Shock Graphs and Shape Matching *

Kaleem Siddiqi[†] **Ali Shokoufandeh**[§] **Sven J. Dickinson**[§]

Steven W. Zucker[†]

[†]Center for Computational Vision & Control

Yale University, New Haven, CT 06520-8285

{siddiqi-kaleem,zucker-steven}@cs.yale.edu

[§]Department of Computer Science &

Center for Cognitive Science

Rutgers University, New Brunswick, NJ 08903

{shokoufa,sven}@cs.rutgers.edu

CVC TR-97-002/CS TR-1147/RU DCS-TR-345

October, 1997

*A condensed version of this paper appears in the Proceedings of the Sixth International Conference On Computer Vision, Bombay, Jan 4-7, 1998.

Abstract

We have been developing a theory for the generic representation of 2-D shape, where structural descriptions are derived from the shocks (singularities) of a curve evolution process, acting on bounding contours. We now apply the theory to the problem of shape matching. The shocks are organized into a directed, acyclic *shock graph*, and complexity is managed by attending to the most significant (central) shape components first. The space of all such graphs is highly structured and can be characterized by the rules of a *shock graph grammar*. The grammar permits a reduction of a shock graph to a unique rooted shock tree. We introduce a novel tree matching algorithm which finds the best set of corresponding nodes between two shock trees in polynomial time. Using a diverse database of shapes, we demonstrate our system's performance under articulation, occlusion, and changes in viewpoint.

Keywords: shape representation; shape matching; shock graph; shock graph grammar; subgraph isomorphism.

1 Introduction

Upon entering a room, one first notices the presence of a particular object, such as a dog, before realizing it is either a Siberian Husky or that it is "Loki", a particular Siberian. This example, modified from important studies by Rosch [43], suggests that there is an organization to our object memory, and that this organization facilitates recognition. Initially, particular instances are not recognized; rather, objects are first categorized generically at a "basic level of abstraction" [43]. The object is recognized as belonging to the category—dog—before more detailed, or subordinate levels, are refined. This motivating example is at the heart of this paper: we seek a technique for object recognition based on such entry-level, generic descriptions.

1.1 Classical Aspects of Shape Recognition

Rosch's experimental observation that basic-level descriptions precede particulars was made about the same time that Fu [15] and others were introducing syntactic pattern recognition. Fu's goal was to define a grammar for patterns, and then to specify automata that could recognize this grammar. However, this program lost favor because there was no clear indication of those pattern features on which the grammar should be based. Bounding and interior image curves were typical, but the graph matching rapidly became intractable. Missing and bogus edges were a problem, and the complexity of curve possibilities exceeded the grammars. With the exception of array grammars [44], little progress was made. Research continues on graph isomorphism algorithms for vision applications, but examples are still typically based on graphs derived from feature points and image curves [19]. Probability measures have also been placed on images and image curves in an attempt to provide a priori information suitable to guide matching [20].

Image curves are also at the heart of boundary-based descriptions, such as those of Hoffman and Richards [41], and alignment techniques, such as the one proposed by Ullman [5]. However, the Hoffman and Richards "codon" vocabulary is only an intermediate step toward more abstract part descriptions, and remains to be completed. It is an attempt to restrict the graph representation to the boundary, which eliminates contours that span several objects. In alignment schemes, the emphasis is not on boundary encoding, but on accounting for the differences between an observed and a stored shape. A clever algorithm by Ullman and Basri [56] interpolates from a linear combination of 2-D views, and impressive results on a Volkswagen image were reported. However, to achieve these results, the edge maps were manually edited so that only those appearing in all views were included [55]. In effect, this implies that edges are significant if and only if they appear in all views of an object which, of course, is impossible to achieve in general. Furthermore, no acceptable solution for automatically finding edge correspondences has been offered.

Considering the boundaries of objects implies a viewpoint dependency to shape recognition [8], but does not specify which features to use for each view. Aspect graphs were introduced by Koenderink and van Doorn [25] to enumerate topologically-distinct views [26]

via singular or catastrophic events. For example, when a cup is rotated, there is a particular viewpoint from which the handle just becomes visible; thereafter, only geometric variations take place until, at another singular viewpoint, the handle disappears. However, aspect graphs and other methods based on algebraic and differential invariants [13] were successfully defined only for specific classes of algebraic surfaces that fit only few (man-made) objects. The techniques are typically difficult to extend to natural objects.

Computer vision approaches to view-based modeling fall broadly into two classes. First, there are feature-based methods which represent each view as a collection of line segments, curves, corners, regions, *etc.* [22, 10, 11, 39]. The success of such methods depends largely on the extent to which the features are present and can be reliably extracted; once again they are not easily applied to natural objects. Second, a number of appearance-based methods have emerged which essentially treat the raw image as a single feature in a high-dimensional space [54, 34]. Whereas such techniques might succeed at recognizing particulars of a specific class, *e.g.*, faces, they cannot predict entry-level categories because there is no abstraction from image data to a model. Returning to the problem of database organization, such techniques would succeed at finding specific instances of Loki's body in a database of photographs of animals, but would fail at separating, for example, photographs of horses from photographs of hands.

In important contrast to the boundary based techniques discussed earlier was Blum's [6] medial axis transform—or skeleton—which preceded Rosch by about a decade. Blum's skeleton is area-based, and provides a description of shapes via the loci of centers of covering balls. The skeleton has the advantage of providing a different (from Fu et al.) type of graph on which to base matching, but again sensitivity causes problems. Proper skeletons can be found interactively, but not automatically, and as with the Fu and the Ullman approaches, the features have to be edited to provide a basis for matching. One option that is worth stressing is the use of hierarchical skeletons [35], because it attempts to capture a notion of "scale" for objects. This is important because, should such scales be available, coarse-to-fine matching strategies could be employed; see also Burbeck and Pizer [9].

In recent work, Sclaroff and Pentland have addressed the problem of 2-D shape matching

using a modal representation corresponding to a shape's generalized axes of symmetry [47]. This compact representation has been used for indexing [46], and offers a frequency-like (coarse to fine) decomposition of a shape. However, its global nature makes it sensitive to large occlusion. Zhu and Yuille have decomposed 2-D shapes into connected mid-grained skeletal parts, and have designed a matching system where similarity between parts is computed as a joint probability [57]. Whereas preliminary results are encouraging, several parameters have to be set, and there appears to be no hierarchy among the parts used for matching. Furthermore, the model "... was created to deal with animate objects and would have to be completely modified to deal with man-made objects like houses and industrial parts [57, p. 209]". Pauwels *et al.* have proposed the use of semi-differential invariants for planar shape recognition under affine distortions, with some robustness to occlusion [33, 38]. Basri *et al.* have proposed various models for measuring the cost of deforming one contour into another, while taking into account its part structure [4]. Gdalyahu and Weinshall have also proposed metric functions for measuring the similarity between two closed planar curves [18]. However, the previous three methods do not explicitly account for a shape's interior, which is key for determining more global properties such as symmetry. Finally, François and Medioni have proposed a connection hierarchy of parts for planar shape recognition [14], obtained from an axial decomposition introduced in [42].

Among the many applications of curve evolution to problems in computer vision and image processing, e.g., see [2, 45, 30], only a handful have addressed the problem of shape representation. Tari et. al have proposed a linear diffusion equation which can be used to build skeletal descriptions directly from greyscale images [52], and is computationally more efficient than those based on standard level set methods [36]. It leads to an approximation of the reaction diffusion space introduced in [23]. Tek et al. have used an orientation propagating distance function to extract symmetries from fragmented contours, labeling the resulting singularities according to whether or not the colliding waves carry "true" orientation information [53]. However, neither of the above efforts have explicitly addressed the problem of shape *recognition*, which is a key focus of this paper.

In summary, a substantial body of work on 2-D shape has contributed a positive set

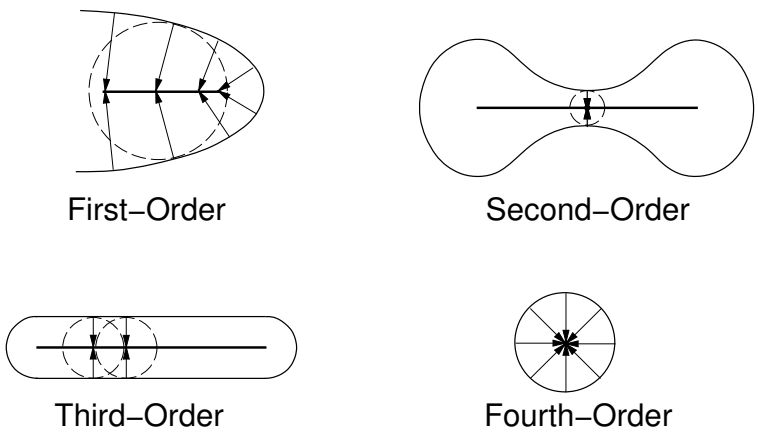

Figure 1: A coloring of shocks into four types. A 1-shock derives from a *protrusion*, and traces out a curve segment of 1-shocks. A 2-shock arises at a *neck*, and is immediately followed by two 1-shocks flowing away from it in opposite directions. 3-shocks correspond to an annihilation into curve segment due to a *bend*, and a 4-shock an annihilation into a point or a *seed*. The loci of these shocks gives Blum's medial axis.

of desiderata, although no technique exists that satisfies all of them. Thus, we seek a representation that is viewpoint dependent to start; that is generic in the sense that a notion of equivalence classes of (qualitatively similar) shapes emerges; that is applicable to natural as well as man-made objects; that is reliably and stably computable; that is capable of supporting efficient (e.g., polynomial-time) recognition in the presence of occlusion and noise, and that places special importance on certain boundary segments. We build our representation on the singularities of a curve evolution process, described next. We shall later abstract this representation into a graph that is particularly suited to efficient and generic shape matching.

1.2 Shapes and Shocks

Particular shapes can vary in detail from one another; variations between shapes derive from an organization of these particular shapes into equivalence classes. Thus certain discrete events are required to separate equivalence classes of continuous ones, and in mathematics such discrete events derive from singularity theory [3]. Kimia, Tannenbaum, and Zucker [23]

applied singularity theory to shape by exploring the consequences of slight boundary deformations. Specifically, for simple closed curves in the plane the following evolution equation was studied:

$$\begin{aligned} \mathcal{C}_t &= (1 + \alpha\kappa)\mathcal{N} \\ \mathcal{C}(s,0) &= \mathcal{C}_0(s). \end{aligned} \tag{1}$$

Here $\mathcal{C}(s,t)$ is the vector of curve coordinates, $\mathcal{N}(s,t)$ is the inward normal, s is the path parameter, and t is the evolutionary time of the deformation. The constant $\alpha \geq 0$ controls the regularizing effects of curvature κ. When α is large, the equation becomes a geometric heat equation; when $\alpha = 0$, the equation is equivalent to Blum's grassfire transformation [7, 23]. In this paper, we shall only be interested in the latter case, under which the evolution equation is hyperbolic and *shocks* [27], or entropy-satisfying singularities can form. Here we shall ignore the dynamics of the shock formation process, and will consider only the static picture obtained in the limit: the locus of shock positions gives Blum's medial axis. However, even in this static limit, the shocks provide additional information beyond that available from their loci: consider a "coloring" of the shocks according to the local variation of the radius function along the medial axis (see Figure 1). The colored description provides a much richer foundation for recognition than that obtained from an unlabeled (Blum) skeleton.

