hat{C}$  is then defined as the cost of the optimal alignment,  $D(C, \hat{C}) = \mu[\alpha^*]$ .

## 5.2 Deform Cost

In analogy to this notion of similarity between curves, the deform cost is defined as the sum of “local shape differences” between corresponding shock segments. This difference is defined by extending the metric between two curves to a metric between two shock segments. Observe that each shock segment represents a pair of segments on the boundary of the shape, as shown in Fig. 18a. Hence, we can view the problem of deforming one edge in a shock graph to an edge in another shock graph in terms of deforming the corresponding boundary segments, i.e., as a “joint curve matching problem.” As in the case of a single pair of curves, we penalize stretching and bending of each pair of boundary segments in terms of length and curvature differences. However, this notion of joint curve matching of shape boundary segment pairs must not only rely on simultaneous pairwise intrinsic differences between the boundaries, but also on changes in the relative pose of the two boundary segments, see Fig. 19.

Consider a pair of edges from two shock graphs,  $S$  and  $\hat{S}$ , each parameterized by  $s$  and  $\hat{s}$ , respectively. Let the boundary segments corresponding to  $S$ ,  $B^+$ , and  $B^-$  be parameterized by  $s^+$  and  $s^-$ , with tangent orientations  $\theta^+(s^+)$  and  $\theta^-(s^-)$ , respectively, Fig. 18b. The boundary segments of  $\hat{S}$  are similarly defined. The cost of matching infinitesimal segments of the shock edges are defined as in curve matching by length differences  $|d\hat{s}^+ - ds^+| + |d\hat{s}^- - ds^-|$  and boundary curvature differences  $|d\hat{\theta}^+ - d\theta^+| + |d\hat{\theta}^- - d\theta^-|$  but, in addition, now augmented with width differences  $2(|\hat{r}_0 - r_0| + \int |d\hat{r} - dr|)$ , and relative orientation differences  $2(|\hat{\phi}_0 - \phi_0| + \int |d\hat{\phi} - d\phi|)$ . As in the case of curves, the *shock alignment curve* is used to mediate the

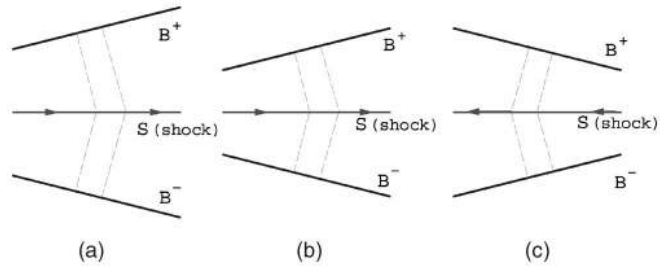


Fig. 19. This figure illustrates why summing the pairwise differences in corresponding boundary segments is not sufficient to measure the distance between two shock segments. In all three cases, the boundary segments are identical up to rotations and translations and the cost of matching each individual pair *independently* is zero. However, the cost of matching the pair as a whole should not be zero, as 1) the segments in (a) are farther apart than those in (b) suggesting that the shape in (a) is *wider* than the shape in (b), and 2) the segments in (b) are widening whereas those in (c) are narrowing.

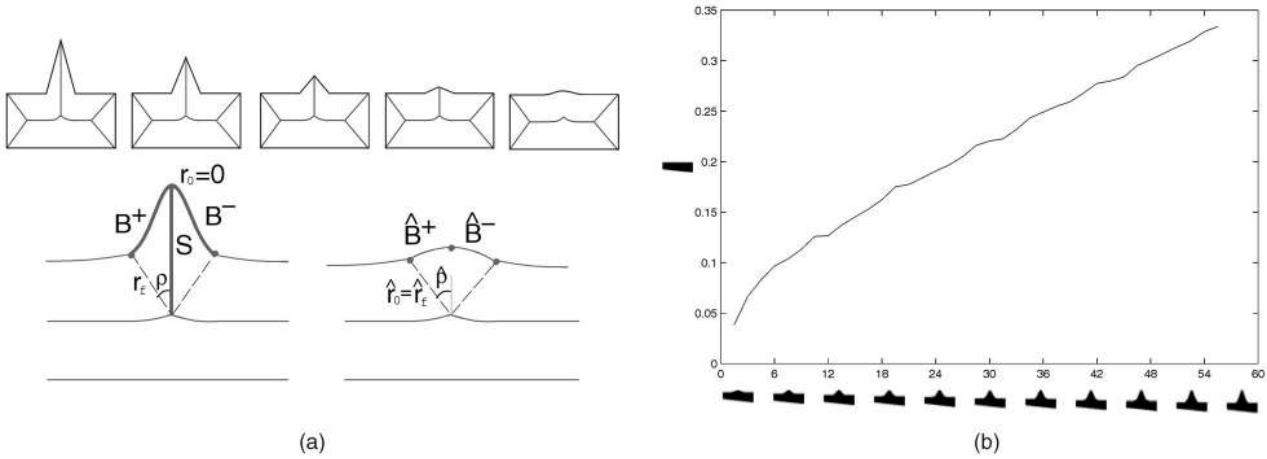


Fig. 20. (a) Splice operation removes a protrusion, and can be viewed as the limit of shrinking the protrusion and replacing it with a circular arc. A sharp protrusion is shown here, hence  $r_0 = 0$ . (b) The plot illustrates that splice cost behaves rather intuitively and increases almost linearly when matching a sequence of shapes with a protrusion of increasing length. The  $x$ -axis is the length of the protrusion in pixels and the  $y$ -axis is the normalized edit cost. Note that, in all cases, the optimal edit sequence involves the splice edit (pruning the protrusion).

match and the optimal alignment curve  $\alpha^*$  is found by minimizing the functional

$$\begin{aligned} \mu[S, \hat{S}; \alpha] = & \int_0^{\tilde{L}} \left[ \left| \frac{d\hat{s}^+}{d\xi} - \frac{ds^+}{d\xi} \right| + \left| \frac{d\hat{s}^-}{d\xi} - \frac{ds^-}{d\xi} \right| \right] d\xi \\ & + R \int_0^{\tilde{L}} \left( \left| \frac{d\hat{\theta}^+}{d\xi} - \frac{d\theta^+}{d\xi} \right| + \left| \frac{d\hat{\theta}^-}{d\xi} - \frac{d\theta^-}{d\xi} \right| \right) d\xi \\ & + 2|\hat{r}_0 - r_0| + 2 \int_0^{\tilde{L}} \left| \frac{d\hat{r}}{d\xi} - \frac{dr}{d\xi} \right| d\xi \\ & + 2R|\hat{\phi}_0 - \phi_0| + 2R \int_0^{\tilde{L}} \left| \frac{d\hat{\phi}}{d\xi} - \frac{d\phi}{d\xi} \right| d\xi, \end{aligned} \quad (5)$$

where  $\cos \phi = -\frac{dr}{ds}$  and  $\tilde{L}$  is length of alignment curve. Note that the reconstruction equation ((2) in Section 2) allows us to write  $\mu$  in terms of shock properties. Specifically, using  $ds^\pm = v^\pm ds$ , and  $d\theta^\pm = \kappa^\pm v^\pm ds$ , we have

$$\begin{aligned} \mu[S, \hat{S}; \alpha] = & \int_0^{\tilde{L}} \left[ \left| \hat{v}^+ \frac{d\hat{s}}{d\xi} - v^+ \frac{ds}{d\xi} \right| + \left| \hat{v}^- \frac{d\hat{s}}{d\xi} - v^- \frac{ds}{d\xi} \right| \right] d\xi \\ & + \int_0^{\tilde{L}} \left[ R \left( \left| \hat{\kappa}^+ \frac{\sqrt{\hat{v}^2 - 1}}{\hat{v}} \frac{d\hat{s}}{d\xi} - \kappa^+ \frac{\sqrt{v^2 - 1}}{v} \frac{ds}{d\xi} \right| \right. \right. \\ & \left. \left. + \left| \hat{\kappa}^- \frac{\sqrt{\hat{v}^2 - 1}}{\hat{v}} \frac{d\hat{s}}{d\xi} - \kappa^- \frac{\sqrt{v^2 - 1}}{v} \frac{ds}{d\xi} \right| \right) \right] d\xi \\ & + \int_0^{\tilde{L}} \left| \frac{1}{\hat{v}} \frac{d\hat{s}}{d\xi} - \frac{1}{v} \frac{ds}{d\xi} \right| d\xi + |\hat{r}_0 - r_0| \\ & + \int_0^{\tilde{L}} R \left| \frac{\hat{a}}{\hat{v}\sqrt{\hat{v}^2 - 1}} \frac{d\hat{s}}{d\xi} - \frac{a}{v\sqrt{v^2 - 1}} \frac{ds}{d\xi} \right| d\xi + R|\hat{\phi}_0 - \phi_0|. \end{aligned}$$

Finally, the deform cost of the shock edges  $S$  and  $\hat{S}$  is defined as the cost of the optimal alignment,

$$d(S, \hat{S}) = \min_{\alpha} \mu[S, \hat{S}; \alpha] \quad (6)$$

which is found by a dynamic-programming method described in [34], [36]. This method has  $O(n^2)$  complexity, where  $n$  is the number of samples along the shock segments. The average running time on an SGI Indigo (195 MHz) for matching two shock segments with 50 points

each is 0.17 seconds. Note that we currently compute the cost for all combinations of shock paths independently, leading to about 3-5 per minutes per match for the entire shock graph. However, there is a great deal of redundancy which can be avoided by revising our implementation to avoid duplicate matches.

### 5.3 Cost of Other Edits

Thus far, we have described how the *deform cost* between two shock segments is computed in an intrinsic manner. Other edits can be viewed as limiting cases of the corresponding deform operation, and penalized accordingly. The *splice* operation prunes a leaf edge in the shock graph. It can be viewed as the result of a deformation sequence that shrinks a shock branch to a point. From the shape perspective, this is equivalent to replacing the boundary segment corresponding to the protrusion with a circular arc. Hence, the splice edit cost can be viewed as the cost of deforming the protrusion to the circular arc and is derived from (5). Observe that the alignment curve, in this case, is horizontal as the second shock edge is a point in this case. The first two terms reduce to the difference in lengths between the protrusion and the circular arc, i.e., the difference in lengths between  $B^\pm$  and  $\hat{B}^\pm$  in Fig. 20, which is given by  $|\int ds^+ + \int ds^- - 2\rho r|$ . The third and fourth terms reduce to 0 as the orientation of the tangents at each endpoint remains the same. The fifth and sixth terms simplify to  $4|\hat{r}_0 - r_0|$  as  $\int_0^{\tilde{L}} \left| \frac{d\hat{r}}{d\xi} - \frac{dr}{d\xi} \right| d\xi = \int_0^{\tilde{L}} |dr| = |r_f - r_0| = |\hat{r}_0 - r_0|$ . Finally, the last two terms reduce to 0 under the optimal pairing of boundary segments. Thus, we get the splice cost to be

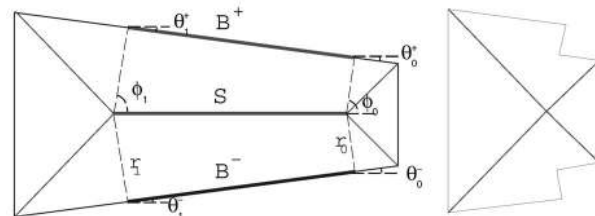


Fig. 21. Contract is not a local operation.

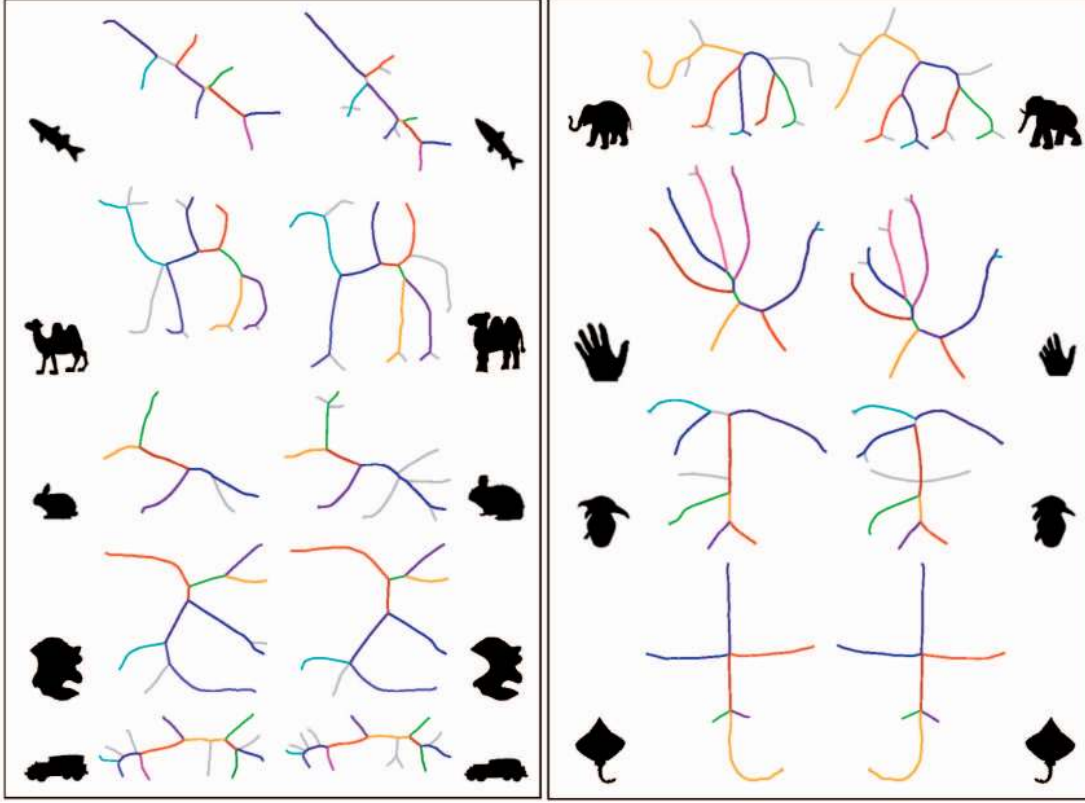


Fig. 22. The result of matching the shock graphs for a few pairs of shapes in the database. Same colors indicate matching shock branches, while the gray colored edges in the shock graphs indicate that they are spliced or contracted. Observe that the correspondence is intuitive in all cases. For example, the head, the fins, and the tails of the two fishes (top left) correspond. Also, note that when matching the elephants (top right), the shock edge corresponding to the tail of the elephant (on the left) is pruned as the tail on the other is occluded.

$$\mu_s = \left| \int ds^+ + \int ds^- - 2\rho\hat{r}_0 \right| + 4|\hat{r}_0 - r_0|. \quad (7)$$

The *contract* operation removes an edge between high degree nodes, see Fig. 21. Unlike other edit operations, the contract operation is not a local operation in that it affects the neighboring shock edges also. However, as an approximation, we measure only the local effect of the contract operation by deforming the contracted edge to a point, and its cost is derived from (5). The first two terms reduce to the lengths of the boundary segments  $B^+$  and  $B^-$ , and is given by  $\int ds^+ + \int ds^-$ . As the boundary segments  $B^+$  and  $B^-$  are mapped to a point, the third and fourth terms reduce to the differences in orientations of the tangents at each endpoints, and is given by  $R((\theta_0^+ - \theta_1^+) + (\theta_0^- - \theta_1^-))$ . The fifth and sixth terms reduce to zero and  $2|r_0 - r_1|$ , respectively. The seventh and eighth terms reduce to zero and  $2R \int d\phi$ , respectively. Thus, the contract cost is given by

$$\begin{aligned} \mu_c = & \int ds^+ + \int ds^- + R((\theta_0^+ - \theta_1^+) + (\theta_0^- - \theta_1^-)) \\ & + 2|r_0 - r_1| + 2R \int d\phi. \end{aligned} \quad (8)$$

Recall that the merge operation merges two edges adjacent to a degree-two node, and does not otherwise affect the topology of the shock graph. Hence, the decision whether or not to merge a set of adjacent edges is captured and integrated in the process of computing the deform cost.

#### 5.4 Edit Distance between Two Shapes

Let the shapes to be matched  $A$  and  $B$  be represented by their shock graphs  $T_A$  and  $T_B$ . For a shock graph  $T_A$ , let  $e_{Ai} \in E_A, i = 1 \dots N_A$  represent the set of edges. In the optimal edit sequence for matching  $A$  and  $B$ , let  $D_A$  denote the set of edges in  $T_A$  that corresponds with edges in  $T_B$ , and let  $S_A$  and  $C_A$  denote the set of edges that are spliced and contracted, respectively. Let  $e_{Bj} = \phi_{AB}(e_{Ai}), e_{Ai} \in D_A, e_{Bj} \in D_B$  denote the corresponding edge of  $e_{Ai}$ . The edit distance of  $A$  and  $B$  is given by

$$\begin{aligned} D(A, B) = & \sum_{e_{Ai} \in D_A, e_{Bj} = \phi_{AB}(e_{Ai})} \text{deformCost}(e_{Ai}, e_{Bj}) \\ & + \sum_{e_{Ai} \in S_A} \text{spliceCost}(e_{Ai}) + \sum_{e_{Ai} \in C_A} \text{contractCost}(e_{Ai}) \\ & + \sum_{e_{Bi} \in S_B} \text{spliceCost}(e_{Bi}) + \sum_{e_{Bi} \in C_B} \text{contractCost}(e_{Bi}). \end{aligned}$$

## 6 RESULTS OF SHOCK-GRAPH MATCHING

In this section, we present the results of our approach. First, we show that shock-graph matching gives intuitive correspondences for a variety of shapes, and is effective in the presence of commonly occurring visual transformations like articulation, shadows and highlights, viewpoint variation, scale changes, boundary perturbations, and partial occlusion. Then, recognition results for two databases consisting of 99 shapes and 216 shapes are shown. Finally, shock-graph