To illustrate the coloring, imagine traversing a path along the medial axis. At a 1-shock the radius function varies monotonically, as is the case for a protrusion. At a 2-shock the radius function achieves a strict local minimum such that the medial axis is disconnected when the shock is removed, e.g., at a neck. At a 3-shock the radius function is constant along an interval, e.g., for a bend with parallel sides.[1] Finally, at a 4-shock the radius function achieves a strict local maximum, as is the case when the evolving curve annihilates into a single point or a seed.

With the above picture in mind, the coloring can be formalized as follows. Let X be the open interior of a simple closed curve, and $Me(X)$ its medial axis (the set of points reached simultaneously by two or more fire fronts). Let $B(x,\epsilon)$ be an open disk of radius

[1] This "parallel" condition reflects the non-genericity of 3-shocks. Nevertheless, "bend"-like structures are abundant in the world; consider the legs of a chair, the fingers of a hand, or the tail of a dog. Numerical techniques have been developed to regularize such configurations into 3-shocks [51].

ϵ centered at $x \in X$, and let $R(x)$ denote the radius of the largest such disk contained in X. Let $N(x,\epsilon) = Me(X) \cap B(x,\epsilon) \backslash \{x\}$ define a "punctured" ϵ-neighborhood of x, one that does not contain x itself. A medial axis point $x \in Me(X)$ is

1. **type 4** if $\exists \epsilon > 0$ s.t. $R(x) > R(y)\ \forall y \in N(x,\epsilon)$;
2. **type 3** if $\exists \epsilon > 0$ s.t. $R(x) = R(y)\ \forall y \in N(x,\epsilon)$ and $N(x,\epsilon) \neq \emptyset$;
3. **type 2** if $\exists \epsilon > 0$ s.t. $R(x) < R(y)\ \forall y \in N(x,\epsilon)$ and $N(x,\epsilon) \neq \emptyset$ and $N(x,\epsilon)$ is not connected; and
4. **type 1** otherwise.

It should be clear that there is a relationship between the above coloring and the velocity function $\frac{dR}{dx}$ along the medial axis [48]. In separate work, we are investigating the Morse function properties of the signed distance function, and the Arnold classification of singularities [3]. In Figure 7 we provide numerical examples of colored medial axis descriptions. As we shall now show, the *coloring* coupled with a measure of *significance* derived from the time of shock formation, is the key to abstracting a representation that supports generic shape matching.

2 The Shock Graph

We shall now abstract the system of shocks derived from the curve evolution process into a graph, which we call the *shock graph*, or **SG**. This construction is inspired by Blum's classic work on axis-morphologies, in which he explored the use of directed graphs based on the medial axis for defining equivalence classes of objects [6]. The shock types will label each vertex in the graph and the shock formation times will direct edges to provide an ordering for matching, and a basis for subgraph approximation.

By the Jordan Curve Theorem, any simple closed curve divides the plane $\mathcal{R}^2$ into exactly two components, one bounded and the other unbounded. We are interested in the bounded interiors of Jordan curves.

Definition 1 *A 2-D shape $\mathcal{O}$ is the bounded interior of a simple closed (Jordan) curve.*

From the coloring of shocks into four types in the previous section, it can be seen that 2-shocks and 4-shocks are isolated points, whereas 1-shocks and 3-shocks are neighbored by other shocks of the same type. To build the shock graph we shall group together shocks of the same type that form a connected component, denoting the groups with labels $\tilde{1}, 2, \tilde{3}$ and 4, and breaking apart the $\tilde{1}$'s at branch-points.[2] Let each shock group be indexed by a distinct integer i, and let t_i denote its time (or times) of formation, corresponding to the radius function evaluated at the shocks in the group. Hence, t_i will be an interval for a $\tilde{1}$; for 2's, $\tilde{3}$'s and 4's it will be a single number. Finally, let # denote a *start* symbol and Φ a *terminal* symbol. The **SG** is a connected graph, rooted at a vertex labeled #, such that all other (non-terminal) vertices are shock groups, and directed edges to non-terminal vertices indicate the genesis of new shock groups.

Definition 2 *The Shock Graph of a 2-D shape,* **SG***$(\mathcal{O})$, is a labeled graph $G = (V, E, \gamma)$, with:*

- **vertices** $V = \{1, ..., n\}$*;*
- **edges** $(i, j) \in E \subseteq V \times V$ *directed from vertex i to vertex j if and only if $i \neq j$, $t_i \geq t_j$, and $i \cup j$ is connected in the plane;*
- **labels** $\gamma : V \rightarrow l$*, with $l \in \{\tilde{1}, 2, \tilde{3}, 4, \#, \Phi\}$; and*
- **topology** *such that, $\forall j \in V$ with $\gamma(j) \neq \#$, $\exists i \in V$ with $(i, j) \in E$.*

The **SG** is built by "reversing" the grassfire evolution, analogous to growing a shape by adding lumps of material onto its seeds. The children of the unique vertex labeled #, at which the graph is rooted, are the last shock groups to form. Vertices with label Φ are leaves of the **SG**, whose parents are the first shock groups to form. This reverse-time dependency is important because the last shocks to form correspond to the most significant (central) shape features.

[2] The ˜ symbol is used to denote a curve segment. A branch-point, where the maximal inscribed disc "touches" the boundary at more than two points, will be shared by all $\tilde{1}$'s that overlap at it.

Proposition 1 *Any 2-D shape $\mathcal{O}$ has a unique corresponding shock graph* **SG***$(\mathcal{O})$.*

PROOF: The uniqueness of the skeleton $S(X)$ follows from its definition as the union of maximum open discs. Hence the medial axis $Me(X)$, which is strictly contained in the skeleton $S(X)$ [48, pp. 382–383], is also unique. (In fact the two sets are very close since $\overline{Me}(X) = \overline{S}(X)$[31].) The coloring of medial axis points into four types in Section 1 is unique, which implies that a unique set of vertices exists for the corresponding **SG**. Finally, by Definition 2, the direction of an edge between two abutting vertices is ambiguous only when $t_i = t_j$ for all shocks in i and j. Due to the continuity of the radius function along the skeleton [48, pp. 381–382], the only possibility is that the two vertices share the point where they touch, in which case we have the contradiction that all shocks in i and j would lie in the *same* $\tilde{3}$, and hence in a single vertex. The uniqueness of the shock graph follows. □

2.1 The Shock Graph Grammar

The notion of entry-level categories for shape that we seek is intimately connected to the topological structure of the shock graph. This structure is highly constrained because the events that govern the birth, combination, and death of shock groups can be abstracted into a small number of rewrite rules, shown in Figure 2. In analogy to Leyton's Process Grammar [29], the rules have been grouped according to the semantic processes that they characterize, although the alphabet of shock types that they operate on is quite different from boundary-based codons.

Definition 3 *The Shock Graph Grammar,* **SGG***, is a quadruple $G = (V, \Sigma, R, S)$, with*

1. $V = \{\tilde{1}, 2, \tilde{3}, 4, \#, \Phi\}$*, the alphabet;*
2. $\Sigma = \{\Phi\}$*, the set of terminals;*
3. $S = \#$*, the start symbol; and*
4. $R = \{R_1, ..., R_{10}\}$*, the set of rules given in Figure 2.*

The rewriting system emphasizes the generative process of growing a shape by placing seeds, adding protrusions, forming unions, and so on. It operates by beginning at the start symbol and repeatedly replacing the left-hand side of a rule by the corresponding right-hand side until no further replacements can be made [28]. It is the **SGG** that captures the beauty of shock graphs, because the rules embody constraints from the domain of curve evolution. In particular,

Proposition 2 *The rewrite rules of the* **SGG** *are sufficient to derive the shock graph* **SG***$(\mathcal{O})$ of any 2-D shape $\mathcal{O}$.*

PROOF: A constructive proof appears in Appendix A. The strategy is to derive the rules by enumerating all legal parents and children for each vertex type.

We can now make several observations. First, since the same shock cannot be born at two distinct times there exists no path from a vertex back to itself. Hence, *the* **SG** *is a directed acyclic graph.* This has important consequences for object matching because the problem of searching directed acyclic graphs is computationally much simpler than that of searching arbitrary graphs [24]. Second, since there exist rules in the **SGG** whose left-hand sides do not consist of single nonterminals, *the* **SGG** *is not context-free.* Third, the rewrite rules indicate that a 2-shock and a 4-shock can only be added by rules 5 and 0 respectively, and that semantically equivalent rules exist for a $\tilde{3}$ (rules 6 and 1). Hence, *a 2-shock and a 4-shock are each semantically equivalent to a $\tilde{3}$ in a specific context.*

The **SG**'s for a variety of shapes are shown in Figure 8. All the graphs were generated automatically from the output of the shock detection process [49] displayed in Figure 7. Following the third observation, only label types $\tilde{1}$ and $\tilde{3}$ have been explicitly assigned. A $\tilde{3}$ with a parent $\tilde{1}$ at each end acts as a 2 (a neck), and a $\tilde{3}$ with a # as a parent acts as a 4 (a seed).

In the next Section we show that a shock graph can be reduced to a unique rooted shock tree, which in turn implies a hierarchical ordering of shape information (shock vertices). We then develop a formal approach to *significance-based matching*, where the key idea is to defeat complexity (when the database of shapes is diverse and large) by attending to most

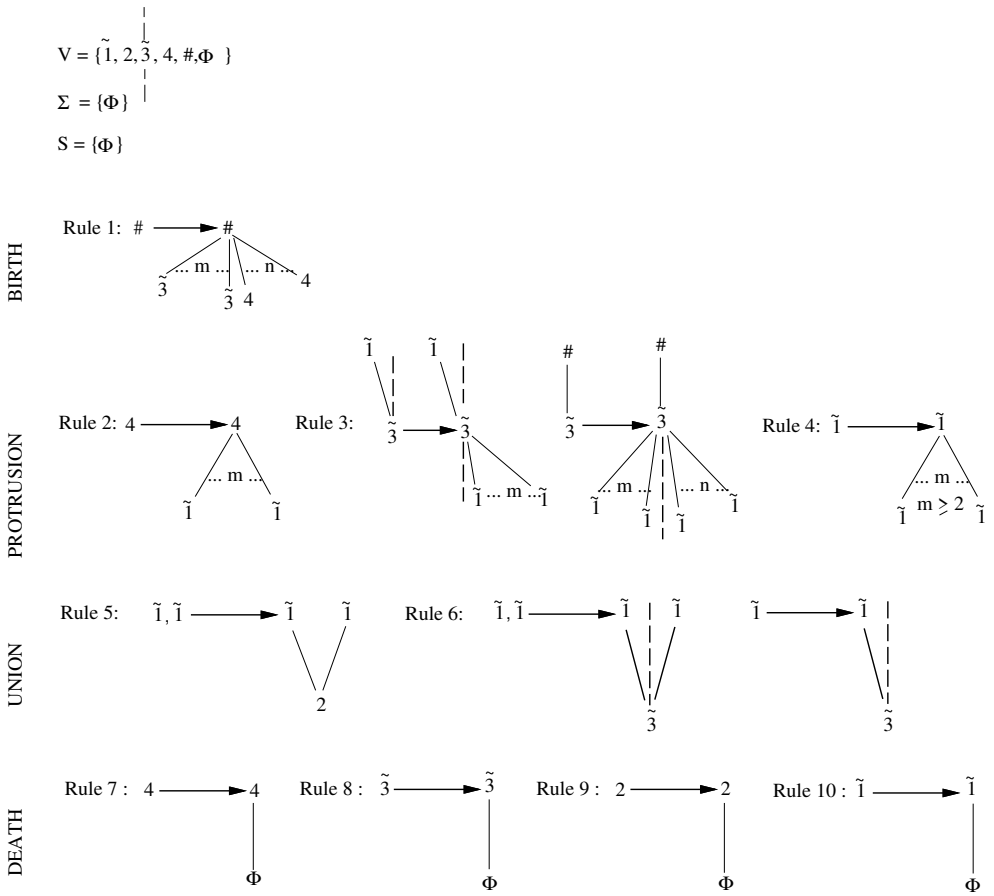

Figure 2: The Shock Graph Grammar, *SGG*. Dashed lines partition distinct ends of a $\tilde{3}$. The rules are grouped according to the different semantic processes (on the left) that they characterize. Note that the grammar is not context-free, e.g., rule 3 indicates that a $\tilde{1}$ can only be added onto an end of a $\tilde{3}$ that has no parent $\tilde{1}$.

significant components first, via a depth-first search of the underlying shock trees.

3 Shock Graph Matching

3.1 Problem Formulation

Given two shock graphs, one representing an object in the scene (V_2) and one representing a database object (V_1), we seek a method for computing their similarity. Unfortunately, due to occlusion and clutter, the shock graph representing the scene object may, in fact,

be embedded in a larger shock graph representing the entire scene. Thus we have a *largest subgraph isomorphism* problem, stated as follows: Given two graphs $G = (V_1, E_1)$ and $H = (V_2, E_2)$, find the maximum integer k, such that there exists two subsets of cardinality k, $E_1' \subseteq E_1$ and $E_2' \subseteq E_2$, and the induced subgraphs $G' = (V_1, E_1')$ and $H' = (V_2, E_2')$ are isomorphic [17]. Further, since our shock graphs are labeled graphs, consistency between node labels must be enforced in the isomorphism.

The largest subgraph isomorphism problem, can be formulated as a $\{0,1\}$ integer optimization problem. The optimal solution is a $\{0,1\}$ bijective mapping matrix M, which defines the correspondence between the vertices of the two graphs G and H, and which minimizes an appropriately defined distance measure between corresponding edge and/or node labels in the two graphs. More formally, we seek the matrix M, the global optimizer of the following [24, 32]:

$$\begin{aligned} \min \quad & -\frac{1}{2} \sum_{u \in V_1} \sum_{v \in V_2} M(u,v) ||u,v|| \\ \text{s.t.} \quad & \sum_{u' \in V_2} M(u,u') \leq 1, \ \forall u \in V_1 \\ & \sum_{v \in V_1} M(v,v') \leq 1, \ \forall v' \in V_2 \\ & M(x,y) \in \{0,1\}, \forall x \in V_1, \ y \in V_2 \end{aligned} \tag{2}$$

where $||.||$ is a measure of the similarity between the labels of corresponding nodes in the two shock graphs (see Section 3.4).

The above minimization problem is known to be NP-hard for general graphs [17], however, polynomial time algorithms exist for the special case of finite rooted trees. Matula and Edmonds [12] describe once such technique, involving the solution of $2n_1n_2$ network flow problems, where n_1 and n_2 represent the number of vertices in the two graphs. The complexity was further reduced by Reyner [40] to $O(n_1^{1.5}n_2)$ (assuming $n_1 \geq n2$), through a reduction to the bipartite matching algorithm of Hopcraft and Karp [21]. If we could transform our directed acyclic shock graphs to finite rooted trees, we could pursue a polynomial time solution to our problem.

In the following subsections we show that for any shock graph, there exists a unique rooted tree. Next, we present a method for comparing the coarse topological structure of two shock

trees which draws on a powerful recent result from the domain of semidefinite programming. Namely, the eigenvalue decomposition of an adjacency matrix corresponding to a shock tree leads to a property that is invariant to any consistent permutation or reordering of its subtrees (submatrices).[3] After defining a suitable measure of shock distance between corresponding nodes in two shock trees, we present a novel modification to Reyner's algorithm [40] which combines coarse topological matching with shock distance to solve our largest isomorphic subgraph problem in polynomial time.

3.2 Shock Graphs to Shock Trees

In this section we present a reduction that takes a DAG representing a shock graph to a unique vertex labeled rooted tree whose size is polynomially bounded by the size of the original shock graph. To begin, let $G = (V, E)$ be a DAG representing a shock graph on n vertices. A loop L is a subgraph of G formed by the intersection of two directed paths. More formally, L originates at a vertex b, follows two paths P_1 and P_2, and ends at the vertex t. We denote b as the *base* of L, t the *tip* of L, and P_1 and P_2 the *wings* of L. Referring to the protrusion and birth rewrite rules (rules $1, 2, 3$ and 4), in Figure 2, the base of L can be a vertex whose type is drawn from the set $\{\#, 4, \tilde{3}, \tilde{1}\}$. The wings, P_1 and P_2, are directed paths consisting of a sequence of vertices whose types are drawn from the set $\{4, \tilde{3}, \tilde{1}\}$ (rules $1, 2, 3, 4$, and 6). Finally, the tip of L can be a vertex of type either 2 or $\tilde{3}$ (rules 5 and 6).

Assume that the tip t of L is a vertex of type 2. Then by rule 9, L *will* be terminated at a vertex labeled Φ. Next, if t is a vertex of type 3, then P_1 and P_2 represent two directed sequences of shocks that enter at opposite ends of t. In this case, t can not satisfy rule 3, since it cannot be the root of any directed subgraph except a single node subgraph having label Φ (rule 8). We therefore conclude that the tips of all loops are adjacent to nodes having type Φ in G, and that each such tip participates in exactly one loop.

In our reduction, for each such tip node t we will maintain duplicate copies t_1 and t_2, and redefine L to be the union of b and two new disjoint paths $P_1' = P_1 \cup \{t_1\} \cup \{\Phi\}$ and

[3] A consistent permutation or reordering of a subtree is any (recursive) re-ordering of the tree's branches that maintains the same parent-child relations.

$P_2' = P_2 \cup \{t_2\} \cup \{\Phi\}$. It is easy to see, by induction on the number of tips in G, that such a reduction is unique and produces a directed, or equivalently, a rooted tree. Further, since G has only $O(n)$ tips, each of which is duplicated at most once, there is an $O(n)$ increase in size of the graph. To perform the reduction, we need only check the in-degree of any $\tilde{3}$'s and 2's, and duplicate them if necessary. The complete reduction is therefore a linear time process in terms of the number of vertices in G.

3.3 An Eigenvalue Characterization of a Shock Tree

The shock tree can be represented as a $\{0,1\}$ adjacency matrix, with 1's indicating adjacent nodes in the tree. Any shock subtree therefore defines a submatrix of the adjacency matrix. If, for a given shock subtree, we compute the eigenvalues of its corresponding submatrix, then the sum of the eigenvalues is invariant to any similarity transformation applied to the submatrix. This means that the eigenvalue sum is invariant to any consistent re-ordering of the subtrees! In terms of our largest subgraph isomorphism problem, finding the two shock subtrees whose eigenvalue sums are closest represents an approximation to finding the largest isomorphic subtrees.[4]

In order to efficiently compute the submatrix eigenvalue sums, we turn to the domain of semidefinite programming. A symmetric $n \times n$ matrix A with real entries is said to be positive semidefinite, denoted as $A \succeq 0$, if for all vectors $x \in R^n$, $x^t A x \geq 0$, or equivalently, all its eigenvalues are non-negative. We say that $U \succeq V$ if the matrix $U - V$ is positive semidefinite. For any two matrices U and V having the same dimensions, we define $U \bullet V$ as their inner product, i.e., $U \bullet V = \sum_i \sum_j U_{i,j} V_{i,j}$. For any square matrix U, we define $\text{trace}(U) = \sum_i U_{i,i}$. Let I denote the identity matrix having suitable dimensions. The following result, due to Overton and Womersley [37], characterizes the sum of the first k largest eigenvalues of a symmetric matrix in the form of a semidefinite convex programming problem:

Theorem 1 *For the sum of the first k eigenvalues of a symmetric matrix A, the following*

[4]This analysis considers only the topological structure of the shock graph. Later, we will factor in geometric information associated with its vertices.

semidefinite programming characterization holds:

$$\begin{aligned} \lambda_1(A) + \ldots + \lambda_k(A) = \quad & \max \quad A \bullet U \\ & \text{s.t.} \quad \text{trace}(U) = k \\ & \quad\;\; 0 \preceq U \preceq I, \end{aligned}$$

or, in a dual setting:

$$\begin{aligned} \lambda_1(A) + \ldots + \lambda_k(A) = \quad & \min \quad kz + \text{trace}(V) \\ & \text{S.T.} \quad zI + V \succeq A \\ & \quad\;\; V \succeq 0. \end{aligned}$$

Before applying the above theorem, we must first convert our shock trees to adjacency matrices. Given a bounded degree, rooted tree $G = (V, E)$ with $|V| = n$ and $|E| = m$, we define the adjacency matrix A of G to be a $n \times n$ symmetric, $\{0, 1\}$ matrix with its (i, j)-th entry $A_{i,j}$ equal to 1 if $(i, j) \in E$, and 0 otherwise. For each vertex $v \in G$, let $\delta(v)$ be the degree of v, and let $\delta(G)$ be the maximum degree over all vertices in G. For every vertex $u \in G$, we define $\chi(u)$ to be a vector in $R^{\delta(G)-1}$, obtained through the following procedure:

> For any child v of u in G, construct the adjacency matrix A_v of the induced subtree rooted at v, and for A_v, compute the quantity $\lambda_v = \lambda_1(A_v) + \ldots + \lambda_{\delta(v)}(A_v)$. Construct $\chi(u)$ as the vector formed by $\{\lambda_{v_1}, \ldots, \lambda_{v_{\delta(u)}}\}$ for which $\lambda_{v_1} \geq \ldots \geq \lambda_{v_{\delta(u)}}$.

The above procedure yields a vector assigned to each vertex in the shock tree, whose elements are the individual eigenvalue sums corresponding to the node's (subtree's) adjacency submatrix. Furthermore, for any rooted subtree, such a decomposition and vector coloring of the vertices is uniquely defined. As stated earlier, the power of the above formulation lies in the fact that if a symmetric matrix A undergoes any orthonormal transformation of the form $P^t AP$, the sum of its eigenvalues remains invariant. This, in turn, implies that this vector labeling of all rooted trees isomorphic to G not only has the same vector labeling but spans the same subspace in $R^{\delta(G)-1}$. Moreover, this extends to any rooted tree which has a subtree isomorphic to a subtree of G. In terms of our shock graphs, invariance to a permutation matrix P implies invariance to a re-ordering of the subtrees of the rooted tree described by A.

It now remains to be shown that such a vector labeling can be computed efficiently, i.e., that the λ_v function can be calculated in polynomial time. The elegance of Theorem (1) lies in the fact that the equivalent semidefinite programming problem can be solved, for any desired accuracy ϵ, in time polynomial in $O(n\sqrt{n}L)$ and $\log\frac{1}{\epsilon}$, where L is an upper bound on the size of the optimal solution, using a variant of the Interior Point method proposed by Alizadeh [1]. In section 3.5, we embed this procedure in our own algorithm for finding the largest isomorphic subtrees corresponding to two shock graphs. In addition, we factor in a measure of similarity between shock geometries, which we now discuss.

3.4 The Distance Between Two Vertices

The eigenvalue characterization introduced in the previous section applies to the problem of determining the topological similarity between two shock trees. Returning to the opening scenario, this, roughly speaking, defines an equivalence class of objects belonging to the same entry-level category. For example, a broad range of dogs will have very similar shock tree structures. On the other hand, when one is interested in discriminating between a short-legged Dachschund and "Loki", a Siberian Husky, geometric properties will play a significant role.