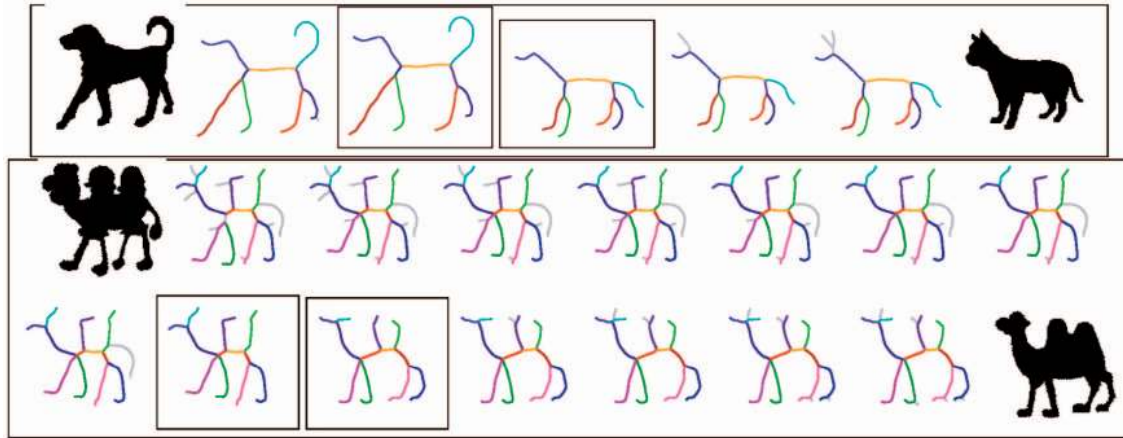


Fig. 23. The intermediate shock graphs in the optimal edit sequence for matching a cat and dog (top) and two camels (bottom). The boxed shock graphs have the same topology.



Fig. 24. **Boundary perturbations:** Shock matching in the presence of boundary noise where the same shape is represented by a coarser discretization. Note that the shock graphs are matched intuitively: All the edges corresponding to the boundary noise are pruned, while matching salient parts.



Fig. 25. **Articulation and deformation of parts:** Matching of the two poses of the “baby” is intuitive. Observe how the legs are matched correctly in spite of the self-occlusion that merges part of the legs (left) and how the arms are pruned (right).

matching is compared to shock-tree matching [31] and shape-contexts matching [4].

Fig. 22 shows that shock-graph matching results in an intuitive pairing of shock segments. In addition to giving the final correspondence, the edit distance algorithm gives a sequence of intermediate shock graphs that identify the optimal transformation of one input shock graph to another, see Fig. 23.

### 6.1 Robustness of Recognition under Visual Transformations

We now examine the robustness of the match under six classes of visual transformations. First, observe that shock computation is sensitive to **boundary perturbations**, which often introduce spurious shock edges and modify existing branches. While approaches relying on the medial axis have advocated either regularization in the detection process or after recovery to address such instabilities [28], one can alternatively view the *regularization as part of the recognition process*. This is illustrated in Fig. 24, where one of the shapes being matched experiences discretization artifacts. The current approach considers all deformation paths between the noisy shape and the noiseless one, and finds that the optimal path consists of edits which prune the spurious

branches. This is mainly because the cost of a single splice as well as the cumulative cost of a large sequence of splices is rather low as compared to large structural changes.

Second, we examine robustness of the match in the presence of **articulation and deformation of parts**. The shock graph of a shape inherently segments the shape into parts, and captures the hierarchical relationship between those parts. Thus, shock-graph matching implicitly involves matching the global hierarchy of parts in addition to matching the individual parts, making it robust to changes which may occur in some of the parts, see Fig. 25.

Third, the presence of **illumination variation** like shadows and highlights often leads to **segmentation errors** in figure-ground segregation. On the one hand, changes in the boundary caused by highlights tend to be small but, typically, affect the shock-graph topology. On the other hand, shadows are often more global in nature and tend not to affect the shock-graph topology. Fig. 26 shows that shock-graph matching is effective in the presence of a limited extent of segmentation errors.

Fourth, we examine the robustness of the proposed technique to **scale variations**. Observe that the shock-graph topology does not change when a global scaling transformation is applied, but the shock edge comparison metric is

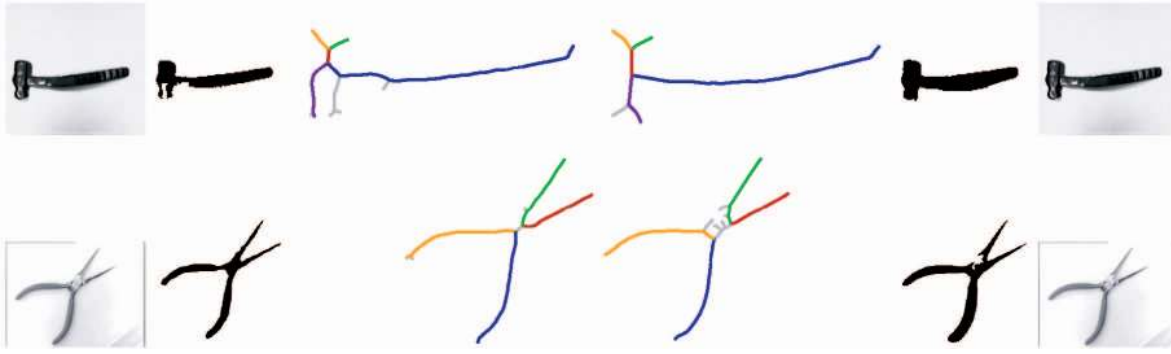


Fig. 26. **Illumination variation:** Shock graph matching gives intuitive correspondence in the presence of segmentation errors due to highlights (hammer in top left and pliers in bottom right) and shadows (hammer in top right). The original images shown on the extreme left and right are from the database of S. Sclaroff, Boston University, and segmented using a region growing technique with manually selected thresholds.



Fig. 27. **Scale variations:** Left: Shock graph matching of the hands gives intuitive correspondence in the presence of a scaling transform. The hand on the right was obtained by scaling the original hand by 225 percent. Right: The plot shows that the edit cost ( $y$ -axis) increases linearly as scaling is increased ( $x$ -axis).

not scale-invariant (as it should not be because scale-invariance requires consideration of the entire shape). However, the behavior of the metric with scale is theoretically well-understood: Outside a limited range of scales, the metric is linear in scale changes based on (5). Fig. 27 illustrates that shock-graph matching gives the intuitive correspondence in the presence of modest amounts of scaling. However, in the presence of large amounts of scaling, in this case more than three times, unintuitive correspondence *can* result if parts of unequal size are present, as the cost of matching equal-size but noncorresponding parts may become dominant.

Fifth, since 2D shapes are typically obtained from projections of 3D objects, robustness to **viewpoint variations** is critical to a 2D shape matching technique. When the viewpoint is varied gradually, the spatial location and the shape of parts typically changes gradually and such changes are handled by the deform edit. Exceptionally, however, at certain views, there are sudden changes that are handled by the remaining edits, e.g., appearance of a part is handled by the splice edit; a change in aspect is handled by the contract edit, etc., see Fig. 28. In general, a change in viewpoint constitutes a one-parameter family of deformations on shape, for which a well-defined neighborhood locally exists through the explicit embedding of transitions.

Sixth, **partial occlusion** is a serious challenge for any recognition framework. We consider two types of occlusion, one where the occluder blends with the shape, and one where the occluder blends with the background. In both types, shock-graph matching gives the intuitive correspondence by pruning out the occluded part, Fig. 29. However, it may fail if a rather large part of the shape is occluded, in which case the splice cost may dominate the total cost.

## 6.2 Indexing Results

We now examine recognition rates for indexing into two databases. The first database was created by assembling sample shapes from a wide spectrum of sources<sup>2</sup> to form nine categories: fish, rabbit, airplane, “greeble,” tool, hand, doll, four-legged animal, and sea-animal. We include 11 shapes in each category to allow for variations in form, as well as for occlusion, articulation, missing parts, etc., for a total of 99 shapes. Each shape is matched against all others, and results are ordered by edit-distance, Table 1. Recognition rates in percent by this measure are (100, 100, 100, 99, 99, 99, 97, 96, 95, 87); in other words, the top three choices are always correct for this database with a fairly slow dropoff in performance for the other choices.

We constructed a second database from samples of a very large database of shapes created for testing the compression rates for MPEG7, kindly provided by Latecki et al. [24]. Our selection from this database consists of 18 categories with 12 shapes in each category, Table 2. Recognition rates in percent are (100, 100, 100, 99, 97, 99, 96, 96, 95, 91, 80); in other words, the top three choices are always correct for this database, and again followed by a slow drop off.

## 6.3 Comparison to Other Approaches

In this section, we compare our approach to the approaches based on matching Pelillo et al.’s rooted shock tree [31] and shape context [4]. We have repeated the indexing experiment reported in [31] using our approach and shape-context matching [4]. The database used in [31] consists of 25 shapes, which can be organized into eight categories, Table 3: brush

<sup>2</sup> “Greebles” are obtained from M. Tarr’s collection, fish and sea-animals from F. Mokhtarian’s database, and tools from S. Sclaroff’s collection.