This geometry is encoded by information contained in each vertex of the shock tree. Specifically, recall from Section 2 that both $\tilde{1}$'s and $\tilde{3}$'s are curve segments of shocks. In the former case, the segment is directed, while in the latter case, there is a partial order but no preferred direction, since all the shocks were formed at the same time. Each shock in a segment is further labeled by its position, its time of formation (radius of the skeleton), and its direction of flow (or orientation in the case of $\tilde{3}$'s), all obtained from the shock detection algorithm [49]. In order to measure the similarity between two vertices u and v, we interpolate a low dimensional curve through their respective shock trajectories, and assign a cost $C(u,v)$ to an affine transformation that aligns one interpolated curve with the other. Intuitively, a low cost is assigned if the underlying structures are scaled or rotated versions of one another.

Assume that S and S' are two (sampled) shock sequences of the form $S = (s_1, \ldots, s_p)$

and $S' = (s'_1, \ldots, s'_q)$, where each shock point s_i represents a 4-tuple (x, y, t, α) corresponding to its Euclidean coordinates (x, y), formation time t, and direction α. Note that when the samples are obtained from a a $\tilde{1}$, the sequence is ordered by time of formation. On the other hand, for a $\tilde{3}$ there is a partial order to the samples, but no preferred direction. In the latter case, both directions will have to be tried. In order to find the 4D-simplex corresponding to the basis for the affine transformation (in a 4-D space) between the two sets, we choose three equidistant points on the chains formed by partial orders $(s_1 \prec \ldots \prec s_p)$ and $(s'_1 \prec \ldots \prec s'_q)$. Clearly, to preserve the partial order of the points in each sequence, s_1 should be transferred to s'_1, and s_p to s'_q.

Let (A, B) be the transformation pair for this partial order and, without loss of generality, assume that $p \leq q$. We apply the transformation (A, B) to sequence S to form the sequence $\widehat{S} = (\hat{s}_1, \ldots, \hat{s}_p)$. Let $\Psi(\widehat{S})$ and $\Psi(S')$ denote the interpolated 4-D curves passing through the points of the sets $\widehat{S}$ and S', respectively. A Hausdorff distance measure between the curves $\Psi(\widehat{S})$ and $\Psi(S')$ is defined by finding the closest point on curve $\Psi(S')$ for each point in the sequence $\widehat{S}$, and the closest point on curve $\Psi(\widehat{S})$ for each point in the sequence S':

$$\Delta(\Psi(\widehat{S}), \Psi(S')) = \sum_{x \in \widehat{S}} \inf_{y \in \Psi(S')} ||x - y||_2 + \sum_{x \in S'} \inf_{y \in \Psi(\widehat{S})} ||x - y||_2.$$

We observe that in a fixed dimension Euclidean space, the distance between a point and a low-degree smooth polynomial curve can be efficiently approximated. For example, if $\Psi(\widehat{S})$ and $\Psi(S')$ are piecewise linear approximations for $\widehat{S}$ and S', $\Delta(\Psi(\widehat{S}), \Psi(S'))$ can be computed in time $O(pq)$.

3.5 Algorithm for Matching Two Shock Trees

Our recursive algorithm for matching the rooted subtrees G and H corresponding to two shock graphs is inspired by the algorithm proposed by Reyner [40]. The algorithm recursively finds matches between vertices, starting at the root of the shock tree, and proceeding down through the subtrees in a depth-first fashion. The notion of a match between vertices incorporates two key terms: the first is a measure of the topological similarity of the sub-

trees rooted at the vertices (see Section 3.3), while the second is a measure of the similarity between the shock geometry encoded at each node (see Section 3.4). Unlike a traditional depth-first search which backtracks to the next statically-determined branch, our algorithm effectively recomputes the branches at each node, always choosing the next branch to descend in a best-first manner. One very powerful feature of the algorithm is its ability to match two trees in the presence of noise (random insertions and deletions of nodes in the subtrees).

Before stating our algorithm, some definitions are in order. Let $G = (V_1, E_1)$ and $H = (V_2, E_2)$ be the two shock graphs to be matched, with $|V_1| = n_1$ and $|V_2| = n_2$. Define d to be the maximum degree of any vertex in G and H, i.e., $d = \max(\delta(G), \delta(H))$. For each vertex v, we define $\chi(v) \in R^{d-1}$ as the unique eigen-decomposition vector introduced in Section 3.3.[5] Furthermore, for any pair of vertices u and v, let $C(u, v)$ denote the shock distance between u and v, as defined in Section 3.4. Finally, let $\Phi(G, H)$ (initially empty) be the set of final node correspondences between G and H representing the solution to our matching problem.

The algorithm begins by forming a $n_1 \times n_2$ matrix $\Pi(G, H)$ whose (u, v)-th entry has the value $C(u, v)||\chi(u) - \chi(v)||_2$, assuming that u and v are compatible in terms of their shock order, and has the value ∞ otherwise. Next, we form a bipartite edge weighted graph $\mathcal{G}(V_1, V_2, E_\mathcal{G})$ with edge weights from the matrix $\Pi(G, H)$.[6] Using the scaling algorithm of Goemans, Gabow, and Williamson [16], we then find the maximum cardinality, minimum weight matching in $\mathcal{G}$. This results in a list of node correspondences between G and H, called $\mathcal{M}_1$, that can be ranked in decreasing order of similarity.

From $\mathcal{M}_1$, we choose (u_1, v_1) as the pair that has the minimum weight among all the pairs in $\mathcal{M}_1$, i.e., the first pair in $\mathcal{M}_1$. (u_1, v_1) is removed from the list and added to the solution set $\Phi(G, H)$, and the remainder of the list is *discarded*. For the subtrees G_{u_1} and H_{v_1} of G and H, rooted at nodes u_1 and v_1, respectively, we form the matrix $\Pi(G_{u_1}, H_{v_1})$

[5] Note that if the maximum degree of a node is d, then excluding the edge from the node's parent, the maximum number of children is $d - 1$. Also note that if $\delta(v) < d$, then then the last $d - \delta(v)$ entries of χ are set to zero to ensure that all χ vectors have the same dimension.

[6] $G(A, B, E)$ is a weighted bipartite graph with weight matrix $W = [w_{ij}]$ of size $|A| \times |B|$ if, for all edges of the form $(i, j) \in E$, $i \in A$, $j \in B$, and (i, j) has an associated weight $= w_{i,j}$.

using the same procedure described above. Once the matrix is formed, we find the matching $\mathcal{M}_2$ in the bipartite graph defined by weight matrix $\Pi(G_{u_1}, H_{v_1})$, yielding another ordered list of node correspondences. The procedure is recursively applied to (u_2, v_2), the edge with minimum weight in $\mathcal{M}_2$, with the remainder of the list discarded.

This recursive process eventually reaches the leaves of the subtrees, forming a list of ordered correspondence lists (or matchings) $\{\mathcal{M}_1, \ldots, \mathcal{M}_k\}$. In backtracking step i, we remove any subtrees from the graphs G_i and H_i whose roots participate in a matching pair in $\Phi(G, H)$ (we enforce a one-to-one correspondence of nodes in the solution set). Then, in a depth-first manner, we first recompute $\mathcal{M}_i$ on the subtrees rooted at u_i and v_i (with solution set nodes removed). As before, we choose the minimum weight matching pair, and recursively descend. Unlike a traditional depth-first search, we are dynamically recomputing the branches at each node in the search tree. Processing at a particular node will terminate when either subtree loses all of its nodes to the solution set.

We can now state the algorithm more precisely:

procedure isomorphism(G,H)
  $\Phi(G, H) \leftarrow \emptyset$
  $d \leftarrow \max(\delta(G), \delta(H))$
  for $u \in V_G$ compute $\chi(u) \in R^{d-1}$ (see Section 3.3)
  for $v \in V_H$ compute $\chi(v) \in R^{d-1}$ (see Section 3.3)
  call match(root(G),root(H))
  return(cost($\Phi(G, H)$))
end

procedure match(u,v)
  do
    {
    let $G_u \leftarrow$ rooted subtree of G at u
    let $H_v \leftarrow$ rooted subtree of H at v
    compute $|V_{G_u}| \times |V_{H_v}|$ weight matrix $\Pi(G_u, H_v)$

$\mathcal{M} \leftarrow$ max cardinality, minimum weight bipartite matching
in $\mathcal{G}(V_{G_u}, V_{H_v})$ with weights from $\Pi(G_u, H_v)$ (see [16])
$(u', v') \leftarrow$ minimum weight pair in $\mathcal{M}$
$\Phi(G, H) \leftarrow \Phi(G, H) \cup \{(u', v')\}$
call match(u',v')
$G_u \leftarrow G_u - \{x | x \in V_{G_u}$ and $(x, w) \in \Phi(G, H)\}$
$H_v \leftarrow H_v - \{y | y \in V_{H_v}$ and $(w, y) \in \Phi(G, H)\}$
}
while ($G_u \neq \emptyset$ and $H_v \neq \emptyset$)

In terms of algorithmic complexity, observe that during the depth-first construction of the matching chains, each vertex in G or H will be matched at most once in the forward procedure. Once a vertex is mapped, it will never participate in another mapping again. The total time complexity of constructing the matching chains is therefore bounded by $O(n^2\sqrt{n \log\log n})$, for $n = \max(n1, n2)$ [16]. Moreover, the construction of the $\chi(v)$ vectors will take $O(n\sqrt{n}L)$ time, implying that the overall complexity of the algorithm is $\max(O(n^2\sqrt{n \log\log n}), O(n^2\sqrt{n}L)$.

The above algorithm provides, in polynomial time better than $O(n^3)$ an approximate optimal solution to the minimization problem in 2. The matching matrix M in (2) can be constructed using the mapping set $\Phi(G, H)$. Our algorithm is particularly well-suited to the task of matching two shock trees since it can find the best correspondence in the presence of occlusion and/or noise in the tree.