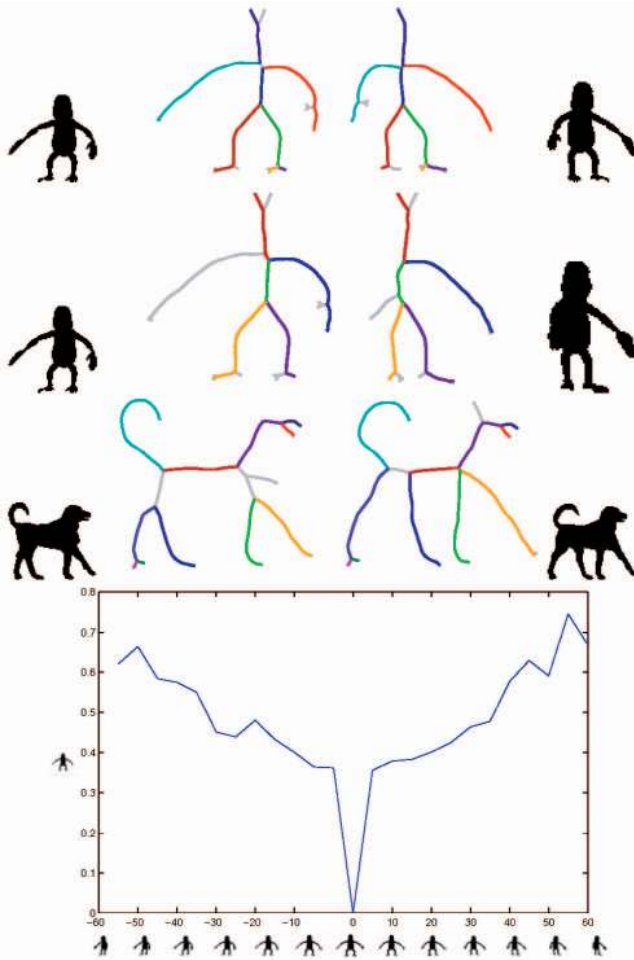


Fig. 28. **Viewpoint variation:** Shock graph matching gives the intuitive correspondence when a change in viewpoint results in 1) the disappearance of a part (left arm of model in second row) and 2) a change in aspect (hind legs of retriever in third row). The plot shows that the normalized edit cost ( $y$ -axis) increases monotonically with viewpoint variation for a local neighborhood. This property has been used for 3D object recognition from 2D views [11].

(1, 2, 3), hammer (4, 5, 6), pliers (7, 8, 9, 10, 11, 12), screwdriver (13, 14, 15), wrench (16, 17), hand (18, 19, 20), profile (21, 22, 23), horse (24, 25). Four categories, namely, hammer, plier, screwdriver, and wrench group into a more abstract superordinate category of "tools." In addition, just based on intuitive shape comparison, note the similarities between the brush and screwdriver categories. The remaining categories are quite distinct; the only possible connection is that both hands and horses have four elongated sequentially placed "limbs."

With this in mind, two types of performance measures are used to compare the results. First, we ask how many of the within category shapes are correctly recalled among the top matches. We highlight correct within-category matches using a yellow surround, and erroneous within-category matches using a red surround. In Pelillo et al.'s rooted shock tree matching, shapes 15, 17, and 21 give rise to a total of five errors. In shape-context matching, shapes 1, 3, 7, 8, 9, 10, 11, 12, 13, 14, and 15 give rise to a total of 21 errors. In the edit-distance approach, shapes 8, 13, 14, 15 give rise to a total of five errors. Clearly, both shock-based approaches perform significantly better than shape-context matching.

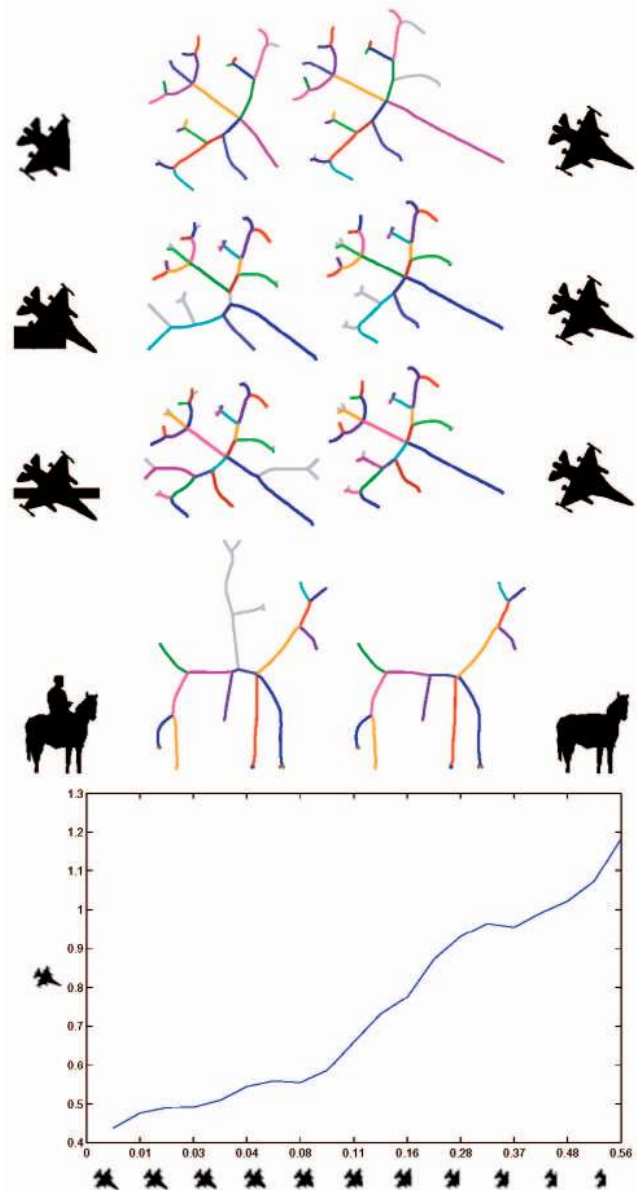


Fig. 29. **Partial occlusion:** Shock graph matching gives intuitive correspondence in the presence of partial occlusion. Observe in particular how the optimal edit sequence between a horse and a rider with the horse by itself involves pruning the shock edges corresponding to the rider (fourth row). The plot shows that normalized edit cost ( $y$ -axis) increases as the fraction of the plane occluded ( $x$ -axis) increases.

A more careful examination of the differences between the two shock-based methods, however, reveals that, in the five errors of Pelillo et al.'s rooted shock-tree matching, the erroneously recalled shapes are in fact perceptually very dissimilar: fat screwdriver (shape 15) is matched to the hands, wrench (shape 17) with the hammer, and profile (shape 21) with hands. In contrast, in three out of five errors of edit-distance shock-graph matching the top-ranking shapes match the query shape very well (depicting shape near category boundary): fat screwdriver (shape 15) with two brushes and screwdriver (shape 13) with brush. In the fourth error, the match between shape 14 and shape 7, the screwdriver is considered a part of the shape 7 (a single ply) with occlusion. The other remaining error (shape 8), however, has a dissimilar shape as a match. Thus, the within-category errors in Pelillo et al.'s rooted shock-tree matching have, in general, dissimilar



TABLE 1

										550	551	560	567	572	589	593	613	616	678						809	812	828	836	838
										350	573	581	600	616	618	646	655	720	770						793	824	860	860	869
										739	748	753	756	756	777	788	811	812	836						932	932	933	937	946
										322	507	572	574	578	589	649	649	704	911						939	942	955	956	957
										209	255	265	268	268	273	276	289	299	334						650	679	697	714	714
										535	556	558	614	628	637	646	652	685	693						702	702	714	720	738
										300	600	607	617	622	628	634	637	641	642						643	643	649	650	654

Left: A database of 99 shapes. Right: 15 nearest neighbors for a few representative shapes ordered by their normalized edit distance. Note that edit distances are multiplied by 1,000 for clarity of presentation. As there are 11 shapes in each category, up to 10 nearest neighbors can be from the same category. The next five matches are shown for completeness. Observe that, in most cases, the top 10 matches are from the same category. In fact, the top three matches are always correct and the only instance where the fourth and the fifth matches are wrong is shown in the last row.

TABLE 2

	426	427	464	472	473	510	511	522	527	544	553	612	616	625	631
	398	485	520	527	528	545	558	574	575	591	616	768	768	777	778
	449	457	477	517	519	521	531	536	562	587	600	619	647	658	658
	705	706	715	721	725	752	760	766	772	800	803	814	814	820	824
	584	614	652	653	661	663	696	699	713	736	768	771	808	817	823
	303	319	320	323	354	360	363	369	377	380	382	604	609	615	620
	186	188	213	277	313	335	338	406	456	477	492	528	557	557	560
	519	530	554	599	610	630	651	665	682	692	702	732	745	760	766
	537	546	558	565	568	592	622	623	635	647	656	667	676	678	693

Left: The database of 216 shapes selected from the MPEG-7 test database [24]. Right: 15 nearest neighbors for a few representative shapes. As there are 12 shapes in each category, up to 11 nearest neighbors can be from the same category. Observe that, in most cases, the top 11 matches are from the same category.

shapes, while in edit-distance shock graph matching, the erroneous recalls are in fact correct in picking similar shapes, which are nevertheless cognitively categorized differently.

A second type of performance measure involves counting the number of unintuitive matches using the superordinate categorization described at the beginning of this section. We have highlighted such matches using a green surround. In Pelillo et al.'s rooted shock tree matching shapes 1, 2, 4, 7-15, 18, 19, 20, 21, 24, 25 give rise to a total of 33 unintuitive matches. For example, the top matches of a brush (shape 2) includes a profile (3rd match), and hands (matches 4-6) where

more similar shapes are available. Similarly, for the screwdrivers (shapes 13, 14, 15) the top matches include the hands. On the other hand, there are three unintuitive matches using shape-context matching for shape 24, and 7 unintuitive matches for shapes 18, 19, 20, and 24 using edit-distance shock-graph matching. This second performance measure, while somewhat subjective (in determining the category structure), is a clear qualitative indicator that the edit-distance shock-graph matching recalls similar shapes better than both Peillo et al.'s rooted shock tree matching and shape-context matching for this database.



TABLE 3  
Comparison of Shock Tree Matching Using Association Graphs [31], Shape-Contexts Matching [4],  
and Shock Graph Edit Distance Using the Database Used in [31]

Shock Tree [31]								Shape Contexts [4]								Shock Graph Edit							
	1	2	3	4	5	6	7		1	2	3	4	5	6	7		1	2	3	4	5	6	7
1								1								1							
2								2								2							
3								3								3							
4								4								4							
5								5								5							
6								6								6							
7								7								7							
8								8								8							
9								9								9							
10								10								10							
11								11								11							
12								12								12							
13								13								13							
14								14								14							
15								15								15							
16								16								16							
17								17								17							
18								18								18							
19								19								19							
20								20								20							
21								21								21							
22								22								22							
23								23								23							
24								24								24							
25								25								25							

The correct within-category matches are highlighted using yellow surround, erroneous within-category matches using red, and unintuitive between-category matches using green. Observe the sensitivity of shape contexts to articulation (pliers: rows 7-12, middle table).

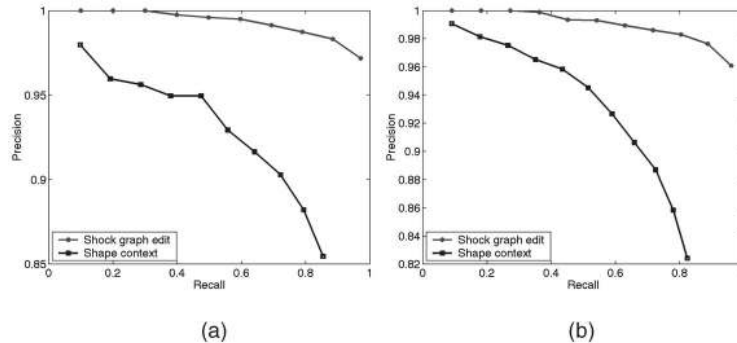
















































































Fig. 30. The precision-recall diagrams for indexing into the database of (a) 99 shapes and (b) 216 shapes. Note that both methods perform well with more than 80 percent precision.

We have also compared our approach to shape-context matching [4] using our databases of 99 and 216 shapes.<sup>3</sup> The results are summarized using precision-recall diagrams, see

3. We have not applied Siddiqi et al.'s rooted shock-tree matching [42] and Pelillo et al.'s [31] to these databases, because currently their code is not publicly available.

Fig. 30. Precision is the ratio of the number of similar shapes retrieved to the total number of shapes retrieved, while recall is the ratio of the number of similar shapes retrieved to the total number of similar shapes in the database [26]. It is clear from these diagrams that shock graph edit-distance outperforms shape-context matching for these databases.

TABLE 4  
This Table Shows the Sensitivity of  
Shape Contexts to Partial Occlusion

Shapes used						
Shape Context [4]		1	2	3	4	5
						
						
						
						
						
						
Shock Graph Edit		1	2	3	4	5
						
						
						
						
						
						

In particular, observe how the occlusion of the thumb results in the hand being matched to an animal (rows 4-6, left table).















































































Finally, we examine the performance of our approach and shape-contexts matching in the presence of partial occlusion and part-based changes like articulation. We have selected a small dataset of six shapes (three hands and three animals) to examine the performance in the presence of partial occlusion, Table 4. Occlusion affects a portion of a shape, and changes in the shock graph representation are limited to the affected portions. On the other hand, shape contexts is global in that all points are used to compute a summary signature for each point. This difference results in shock graph matching outperforming the shape contexts matching, Table 4. Shock graphs inherently partitions the shape and captures the hierarchical relationship among the parts. This allows it to be robust to part-based changes like articulation, see Table 5.

## 7 DISCUSSION AND CONCLUSIONS

We have presented a novel recognition framework based on considering different deformation paths of shock graphs of 2D shapes which brings them into correspondence, and finding the optimal one. Our approach to finding the optimal deformation path relies on a formal classification of shock transitions which is used to discretize the shape space and deformation paths. We employ a graph edit distance algorithm to find the optimal deformation sequence. The cost of the optimal sequence gives a metric of dissimilarity which is then used for indexing into databases based on shape. The effectiveness of the approach in the context of a variety of visual transformations is demonstrated and indexing into two separate databases of 99 and 216 shapes results in 100 percent accuracy for the top three matches. Note that this approach measures distance between shapes in a *dynamic* fashion in that intermediate representations are explicitly considered [25], [52] during the process of finding the least-cost deformation path. This is in contrast to the *static* approach where two shape representations are directly compared [13], [33], [2], [4], [31], [39], [42] to determine the distance.

A question which naturally arises is whether the introduction of an additional “medial” machinery can be justified as compared to point-based and curve-based approaches. First, observe that, in the medial/shock graph representation, there

TABLE 5  
This Table Illustrates the Sensitivity of  
Shape Contexts to Articulation

Shapes used						
Shape Context [4]		1	2	3	4	5
						
						
						
						
						
						
Shock Graph Edit		1	2	3	4	5
						
						
						
						
						
						

Observe how shape context matches the pliers to sticks in the same relative pose, and not to other pliers.

is an increased level of “organization” as compared to curve or point-based representations. In point-based representations, a point is identified by its local or global context. In curves, the point’s identity is established through the ordered 1D neighborhood. In shock-graph (or medial) representations, the point’s identity is established through a *pair* of joint 1D ordered neighborhoods that are captured via the medial segment. This additional level of organization in going from points to shocks can greatly constrain a match as highlighted by matching icons in Fig. 31, and thus, avoid potential mismatches and preserve the *coherence of shapes* in the match process.

Another difference between point-based approaches [4], [9], and both curve-based and medial-based approaches is that the arrangement and spatial layout of local features is necessarily captured via embedding it in an *extrinsic* framework for the former, while it can be captured *intrinsically* in the latter. That a point’s identity is measured in terms of its signature in an external coordinate system with respect to other points is precisely what makes it sensitive to shape deformations which are small in “intrinsic” deformation energy, but which are large in extrinsic deformation energy, e.g., bending, articulation, etc. In other words, instead of measuring the *relative* spatial arrangements of local features, as can be done in curve and medial representations, extrinsic methods capture the absolute spatial arrangement. This difference is what makes the point-based methods sensitive to object-based changes like articulation and occlusion, and to global rotations, as illustrated in Section 6.3.

There are a few areas for future improvement. The current approach to computing the edit costs is not scale-invariant. The individual edit costs themselves should not be scale-invariant since the determination of scale requires the consideration of the entire shape. However, the overall metric needs to be scale-invariant. We plan to examine the match at several scales and determine the global scale by selecting the one that minimizes the match cost. Since the edit costs are linear outside a limited range of scales, only a few scales need to be examined [33]. A second direction involves modifying how the matching algorithm penalizes the deletion of a portion of a shape that consists of several parts. Currently, such a deletion is penalized by the sum of the costs of the



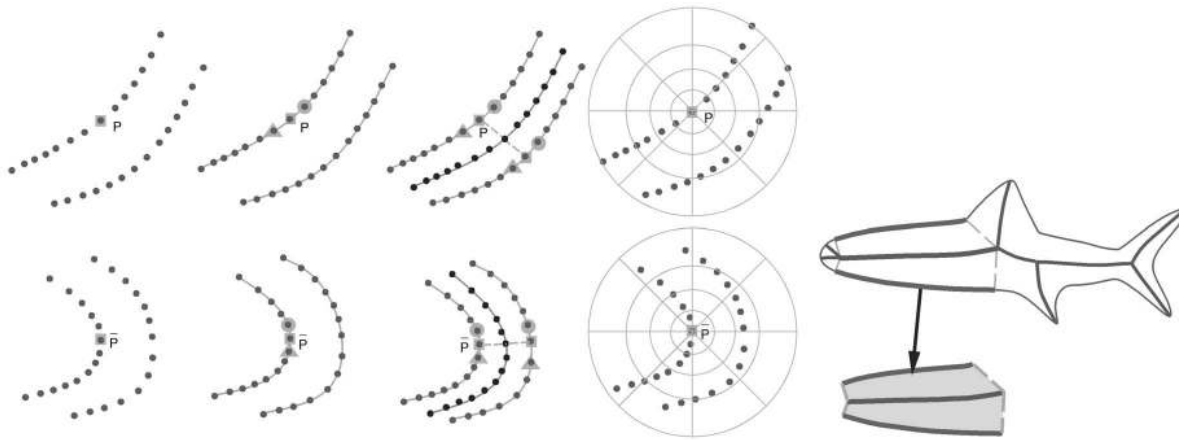


Fig. 31. Left: A schematic comparison of shape representation using unorganized points, outline curve, and shock graph. The point-based representation uses a local or global similarity measure. The curve representation induces an ordering of points. The shock graph representation introduces an additional constraint by pairing two boundary segments (shown by dotted lines) to represent the interior of the shape. The main point is that the extent of organization constrains the match. For example, in matching  $P$  to  $\bar{P}$  using the curve-based representation, neighboring points to  $P$  and  $\bar{P}$  in the boundary need to be matched as well, while using the shock-based representation, this also extends to neighboring points on an associated pair of boundaries. Similarly, the shape context forces a global context to be matched, but this is too rigid to accept articulation and occlusion (which is fine for some cases, e.g., font recognition). In contrast, the medial context is local with respect to the shape. Right: Each shock segment corresponds to two boundary segments, and represents a shape fragment (shaded).

individual parts being deleted, which can lead to unintuitive matches in cases where a rather large portion of the shape is occluded. In future, we plan to incorporate a more explicit notion of parts to address this issue.