3.6 An Illustrative Example

To illustrate the matching algorithm, we consider the two shock trees shown in Figure 3 (top), each of which describes a different view of a brush. The underlying shocks, along with the final computed correspondences between nodes, are depicted in Figure 3 (bottom). The sequence of steps in finding this best correspondence (minimum-weight maximum cardinality

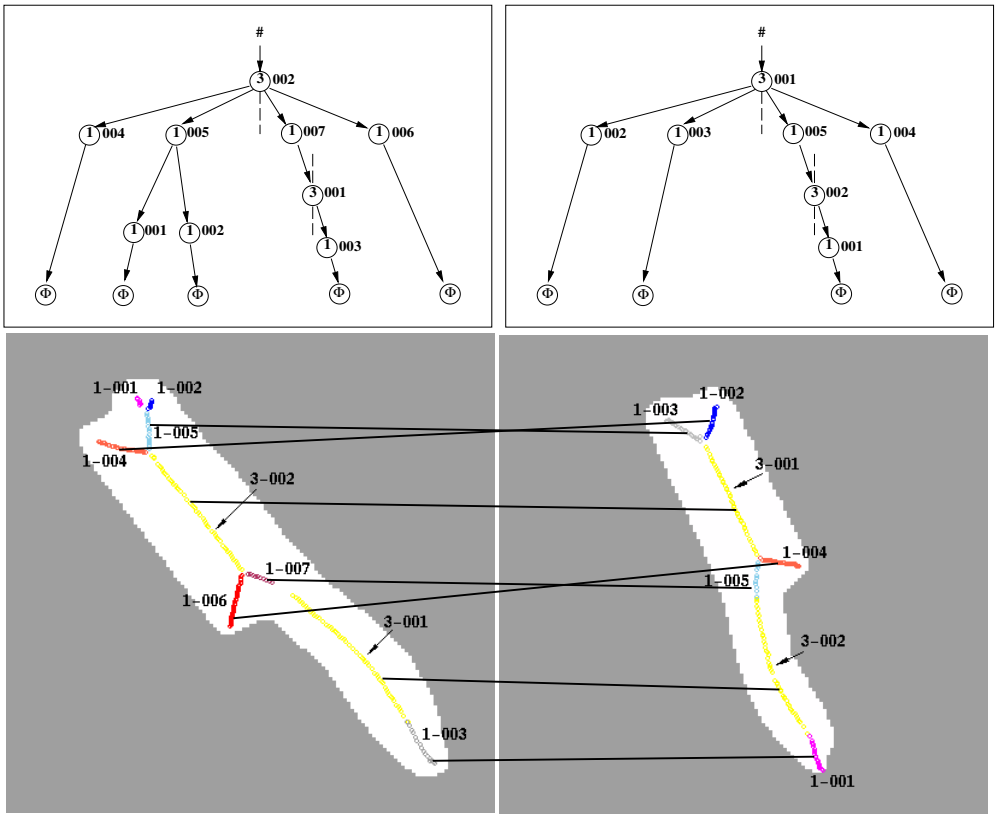

Figure 3: TOP: The shock trees derived for two different views of a brush. BOTTOM: The correspondences between nodes in the shock trees computed by the matching algorithm.

matching) between the two shock trees is shown in Figure 4. We briefly describe each step in the sequence:

- **Steps 1–4**

  The algorithm finds the minimum weight matching between the two shock trees, seeking to find the two subtrees which are maximally similar in terms of their topological structure and the geometry of their root nodes (shocks). In this example, the two subtrees rooted at 1-007 and 1-005 (denoted by bold circles in Figure 4) are selected as most similar. In step 2, this pair is added to the set of final correspondences (denoted by short-dashed circles), and the algorithm is recursively applied to the subtrees of 1-007 and 1-005. In this manner, the correspondences (3-001,3-002) and (1-003,1-001) are added to the set of final correspondences.

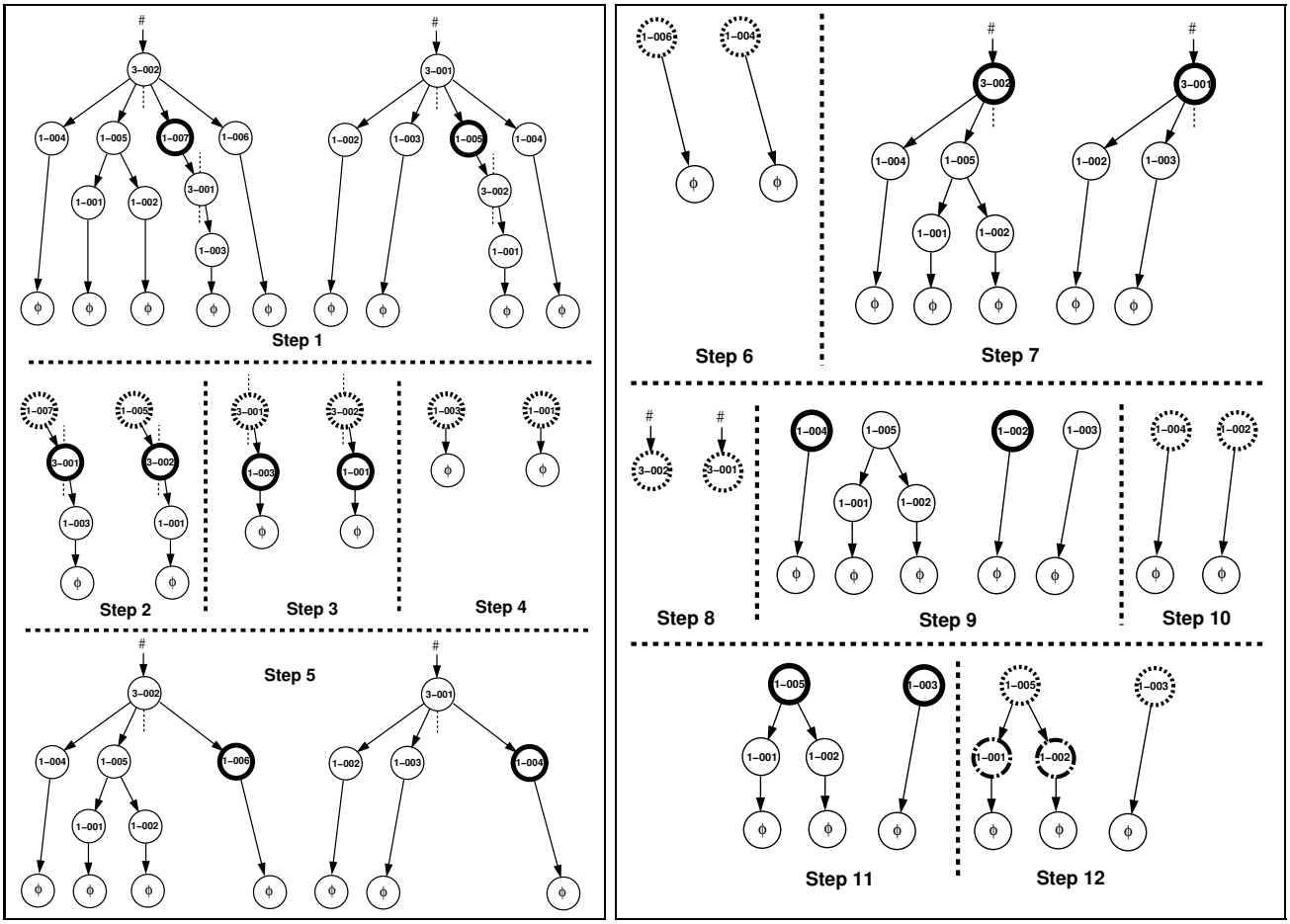

Figure 4: Step-by-step execution of the matching algorithm applied to the shock trees in Fig. 3. The roots of subtrees selected as most similar are denoted by bold circles. These are subsequently added to the set of final correspondences (short-dashed circles). Unmatched nodes are denoted by long-dashed circles.

- **Steps 5–6**

  After descending to the bottom of the subtrees rooted at (1-007,1-005), control is returned to (1-007,1-005) and these two subtrees are removed from the original shock graphs. From the resulting shock subtrees, we repeat the process of finding the best corresponding subtrees. In step 5, the subtree pair (1-006,1-004) is selected and added to the final correspondences in step 6.

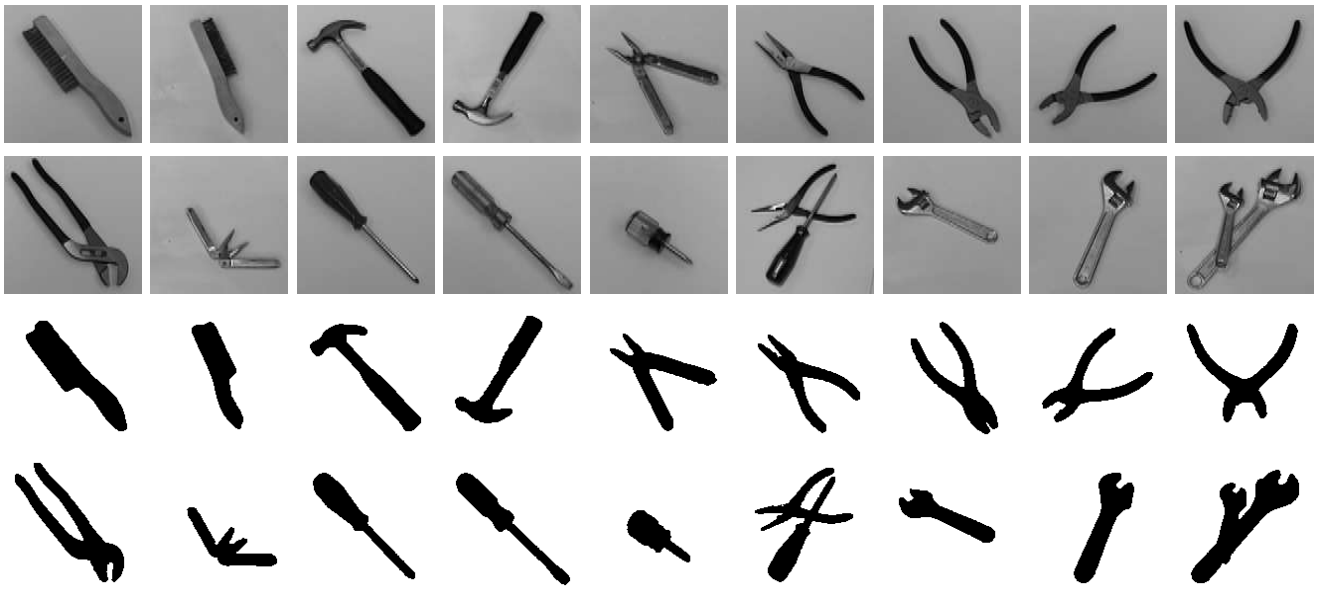

Figure 5: The 18 tool shapes used in our experiments. TOP: The originals. BOTTOM: The silhouettes were segmented automatically using the active contour developed in [50].

- **Steps 7–12**

  After removing the subgraphs originating at (1-006,1-004), a new pair (3-002,3-001) is selected in step 7, and added to the final correspondences in step 8. After removing this new pair, the process is applied to the remaining shock forests in step 9, resulting in the selection of the pair (1-004,1-002). This pair is added to the final correspondences in step 10. In step 11, the pair (1-005,1-003) is selected and added to the final correspondences in step 12. The algorithm terminates by leaving the nodes 1-001 and 1-002 as unmatched vertices (denoted by long-dashed circles) in the shock tree corresponding to leftmost object in Figure 3 (bottom).

4 Examples

We demonstrate our shape matching system with several examples. To evaluate its performance under occlusion, articulation of structures, and changes in viewing and imaging conditions, we constructed our own database of tool images, and selected 18 for the exper-

Figure 6: The 8 biological shapes. The hands are variations of a range image segmented from the NRCC database, and the da Vinci face profiles and horses were scanned from a book of his sketches.

iments described here, as shown in Figure 5 (top). The binary silhouettes were extracted automatically using the active contour developed in [50], as shown in Figure 5 (bottom). Observe that due to shadows and highlights, there may be slight discrepancies between the segmented outlines and the "true" ones; our matching algorithm is designed to robustly handle such discrepancies. We supplemented the tool shapes with silhouettes of 8 biological shapes, Figure 6.

The shock-based descriptions of representative shapes, numerically computed using the algorithms developed in [49], are shown in Figure 7, with the derived shock graphs in Figure 8. Note that apart from a "smallest scale" regularization induced by the sampling grid, the procedure for shock detection is automatic [49]. Notice how for each shape a hierarchy of components emerges, with the most significant components (e.g., the palm of the hand, and the neck of the pliers) placed closest to the root node. Similar descriptions were computed for each of the shapes in the database.

To evaluate our matcher's ability to compare objects based on their prototypical or coarse shape, we chose as a prototype for each of our 9 object classes, that object whose total distance to the other members of its class was a minimum.[7] We then computed the similarity between each remaining object in the database and each of the class prototypes, with the results shown in Table 1. For each row in the table, a box has been placed around the most similar shape. We note that for the 15 test shapes drawn from 9 classes, all but one are most similar to their class prototype, with the class prototype coming in a close second in that case. The recovered correspondences between nodes for the best matches in rows 1,

[7]For each of the three classes having only two members, the class prototype was chosen at random.