Our initial shape comparison results are rather promising and support further exploration of this framework. We aim to explore recognition rates on a much larger database, as we believe that it is in the context of a much larger space of shapes that the effectiveness of this algorithm can truly be studied. However, two developments are needed. First, the algorithm for matching shapes typically takes about 3-5 minutes on an SGI Indigo II (195 MHz) for the examples presented here, which limits the number of shapes that can be practically matched. We plan to improve the computational efficiency by revising and optimizing the “development” code, and also by revising the algorithm itself. We are also developing a coarse-to-fine matching strategy, notion of exemplars to represent a shape category [38], and a nearest neighbor search method [35], which can be integrated in the indexing scheme to avoid unnecessary matches, thus reducing the computational burden.

## APPENDIX A

### ALGORITHM TO COMPUTE THE ALIGNMENT CURVE

This section briefly reviews how the optimal alignment curve of two curves  $\mathcal{C}$  and  $\hat{\mathcal{C}}$  is computed by minimizing the cost functional in (4) using dynamic-programming. For details, see [33], [34], [36]. The alignment curve is a curve in the 2D plane whose axes are specified by  $\mathcal{C}$  and  $\hat{\mathcal{C}}$ . We discretize  $\mathcal{C}$  and  $\hat{\mathcal{C}}$  by samples  $s_1, s_2, \dots, s_n$  and  $\hat{s}_1, \hat{s}_2, \dots, \hat{s}_m$ , respectively. Let  $\mathcal{C}|_{[s_i, s_j]}$  be the subsegment of curve  $\mathcal{C}$  with  $s_i$  and  $s_j$  as endpoints. Let  $d(i, j)$  denote the cost of matching the segments  $\mathcal{C}|_{[s_1, s_i]}$  and  $\hat{\mathcal{C}}|_{[\hat{s}_1, \hat{s}_i]}$ , and let  $\delta([k, i], [l, j])$  be the cost of matching the segments  $\mathcal{C}|_{[s_k, s_i]}$  and  $\hat{\mathcal{C}}|_{[\hat{s}_l, \hat{s}_i]}$ . As distance is defined in terms of a functional, it satisfies the following rule

$$d(i, j) = \min_{k, l} [d(i - k, j - l) + \delta([i - k, i], [j - l, j])], \quad (9)$$

which gives a recipe for computing  $d(\mathcal{C}, \hat{\mathcal{C}})$  via dynamic programming. The dynamic-programming cost table is sequentially updated and the optimal alignment curve is found by tracing through the cost table. We limit the choices of the  $k$  and  $l$ , as a first approximation, to nine values achieved by using the template shown in Fig. 32.

## APPENDIX B

### EDIT-DISTANCE ALGORITHM AND COMPUTATIONAL COMPLEXITY

This section reviews the algorithm for finding the edit-distance between two unrooted trees, which is used to find the globally optimal sequence of transitions between two shock graphs in polynomial time [22], [23]. We first describe the case of two rooted trees  $T_1$  and  $T_2$ ; the case of unrooted trees is discussed later. Given a rooted tree  $T$ , each edge  $\{x, y\}$  of  $T$  connecting nodes  $x$  and  $y$  can be represented by two oppositely directed arcs  $(x, y)$  and  $(y, x)$ , called *darts*. The *Euler tour*  $E(T)$  traverses the tree and is a cycle consisting of all the darts. (An example is shown in Fig. 33.) An *Euler string* is a consecutive subsequence of the dart-sequence which represents the Euler tour. Observe that a portion of a tree  $T_i$  can be specified by the Euler string  $s_i$ , and is denoted by the pair  $H_i = (T_i, s_i)$ . A *subproblem* is specified by the pair  $(H_1, H_2) = ((T_1, s_1), (T_2, s_2))$ . Dynamic programming is used to solve all the subproblems, i.e., compute the tree edit-distance between the portions of the trees specified by the Euler strings  $s_1$  and  $s_2$ . Note that the edit-distance for each subproblem is, in turn, computed from the edit distance for a subproblem having shorter strings (i.e., the sum of the lengths of the two strings is smaller). Finally, the edit-distance between two trees is computed as a subproblem where the Euler strings encompass the trees.

First, we sketch how a subproblem is solved in the absence of *splice* and *merge* operations. This method is due to Zhang and Shasha [51], though the formulation presented here is different. To compute the edit distance for  $((T_1, s_1), (T_2, s_2))$ , consider the rightmost darts  $d_1$  of  $s_1$  and  $d_2$  of  $s_2$ . In the

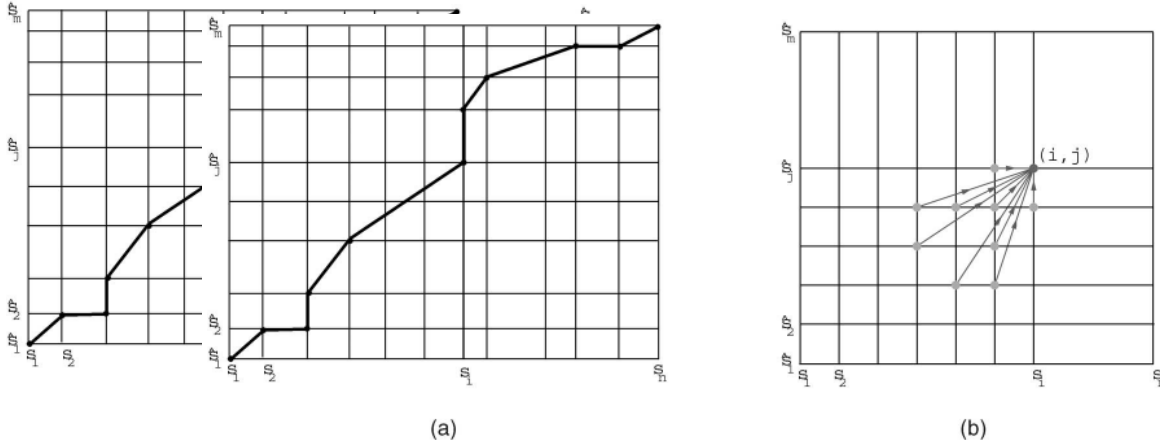


Fig. 32. (a) The grid used by the dynamic-programming algorithm to compute the optimal alignment curve. Discrete samples along the curves specify the axes. A sample alignment curve is shown in bold. (b) The template that is used to limit the choices when updating the cost at location  $(i, j)$  in the dynamic-programming cost table.

optimal edit sequence, there are only three possibilities concerning these darts: 1) the edge  $e_1$  corresponding to  $d_1$  gets contracted, 2) the edge  $e_2$  corresponding to  $d_2$  is similarly contracted, and 3) the two edges  $e_1$  and  $e_2$  are matched together. Using this observation, the edit-distance for  $((T_1, s_1), (T_2, s_2))$  can be expressed as the minimum of three quantities, each involving the edit distance for a subproblem with smaller substrings, one for each of the three Cases 1, 2, and 3. For  $i = 1, 2$ , write  $s_i$  as  $s'_i d_i$  (see Fig. 34.) The quantity for Case 1 is the cost of contracting  $e_1$  plus the edit-distance for  $((T_1, s'_1), (T_2, s_2))$ . The quantity for Case 2 is the cost of contracting the edge corresponding to the rightmost dart in  $s_2$  plus the edit-distance for  $((T_1, s_1), (T_2, s'_2))$ . The quantity for Case 3 is the sum of three parts. For  $i = 1, 2$ , write  $s_i$  as  $s_i^{\text{upper}} d'_i s_i^{\text{lower}} d_i$ , where  $d'_i$  and  $d_i$  are the darts corresponding to  $e_i$ . (See Fig. 35.) The three parts are:

1. the cost of matching the edge of  $T_1$  corresponding to the rightmost dart of  $s_1$  against the edge of  $T_2$  corresponding to the rightmost dart of  $T_2$ ,
2. the edit-distance for  $((T_1, s_1^{\text{lower}}), (T_2, s_2^{\text{lower}}))$ , and
3. the edit-distance for  $((T_1, s_1^{\text{upper}}), (T_2, s_2^{\text{upper}}))$ .