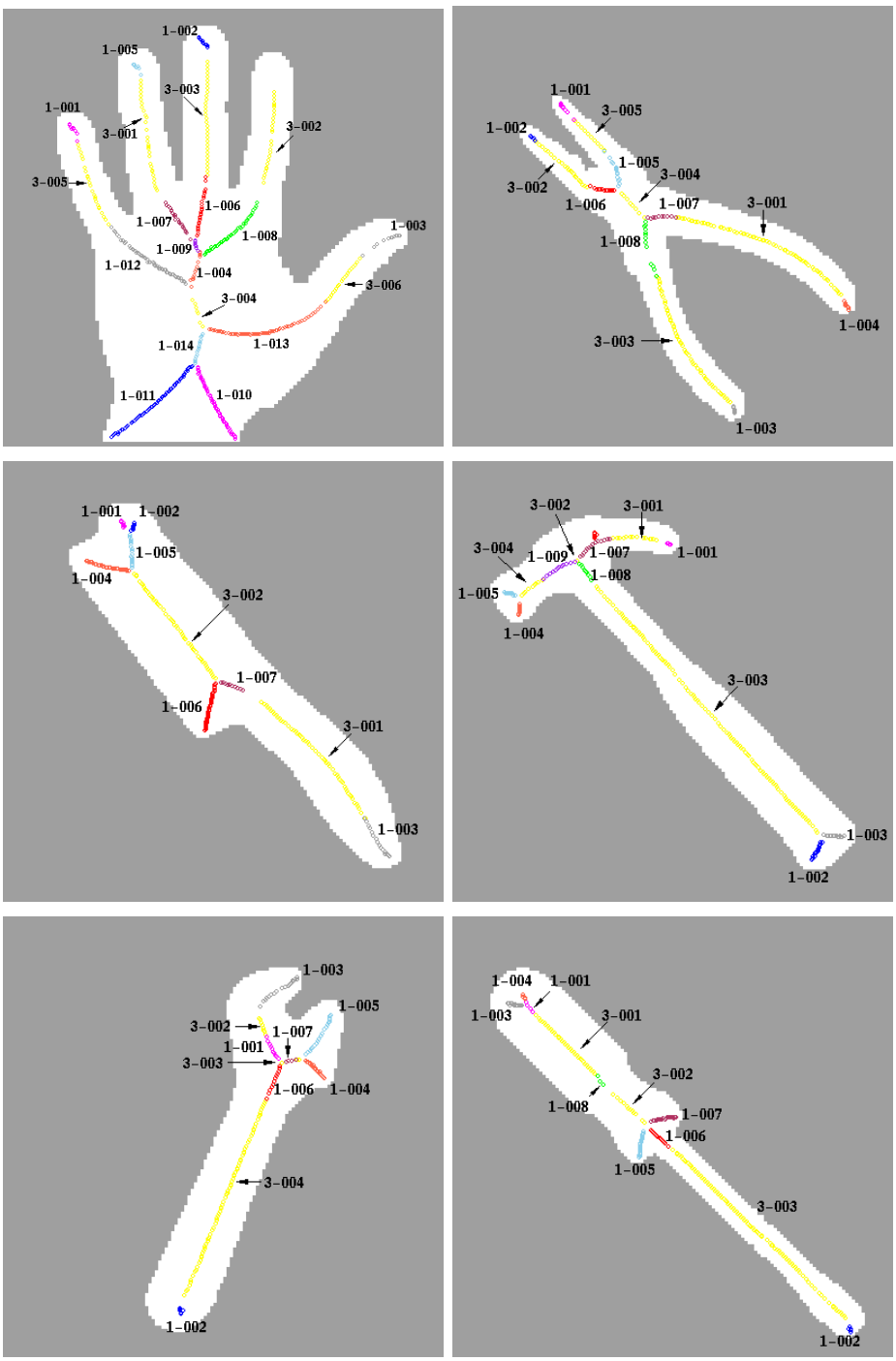

Figure 7: The shocks computed for a hand (segmented from the NRCC database), and a plier, a brush, a hammer, a wrench and a screwdriver, all from our own tool database. The labels correspond to vertices in the derived shock graphs, as shown in Figure 8.

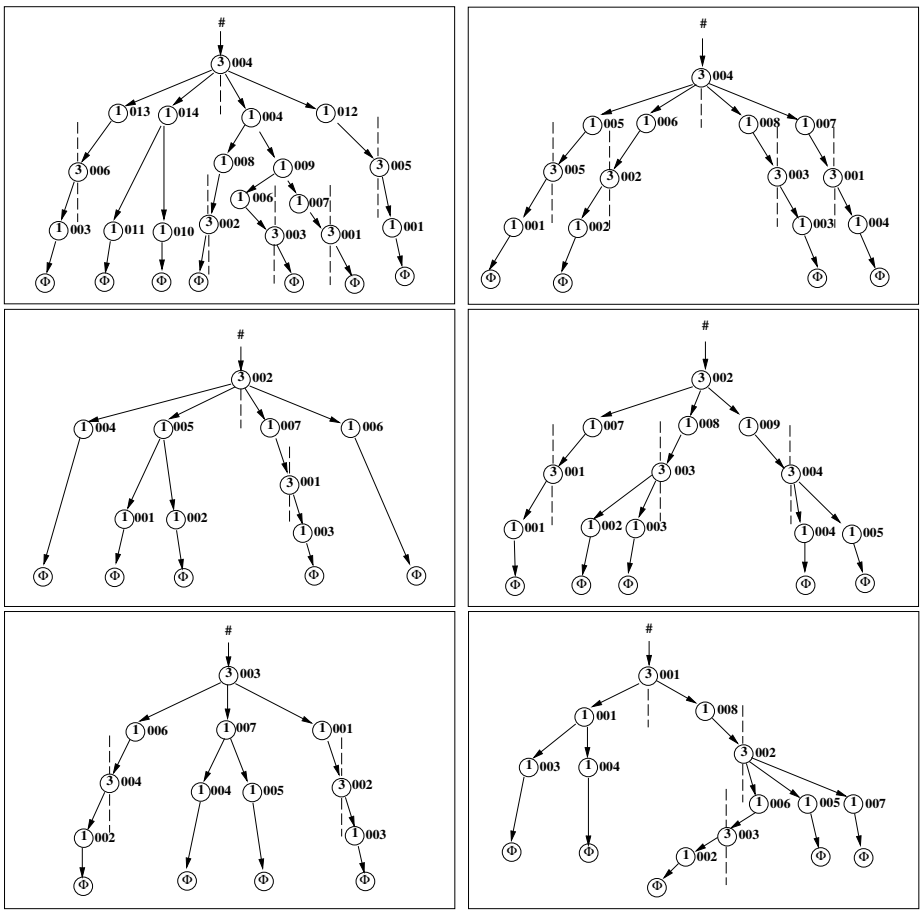

Figure 8: The shock graphs for a hand (top left), a plier (top right), a brush (middle left), a hammer (middle right), a wrench (bottom left) and a screwdriver (bottom right). Compare with Figure 7. The vertices are labeled according to their type, with the arrows in the direction of shape growth. The distinct ends of a $\tilde{3}$ are partitioned with a dashed line.

4, and 9 in Table 1, are shown in Figures 3 and 9.

Three very powerful features of our system our worth highlighting. First, the method is truly generic: the matching scores impose a partial ordering in each row, which reflects the qualitative similarity between structurally similar shapes. An increase in structural complexity is reflected in a higher cost for the best match, e.g., in the bottom two rows of Figure 1. Second, the procedure is designed to handle noise or occlusion, manifest as missing or additional vertices in the shock graph. Third, the depth-first search through subtrees is extremely efficient.

Instance	Distance to Class Prototype								
	0.02	2.17	4.48	3.55	2.96	0.21	4.58	14.33	10.01
	2.39	0.10	5.97	15.90	3.98	0.14	26.12	17.28	28.94
	10.89	4.72	2.08	12.24	3.12	2.15	19.73	10.11	12.64
	7.15	6.42	1.19	1.35	5.10	3.38	10.58	11.11	11.11
	4.08	7.72	2.98	1.49	4.26	4.14	26.60	13.54	14.21
	14.77	6.72	5.69	0.36	2.30	5.90	10.58	16.25	19.10
	7.86	8.90	5.94	0.74	1.59	1.10	10.81	10.39	16.08
	2.66	4.23	3.23	6.47	0.62	1.48	11.73	15.38	15.15
	3.18	5.31	1.25	4.64	0.60	1.30	14.18	17.22	9.08
	4.55	0.76	1.32	2.86	1.49	0.11	21.38	15.35	13.04
	6.77	19.46	22.11	13.27	8.21	29.50	0.15	5.12	5.03
	8.73	23.14	31.45	24.41	10.16	31.08	0.18	8.45	7.05
	12.46	19.0	27.40	14.58	24.26	17.10	8.85	7.49	16.93
	13.86	23.07	12.81	11.24	17.48	23.23	6.02	6.92	3.06
	15.73	21.28	14.10	12.46	19.56	19.21	9.53	7.12	5.06

Table 1: Experiment 1: similarity between database shapes and class prototypes. In each row, a box is drawn around the most similar shape (see the text for a discussion).

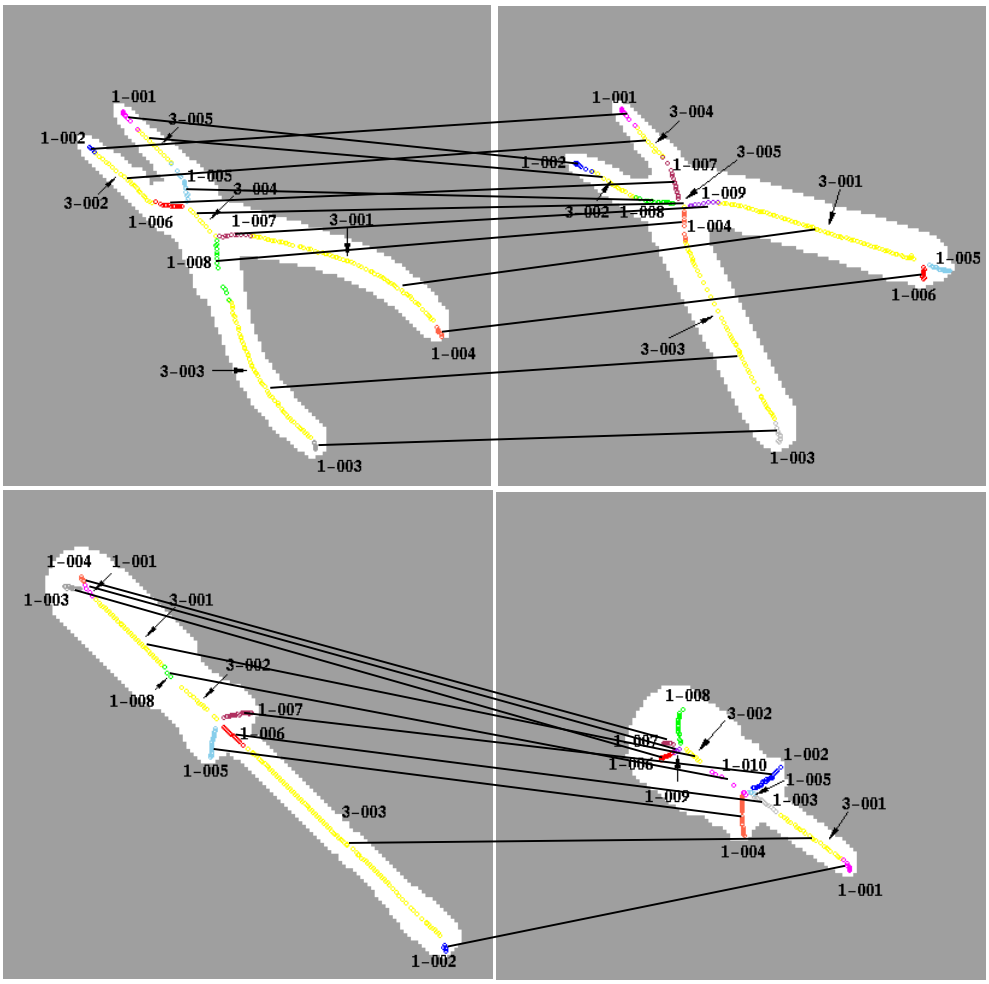

Figure 9: TOP TO BOTTOM: The computed correspondences between nodes for the best matches in rows 4 and 9 of Table 1.

In the next two experiments, Tables 2 and 3, we compare a number of objects to other members of their class as well as to a member from a different class. The objects have been chosen to illustrate the power of the matcher to deal with changes in image plane rotation, scale, deformation, occlusion, translation, and even slight rotation in depth. In both experiments, the results reflect the matcher's ability to compare shapes within the same class, at a finer scale.

Instance	**Distance to Class Exemplars**			
	8.89	0.38	4.70	5.94
	1.49	0.36	1.35	0.89
	5.21	9.95	0.57	10.01
	1.17	7.02	2.91	2.08

Table 2: Experiment 2: similarity between members of a class. Each row of the table highlights different aspects of matching invariance (in addition to translation): Rows 1 and 2: invariance to deformation, image rotation, and illumination; Row 3: invariance to deformation, scaling, and occlusion; and Row 4: invariance to deformation, scaling, image rotation, and illumination.

Instance	**Distance to Class Exemplars**			
	0.09	0.29	0.16	5.42
	1.48	1.30	3.46	0.11
	6.53	0.79	5.24	0.37

Table 3: Experiment 3: similarity between members of a class. Each row of the table highlights different aspects of matching invariance (in addition to translation): Row 1: invariance to scaling, deformation (different taper), and occlusion; Row 2: invariance to scaling, image rotation, and slight rotation in depth; and Row 3: invariance to image rotation, scaling, and occlusion.

5 Conclusions

In this paper, we have approached the problem of generic 2-D shape recognition by abstracting a representation of shape based on singularities of a curve evolution process into a shock graph, whose structure is characterized by a shock graph grammar. We have introduced a specific matching algorithm that manages complexity by matching most significant components first. The algorithm takes into account both the coarse topology of two shock trees as well as the geometry associated with the shocks in each vertex. The novelty of the algorithm lies in its eigen-decomposition of a shock tree. This representation provides a powerful means for efficiently computing the best correspondence between two shock graphs in the presence of noise and occlusion. Experiments with a variety of shapes demonstrate that the approach is generic, and robust under articulation, occlusion and changes in viewpoint.

Our work has lead to two exciting directions for research which we are currently pursuing. First, it should be clear that object representation and the development of matching algorithms are not independent of how a large database of objects should be organized. In future work we shall extend our matching algorithms to provide a framework for indexing 2-D objects. Using a vector of eigenvalue sums computed on the subtrees of a shock tree, similar subtrees can be retrieved from a database via a simple vector norm. Second, building on ideas from aspect graphs, we shall extend our approach in 2-D to a view-based strategy for generic 3-D object recognition. The intuitive idea is that a collection of sufficiently distinct projected views of an object can be represented by concatenating the associated shock graphs, after which a matching algorithm very similar to the one we have introduced here can be directly applied. Although much work remains to be done, empirical evidence indicates that the shock graph is quite stable under small changes in viewpoint. In contrast, except in constrained environments, the stable extraction of 3-D components has proved notoriously difficult.

A Local Configurations in the Shock Graph

A constructive approach to characterizing the structure of the shock graph is to determine all legal parents and children for each vertex type. An exhaustive enumeration of these local configurations remains tractable, since we have a small number of vertex types. A similar tabulation was first provided (though without proof) by Blum [6, p. 257], and related results were also derived in [49].

Recall that the **SG** is rooted at a unique vertex labeled # and has one or more terminal vertices labeled Φ. All other vertices are shock groups taken from the set $V \setminus \{\#, \Phi\} = \{\tilde{1}, 2, \tilde{3}, 4\}$. It follows from the "coloring" in Section 1 and the continuity of the radius function along the skeleton [48, pp. 381–382] that no 2, $\tilde{3}$, or 4 can "touch" a distinct 2, $\tilde{3}$ or 4. Therefore,

Observation 1 *No 2, $\tilde{3}$ or 4 can have a 2, $\tilde{3}$ or 4 as a parent or as a child.*

As a consequence, if a vertex has multiple parents or children taken from the set $V \setminus \{\#, \Phi\}$, such vertices can only be $\tilde{1}$'s. The following lemma holds for the case of multiple parents.

Lemma 1 *A vertex in the* **SG** *can have at most two $\tilde{1}$'s as parents. If it has exactly two $\tilde{1}$'s as parents, it must be a 2-shock.*

PROOF: Recall that a $\tilde{1}$ is a curve segment of 1-shocks along which the radius function varies monotonically. Let a $\tilde{1}$ of length L be parametrized by arc-length s, such that $R(s)$ increases monotonically with s. Let $\mathbf{x}(0) = (x(0), y(0))$ denote the first point, where the radius function is smallest. For each $s \in (0, L)$, there exists a continuous mapping from $\mathbf{x}(s)$ to its associated pair of bi-tangent points $(\mathbf{p}(s), \mathbf{q}(s))$, by which the two boundary segments associated with the interval $(0, L)$ can be reconstructed, see Figure 10 (a). Let $\alpha(s)$ be the angle between the line segments $\overline{\mathbf{p}(s)\mathbf{x}(s)}$ and $\overline{\mathbf{q}(s)\mathbf{x}(s)}$, on the narrower side. Two conditions must hold: 1) $\alpha(s)$ must be $\leq \pi$ for each $s \in (0, L)$, since the radius function $R(s)$ increases monotonically with s, 2) no other shock can lie in the reconstructed (shaded) region in Figure 10 (a), because the grassfire can only traverse a point in the plane once (the entropy condition in [23]).

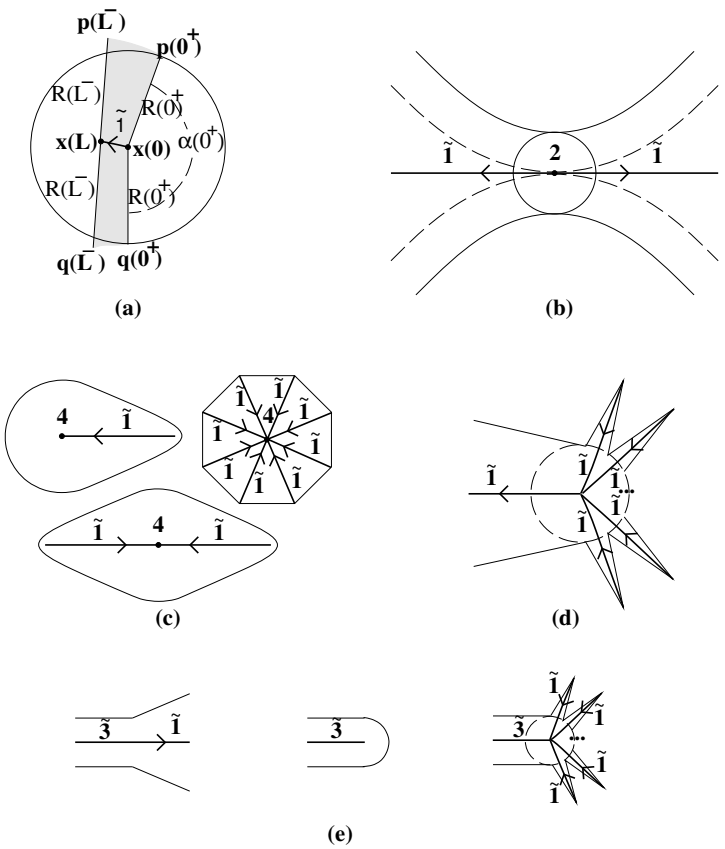

Figure 10: An illustration of shock patterns, with arrows drawn in the direction of increasing radius. (a) Geometry of a $\tilde{1}$. (b) Two $\tilde{1}$'s flow outwards from a 2. (c) An integer number of $\tilde{1}$'s may flow into a 4. (d) Two or more $\tilde{1}$'s may flow into a $\tilde{1}$. (e) An end-point of a $\tilde{3}$ may have a single $\tilde{1}$ flowing out of it, no shocks flowing into or out of it, or an integer number of $\tilde{1}$'s flowing into it.

Now, consider a second distinct $\tilde{1}$ such that its first point (where the radius function is smallest) coincides with $\mathbf{x}(0)$. The above two conditions must hold for the second $\tilde{1}$ as well. Hence, the only possibility is that $\alpha(0^+)$ is identically π for points infinitesimally close to $\mathbf{x}(0)$ on each $\tilde{1}$. As a consequence, the regions reconstructed by the two $\tilde{1}$'s will have no empty space between them, so the first points of no other $\tilde{1}$'s can coincide with $\mathbf{x}(0)$. We note that by the coloring in Section 1, $\mathbf{x}(0)$ is a 2-shock, as illustrated in Figure 10 (b). The result follows. □

To determine the remaining local configurations it is now sufficient to enumerate all the possible children for each vertex type. A child corresponds to an abutting vertex, containing

no shocks that formed after any of the shocks in the vertex under consideration (Definition 2). Note that by Observation 1, if the vertex is of type 2, $\tilde{3}$, or 4, only children of type $\tilde{1}$ have to be considered.

Children of a #: Since the **SG** is built in "reverse" time, any children of the # symbol will be the last shocks to form in the forward evolution. By the coloring in Section 1, an isolated point of annihilation is a 4; an annihilating curve segment is a $\tilde{3}$. These situations correspond to Rule 1 in Figure 2, which states that any number of $\tilde{3}$'s or 4's can act as seeds for the shape. These are the only possible children. A Φ cannot be a child since the interior of a Jordan curve is non-empty, and hence a null shape is disallowed.

Children of a $\tilde{1}$: A child of a $\tilde{1}$ can have no shock whose time of formation is greater than that of any shock in the $\tilde{1}$ (Definition 2). Hence, by the coloring in Section 1, a child cannot be a 4. It is possible for two or more distinct $\tilde{1}$'s to be children, e.g., one could place two or more triangular wedges around the dashed circle in Figure 10 (d). This corresponds to Rule 4. By Lemma 1, a 2 must be a child of two distinct $\tilde{1}$'s. This corresponds to Rule 5. A $\tilde{1}$ cannot have two or more $\tilde{3}$'s as children. (Refer to the proof of Lemma 1 but now let $\mathbf{x}(0)$ in Figure 10 (a) coincide with the end point of a $\tilde{3}$. A similar argument as before shows that no other $\tilde{3}$'s can have $\mathbf{x}(0)$ as an end point.) However, a single $\tilde{3}$ could be a child, as in Figure 10 (e) (left). The other end of the $\tilde{3}$ may or may not be the child of a distinct $\tilde{1}$. Thus, in contrast to Rule 5, we have two separate cases in Rule 6. Finally, when a $\tilde{1}$ has no abutting shock with smaller radius, it has a Φ as a child. This corresponds to Rule 10.