Thus, the recurrence relation is

$$\begin{aligned}
 D((T_1, s_1), (T_2, s_2)) &= \min \{ \text{contract-cost}(e_1) + D((T_1, s'_1), (T_2, s_2)), \\
 &\quad \text{contract-cost}(e_2) + D((T_1, s_1), (T_2, s'_2)), \\
 &\quad \text{match-cost}(e_1, e_2) + D((T_1, s_1^{\text{lower}}), (T_2, s_2^{\text{lower}})) \\
 &\quad + D((T_1, s_1^{\text{upper}}), (T_2, s_2^{\text{upper}})) \}.
 \end{aligned}$$

Next, we discuss how the above approach can be adapted to handle *merge* and *splice*. This algorithm is due to Klein et al. [23]. Note that repeated use of *merge* or *splice*

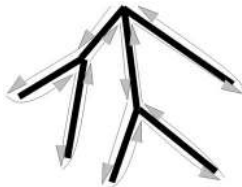


Fig. 33. The dashed arrows show an Euler tour of a tree.

operations can combine a path of edges into a single edge. Hence, we assume here (and discussed in Section 5), a method for assigning a match cost not to a pair of edges but to a pair of paths.

In computing the edit distance for  $((T_1, s_1), (T_2, s_2))$ , there are now many additional cases: For each dart  $a_1$  in  $s_1$  that is a descendent in  $T_1$  of the last dart in  $s_1$  and for each dart  $a_2$  in  $s_2$  that is a descendent in  $T_2$  of the last dart in  $s_2$ , there is a quantity: The cost of matching the path in  $T_1$  from  $a_1$  to the last dart in  $s_1$  against the path in  $T_2$  from  $a_2$  to the last dart in  $s_2$ , plus the edit-distance of  $((T_1, a_1), (T_2, a_2))$ . In this sketch of the algorithm, we have omitted some technical details arising in handling *merge* and *splice*.

Thus far, we have discussed how to compute the edit distance between two rooted trees. This algorithm finds not the best match but the best match subject to the constraint that the two roots are matched to each other (and hence, that these roots are not removed by splice operations). To remove this constraint, we must modify the above algorithm. A straightforward way of doing this is to run the rooted algorithm once for each possible choice of roots, then take the best match among all of those obtained. This approach is guaranteed to find the optimal solution.

A heuristic that yields a faster algorithm but one that is not guaranteed to find the optimal solution is as follows. Choose a small set of candidate roots for  $T_2$  and run the rooted algorithm for every possible root for  $T_1$  and only the candidate roots for  $T_2$ . This approach finds the best match subject to the constraint that not all of the candidate roots is eliminated by splices. In order to make it likely that the optimal solution

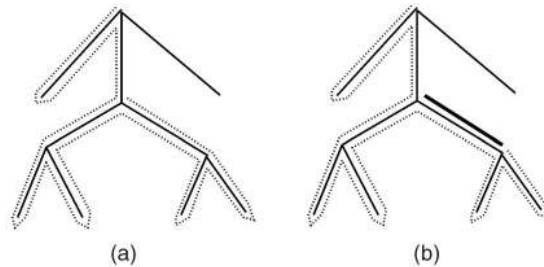


Fig. 34. (a) An Euler string  $s$  (shown dashed). (b)  $s$  is written as  $s'd$  where  $s'$  is a substring (shown dashed) and  $d$  is the rightmost dart (shown solid).



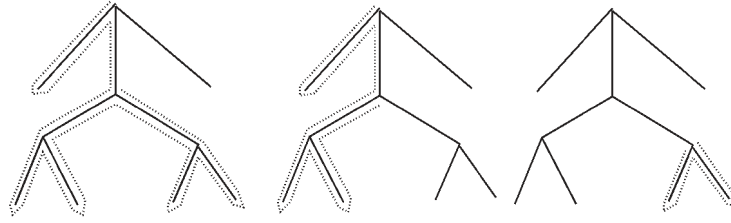


Fig. 35. Left: An Euler string  $s$  (shown dashed). Middle and right:  $s$  is written as  $s^{\text{upper}} d' s^{\text{lower}} d$ , where  $s^{\text{upper}}$  is the upper substring (middle) and  $s^{\text{lower}}$  is the lower substring (right).

satisfies this constraint, the set of candidate roots should be chosen to contain nodes whose elimination is expensive. In the experiments described in this paper, we chose a singleton set consisting of that degree-three node  $r$  with outgoing darts  $g_1, g_2, g_3$  so as to maximize  $\hat{c} = \min_j \text{splice cost}(g_i)$ . This guarantees that the prevented splices will cost at least  $\hat{c}$ . When using the heuristic, we can obtain an even faster algorithm by applying a technique due to Zhang and Shasha. The details of this technique are beyond the scope of this paper.

Now, we turn to time complexity analysis. We sketch a straightforward analysis of the nonheuristic algorithm (i.e., all roots of  $T_1$  and  $T_2$  are considered). For  $i = 1, 2$ , let  $n_i$  be the number of nodes in  $T_i$ . Then, the number of subproblems is  $O(n_1^2 n_2^2)$ . For each subproblem  $((T_1, s_1), (T_2, s_2))$ , the number of quantities involved is the number of darts in  $s_1$  that are descendants of the rightmost dart of  $s_1$ , times the number of darts in  $s_2$  that are descendants of the rightmost dart of  $s_2$ . This product is  $O(n_1 n_2)$ , so the total time is  $O(n_1^3 n_2^3)$ .

The heuristic method (using one candidate root for  $T_2$ ) can be shown to require time  $O(n_1^2 \text{diameter}(T_1) n_2^2 \text{collapsedDepth}(T_2) \text{depth}(T_2))$ , where the diameter  $\text{diameter}(T_1)$  is the maximum number of edges on a path in  $T_1$ , the depth  $\text{depth}(T_2)$  is the maximum number of edges on a path in  $T_2$  that starts from the root, and  $\text{collapsedDepth}(T_2)$  is a quantity defined by Zhang and Shasha, and shown to be no more than the depth of  $T_2$ . If each of the two trees has no more than  $n$  nodes and has diameter no more than  $k$ , the time is  $O(n^3 k^3)$ . The analysis of the heuristic method is beyond the scope of this paper.

## ACKNOWLEDGMENTS

P.N. Klein would like to acknowledge the support of US National Science Foundation grant CCR-9700146. B.B. Kimia would like to acknowledge the support of US National Science Foundation grants IRI-9700497 and IRI-0083231. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the US National Science Foundation.

## REFERENCES

- [1] N. Ayache and O. Faugeras, "HYPER: A New Approach for the Recognition and Positioning of Two-Dimensional Objects," *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 8, no. 1, pp. 44-54, 1986.
- [2] R. Basri, L. Costa, D. Geiger, and D. Jacobs, "Determining the Similarity of Deformable Shapes," *Vision Research*, vol. 38, pp. 2365-2385, 1998.
- [3] S. Belongie, J. Puzhicha, and J. Malik, "Shape Matching and Object Recognition Using Shape Contexts," *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 24, no. 4, pp. 509-522, Apr. 2002.
- [4] S. Belongie and J. Malik, "Matching with Shape Contexts," *Proc. IEEE Workshop Content-based Access of Image and Video Libraries*, 2000.
- [5] H. Blum, "Biological Shape and Visual Science," *J. Theoretical Biology*, vol. 38, pp. 205-287, 1973.
- [6] J. Bruce, P. Giblin, and C. Gibson, "Symmetry Sets," *Proc. Royal Soc. Edinburgh*, vol. 101A, pp. 163-186, 1985.
- [7] H. Bunke, "On a Relation between Graph Edit Distance and Maximum Common Subgraph," *Pattern Recognition Letters*, vol. 18, no. 8, pp. 689-694, Aug. 1997.
- [8] H. Bunke and K. Shearer, "A Graph Distance Metric Based on the Maximal Common Subgraph," *Pattern Recognition Letters*, vol. 19, nos. 3-4, pp. 255-259, 1998.
- [9] H. Chui and A. Rangarajan, "A New Algorithm for Non-Rigid Point Matching," *Proc. IEEE Conf. Computer Vision and Pattern Recognition*, vol. II, pp. 44-51, 2000.
- [10] I. Cohen, N. Ayache, and P. Sulger, "Tracking Points on Deformable Objects Using Curvature Information," *Proc. European Conf. Computer Vision*, pp. 458-466, 1992.
- [11] C.M. Cyr and B.B. Kimia, "3D Object Recognition Using Shape Similarity-Based Aspect Graph," *Proc. Int'l Conf. Computer Vision*, pp. 254-261, 2001.
- [12] W. Dewaard, "An Optimized Minimal Edit Distance for Hand-Written Word Recognition," *Pattern Recognition Letters*, vol. 16, no. 10, pp. 1091-1096, 1995.
- [13] Y. Gdalyahu and D. Weinshall, "Flexible Syntactic Matching of Curves and Its Application to Automatic Hierarchical Classification of Silhouettes," *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 21, no. 12, pp. 1312-1328, Dec. 1999.
- [14] P.J. Giblin and B.B. Kimia, "On the Intrinsic Reconstruction of Shape from Its Symmetries," *Proc. IEEE Conf. Computer Vision and Pattern Recognition*, pp. 79-84, 1999.
- [15] P.J. Giblin and B.B. Kimia, "On the Local Form and Transitions of Symmetry Sets, and Medial Axes, and Shocks in 2D," *Proc. Int'l Conf. Computer Vision*, pp. 385-391, 1999.
- [16] P.J. Giblin and B.B. Kimia, "Transitions of the 3D Medial Axis under a One-Parameter Family of Deformations," *Proc. European Conf. Computer Vision*, pp. 718-724, 2002.
- [17] S. Gold and A. Rangarajan, "A Graduated Assignment Algorithm for Graph Matching," *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 18, no. 4, pp. 377-388, Apr. 1996.
- [18] D.G. Kendall, D. Barden, T.K. Carne, and H. Le, *Shape and Shape Theory*. Chichester, UK: John Wiley and Sons, Inc., 1999.
- [19] B.B. Kimia, "Conservation Laws and a Theory of Shape," PhD dissertation, McGill Center for Intelligent Machines, McGill Univ., Montreal, Canada, 1990.
- [20] B.B. Kimia, J. Chan, D. Bertrand, S. Coe, Z. Roadhouse, and H. Tek, "A Shock-Based Approach for Indexing of Image Databases Using Shape," *SPIE: Multimedia Storage and Archiving Systems II*, vol. 3229, pp. 288-302, 1997.
- [21] P. Klein, "Computing the Edit Distance between Unrooted Ordered Trees," *Proc. European Symp. Algorithms*, pp. 91-102, 1998.
- [22] P. Klein, T. Sebastian, and B. Kimia, "Shape Matching Using Edit-Distance: An Implementation," *ACM-SIAM Symp. Discrete Algorithms*, pp. 781-790, 2001.
- [23] P. Klein, S. Tirathapara, D. Sharvit, and B. Kimia, "A Tree-Edit Distance Algorithm for Comparing Simple, Closed Shapes," *Proc. ACM-SIAM Symp. Discrete Algorithms*, pp. 696-704, 2000.

- [24] L.J. Latecki, R. Lakamper, and U. Eckhardt, "Shape Descriptors for Non-Rigid Shapes with a Single Closed Contour," *IEEE Conf. Computer Vision and Pattern Recognition*, pp. 424-429, 2000.
- [25] T. Liu and D. Geiger, "Approximate Tree Matching and Shape Similarity," *Proc. Int'l Conf. Computer Vision*, pp. 456-462, 1999.
- [26] E. Milios and E. Petrakis, "Shape Retrieval Based on Dynamic Programming," *IEEE Trans. Image Processing*, vol. 9, no. 1, pp. 141-146, 2000.
- [27] R. Myers, R. Wilson, and E. Hancock, "Bayesian Graph Edit Distance," *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 22, no. 6, pp. 628-635, June 2000.
- [28] R.L. Ogniewicz, *Discrete Voronoi Skeletons*. Hartung-Gorre, 1993.
- [29] P.J. Giblin and B.B. Kimia, "On the Intrinsic Reconstruction of Shape from Its Symmetries," *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 25, no. 7, pp. 895-911, July 2003.
- [30] M. Pelillo, "Matching Free Trees, Maximal Cliques, and Monotone Game Dynamics," *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 24, no. 11, pp. 1535-1541, Nov. 2002.
- [31] M. Pelillo, K. Siddiqi, and S. Zucker, "Matching Hierarchical Structures Using Association Graphs," *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 21, no. 11, pp. 1105-1120, Nov. 1999.
- [32] M. Pelillo, K. Siddiqi, and S.W. Zucker, "Many-to-Many Matching of Attributed Trees Using Association Graphs and Game Dynamics," *Proc. Int'l Workshop Visual Form*, pp. 583-593, 2001.
- [33] T.B. Sebastian, P.N. Klein, and B.B. Kimia, "On Aligning Curves," *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 25, no. 1, pp. 116-125, Jan. 2003.
- [34] T.B. Sebastian, J.J. Crisco, P.N. Klein, and B.B. Kimia, "Constructing 2D Curve Atlases," *Math. Methods in Biomedical Image Analysis*, pp. 70-77, 2000.
- [35] T.B. Sebastian and B.B. Kimia, "Metric-Based Shape Retrieval in Large Databases," *Proc. Int'l Conf. Pattern Recognition*, vol. 3, pp. 291-296, 2002.
- [36] T.B. Sebastian, P.N. Klein, and B.B. Kimia, "Alignment-Based Recognition of Shape Outlines" *Proc. Int'l Workshop Visual Form*, pp. 606-618, 2001.
- [37] T.B. Sebastian, P.N. Klein, and B.B. Kimia, "Recognition of Shapes by Editing Their Shock Graphs," *Proc. Int'l Conf. Computer Vision*, pp. 755-762, 2001.
- [38] T.B. Sebastian, P.N. Klein, and B.B. Kimia, "Shock-Based Indexing into Large Shape Databases," *Proc. European Conf. Computer Vision*, pp. 731-746, 2002.
- [39] D. Sharvit, J. Chan, H. Tek, and B.B. Kimia, "Symmetry-Based Indexing of Image Databases," *J. Visual Comm. and Image Representation*, vol. 9, no. 4, pp. 366-380, 1998.
- [40] K. Siddiqi, B. Kimia, A. Tannenbaum, and S. Zucker, "Shocks, Shapes, and Wiggles," *Image and Vision Computing*, vol. 17, nos. 5-6, pp. 365-373, 1999.
- [41] K. Siddiqi and B.B. Kimia, "A Shock Grammar for Recognition," *Proc. IEEE Conf. Computer Vision and Pattern Recognition*, pp. 507-513, 1996.
- [42] K. Siddiqi, A. Shokoufandeh, S. Dickinson, and S. Zucker, "Shock Graphs and Shape Matching," *Int'l J. Computer Vision*, vol. 35, no. 1, pp. 13-32, 1999.
- [43] H.D. Tagare, "Shape-Based Non-Rigid Correspondence with Application to Heart Motion Analysis," *IEEE Trans. Medical Imaging*, vol. 18, no. 7, pp. 570-578, 1999.
- [44] K.-C. Tai, "The Tree-to-Tree Correction Problem," *J. Assoc. Computing Machinery*, vol. 26, pp. 422-433, 1979.
- [45] H. Tek, "The Role of Symmetry Maps in Representing Objects in Images," PhD dissertation, Division of Eng., Brown Univ., Providence, R.I., July 1999.
- [46] H. Tek and B.B. Kimia, "Symmetry Maps of Free-Form Curve Segments via Wave Propagation," *Proc. Int'l Conf. Computer Vision*, pp. 362-369, 1999.
- [47] A. Torsello and E.R. Hancock, "Computing Approximate Tree Edit Distance Using Relaxation Labeling," *Pattern Recognition Letters*, pp. 1089-1097, 2003.
- [48] A. Torsello and E.R. Hancock, "A Skeletal Measure of 2D Shape Similarity," *Proc. Int'l Workshop Visual Form*, pp. 260-271, 2001.
- [49] R. Wagner and M. Fischer, "The String-to-String Correction Problem," *J. Assoc. Computing Machinery*, vol. 21, pp. 168-173, 1974.
- [50] W. Wallis, P. Shoubridge, M. Kraetz, and D. Ray, "Graph Distances Using Graph Union," *Pattern Recognition Letters*, vol. 22, no. 6-7, pp. 701-704, 2001.

- [51] K. Zhang and D. Sasha, "Simple Fast Algorithms for the Editing Distance between Trees and Related Problems," *SIAM J. Computing*, vol. 18, pp. 1245-1262, 1989.
- [52] S.C. Zhu and A.L. Yuille, "FORMS: A Flexible Object Recognition and Modeling System," *Int'l J. Computer Vision*, vol. 20, no. 3, pp. 187-212, 1996.
- [53] P.J. Giblin and B.B. Kimia, "On the Local Form and Transitions of Symmetry Sets, Medial Axes, and Shocks," *Int'l J. Computer Vision*, vol. 54, no. 1-3, pp. 143-157, 2003.
- [54] H. Tek and B.B. Kimia, "Symmetry Maps of Free-Form Curve Segments Via Wave Propagation," *Int'l J. Computer Vision*, vol. 54, no. 1-3, pp. 35-81, 2003.



include segmentation, shape-based analysis, and computer-aided detection. He has published papers in medical image segmentation, shape-based recognition, and retrieval. He is a member of the IEEE.



**Philip N. Klein** received the BA degree in applied mathematics from Harvard and the PhD degree in computer science from MIT. He is a professor of computer science at Brown University. His research interests include the analysis and development of algorithms, especially for problems involving graphs or discrete optimization. He is a member of the ACM.



**Benjamin B. Kimia** received the BEng Honors degree from McGill University, Montreal, Canada in 1982, followed by the MEng (1986) and the PhD (1991) degrees in the areas of computer vision and image processing. He is an associate professor in the Division of Engineering at Brown University. He is also the associate director of the Laboratory for Engineering Man/Machine Systems (LEMS), an interdisciplinary group focused on signal and image processing, control, multimedia, and computer engineering. Professor Kimia's current research interests are focused on mathematical, psychophysical, neurophysiological, and computational models for visual processing with applications to medical imaging, indexing into large image databases using shape, and digital archaeology. His research program is based on symmetry-based representations of 2D and 3D images for segmentation, recognition, categorization, registration, and visualization.

► For more information on this or any other computing topic, please visit our Digital Library at [www.computer.org/publications/dlib](http://www.computer.org/publications/dlib).