Children of a 2: If a 2 has a $\tilde{1}$ as a child, its time of formation must be greater than that of all shocks in the $\tilde{1}$ (Definition 2). However, this would violate the coloring in Section 1 (see Figure 10 (b)). Therefore, the only possible child is a Φ, corresponding to Rule 9.

Children of a $\tilde{3}$: In contrast to a $\tilde{1}$, which can only have children at the end point where the radius function is smallest (Definition 2), a $\tilde{3}$ can have children at either end point. It is possible for an integer number of $\tilde{1}$'s to be children, e.g., one could place one or more

triangular wedges around the dashed circle in Figure 10 (e) (right). However, by the same argument as in the previous paragraph, children can only be present at an end point that does not have a $\tilde{1}$ as a parent. Hence, we have two cases in Rule 3: if one end point of a $\tilde{3}$ has a parent $\tilde{1}$, an integer number of $\tilde{1}$'s can be children at the other end point; if neither end point has a parent $\tilde{1}$, an integer number of $\tilde{1}$'s can be children at each end point. Finally, when the $\tilde{3}$ has no abutting shock with smaller radius at either end point, as in Figure 10 (e) (middle), it has a Φ as a child. This corresponds to Rule 8.

Children of a 4**:** By the coloring in Section 1, a 4 corresponds to a medial axis point where the radius function achieves a strict local maximum. It may have an integer number of $\tilde{1}$'s as children, examples of which appear in Figure 10 (c). This corresponds to Rule 2. When it has no abutting $\tilde{1}$'s, as in the case of a perfect circle, it has a Φ as a child. This corresponds to Rule 7.

The above enumeration of all legal local vertex configurations in the **SG** shows that its structure is highly constrained. In particular, since a rewrite rule exists in Figure 2 for each legal parent/child of each vertex type, the proof of Proposition 2 is complete.

Acknowledgements We thank Jonas August and Allen Tannenbaum for many helpful discussions. Kaleem Siddiqi and Steven Zucker were supported by grants from AFOSR, NSERC, and NSF. Sven Dickinson gratefully acknowledges the support of NSF CAREER grant IRI-9623913.

References

[1] F. Alizadeh. Interior point methods in semidefinite programming with applications to combinatorial optimization. *SIAM J. Optim.*, 5(1):13–51, 1995.

[2] L. Alvarez, P. L. Lions, and J. M. Morel. Image selective smoothing and edge detection by nonlinear diffusion. *SIAM J. Numer. Anal.*, 29:845–866, 1992.

[3] V. Arnold. *The Theory of Singularities and Its Applications.* Lezioni Fermiane, Piza, Italy, 1991.

[4] R. Basri, L. Costa, D. Geiger, and D. Jacobs. Determining the similarity of deformable shapes. In *Proceedings, ICCV Workshop on Physics-Based Modeling in Computer Vision*, pages 135–143, 1995.

[5] R. Basri and S. Ullman. The alignment of objects with smooth surfaces. In *Second International Conference on Computer Vision (Tampa, FL, December 5–8, 1988)*, pages 482–488, Washington, DC, 1988. Computer Society Press.

[6] H. Blum. Biological shape and visual science. *J. Theor. Biol.*, 38:205–287, 1973.

[7] R. Brockett and P. Maragos. Evolution equations for continuous-scale morphology. In *Proceedings of the IEEE Conference on Acoustics, Speech and Signal Processing*, San Francisco, CA, March 1992.

[8] H. H. Bulthoff and S. Edelman. Psychophysical support for a two-dimensional view interpolation theory of object recognition. *Proceedings of the National Academy of Sciences*, 89:60–64, January 1992.

[9] C. A. Burbeck and S. M. Pizer. Object representation by cores: Identifying and representing primitive spatial regions. *Vision Research*, 35:1917–1930, 1995.

[10] J. Burns and L. Kitchen. Recognition in 2D images of 3D objects from large model bases using prediction hierarchies. In *Proceedings, International Joint Conference on Artificial Intelligence*, pages 763–766, Milan, Italy, 1987.

[11] S. Dickinson, A. Pentland, and A. Rosenfeld. 3-D shape recovery using distributed aspect matching. *IEEE PAMI*, 14(2):174–198, 1992.

[12] J. Edmonds and D. Matula. An algorithm for subtree identification. *SIAM Rev.*, 10:273–274 (Abstract), 1968.

[13] D. Forsyth, J. Mundy, A. Zisserman, C. Coelho, A. Heller, and C. Rothwell. Invariant descriptors for 3d object recognition and pose. *IEEE PAMI*, 13:971–992, October 1991.

[14] A. François and G. Medioni. Generic shape learning and recognition. In *International Workshop on Object Representation in Computer Vision*, April 1996.

[15] K. Fu, editor. *Syntactic Pattern Recognition, Applications*. Springer-Verlag, New York, 1977.

[16] H. N. Gabow, M. X. Goemans, and D. P. Williamson. An efficient approximate algorithm for survivable network design problems. *Proc. of the Third MPS Conference on Integer Programming and Combinatorial Optimization*, pages 57–74, 1993.

[17] M. Garey and D. Johnson. *Computer and Intractability: A Guide to the Theory of NP-Completeness*. Freeman, San Francisco, 1979.

[18] Y. Gdalyahu and D. Weinshall. Measures for silhouettes resemblance and representative silhouettes of curved objects. In *Computer Vision - ECCV '96*, Cambridge, UK, April 1996.

[19] S. Gold and A. Rangarajan. A graduated assignment algorithm for graph matching. *IEEE PAMI*, 18(4):377–388, 1996.

[20] U. Grenander. *Elements of Pattern Theory*. Johns Hopkins University Press, New Baltimore, 1996.

[21] J. Hopcroft and R. Karp. An $n^{\frac{5}{2}}$ algorithm for maximum matchings in bipartite graphs. *SIAM J. Comput.*, 2:225–231, 1973.

[22] K. Ikeuchi and T. Kanade. Automatic generation of object recognition programs. *Proceedings of the IEEE*, 76:1016–1035, 1988.

[23] B. B. Kimia, A. Tannenbaum, and S. W. Zucker. Shape, shocks, and deformations I: The components of two-dimensional shape and the reaction-diffusion space. *International Journal of Computer Vision*, 15:189–224, 1995.

[24] J. Kobler. *The graph isomorphism problem: its structural complexity*. Birkhauser, Boston, 1993.

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

[26] D. J. Kriegman and J. Ponce. On recognizing and positioning curved 3d objects from image contours. *IEEE PAMI*, 12:1127–1137, 1990.

[27] P. D. Lax. Shock waves and entropy. In E. H. Zarantonello, editor, *Contributions to Nonlinear Functional Analysis*, pages 603–634, New York, 1971. Academic Press.

[28] H. Lewis and C. Papadimitriou. *Elements of the Theory of Computation*. Prentice-Hall, 1981.

[29] M. Leyton. A process grammar for shape. *Artificial Intelligence*, 34:213–247, 1988.

[30] R. Malladi, J. A. Sethian, and B. C. Vemuri. Shape modeling with front propagation: A level set approach. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 17(2):158–175, February 1995.

[31] G. Matheron. Examples of topological properties of skeletons. In J. Serra, editor, *Image Analysis and Mathematical Morpholpogy, Part II: Theoretical Advances*, pages 217–238. Academic Press, 1988.

[32] E. Mjolsness, G. Gindi, and P. Anandan. Optimization in model matching and perceptual organization. *Neural Computation*, 1:218–229, 1989.

[33] T. Moons, E. Pauwels, L. V. Gool, and A. Oosterlinck. Foundations of semi-differential invariance. *International Journal of Computer Vision*, 14(1):25–47, January 1995.

[34] H. Murase and S. Nayar. Visual learning and recognition of 3-D objects from appearance. *International Journal of Computer Vision*, 14:5–24, 1995.

[35] R. L. Ogniewicz and O. Kübler. Hierarchic voronoi skeletons. *Pattern Recognition*, 28:343–359, 1995.

[36] S. J. Osher and J. A. Sethian. Fronts propagation with curvature dependent speed: Algorithms based on hamilton-jacobi formulations. *Journal of Computational Physics*, 79:12–49, 1988.

[37] M. L. Overton and R. S. Womersley. Optimality conditions and duality theory for minimizing sums of the largest eigenvalues of symmetric matrices. *Math. Programming*, 62(2):321–357, 1993.

[38] E. Pauwels, T. Moons, L. J. V. Gool, P. Kempenaers, and A. Oosterlinck. Recognition of planar shapes under affine distortion. *International Journal of Computer Vision*, 14(1):49–65, January 1995.

[39] A. Pope and D. Lowe. Learning object recognition models from images. In *Proceedings, IEEE International Conference on Computer Vision*, pages 296–301, Berlin, May 1993.

[40] S. W. Reyner. An analysis of a good algorithm for the subtree problem. *SIAM J. Comput.*, 6:730–732, 1977.

[41] W. Richards and D. D. Hoffman. Codon constraints on closed 2d shapes. *CVGIP*, 31(2):265–281, 1985.

[42] H. Rom and G. Medioni. Hierarchical decomposition and axial shape description. *IEEE PAMI*, 15(10):973–981, October 1993.

[43] E. Rosch, C. B. Mervis, W. D. Gray, D. M. Johnson, and P. Boyes-Braem. Basic objects in natural categories. *Cognitive Psychology*, 8:382–439, 1976.

[44] A. Rosenfeld. *Picture Languages: Formal Models for Picture Recognition.* Academic Press, New York, 1979.

[45] G. Sapiro and A. Tannenbaum. Affine invariant scale-space. *International Journal of Computer Vision*, 10:25–44, 1993.

[46] S. Sclaroff. Deformable prototypes for encoding shape categories in image databases. *Pattern Recognition*, 30(4), 1997.

[47] S. Sclaroff and A. Pentland. Modal matching for correspondence and recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 17(6):545–561, June 1995.

[48] J. Serra. *Image Analysis And Mathematical Morphology.* Academic Press, 1982.

[49] K. Siddiqi and B. B. Kimia. A shock grammar for recognition. Technical Report LEMS 143, LEMS, Brown University, September 1995.

[50] K. Siddiqi, Y. B. Lauzière, A. Tannenbaum, and S. W. Zucker. Area and length-minimizing flows for shape segmentation. *IEEE Transactions on Image Processing*, to appear, 1997.

[51] K. Siddiqi, A. Tannenbaum, and S. W. Zucker. Hyperbolic "smoothing" of shapes. *Sixth International Conference On Computer Vision*, To appear, 1998.

[52] Z. S. G. Tari, J. Shah, and H. Pien. Extraction of shape skeletons from grayscale images. *Computer Vision and Image Understanding*, May 1997.

[53] H. Tek, P. Stoll, and B. Kimia. Shocks from images: Propagation of orientation elements. In *International Conference on Computer Vision and Pattern Recognition*, pages 839–845. IEEE Computer Society Press, 1997.

[54] M. Turk and A. Pentland. Eigenfaces for recognition. *Journal of Cognitive Neuroscience*, 3(1):71–86, 1991.

[55] S. Ullman. The visual analysis of shape and form. In M. S. Gazzaniga, editor, *The Cognitive Neurosciences*, pages 339–350, Cambridge, MA, 1995. MIT Press.

[56] S. Ullman and R. Basri. Recognition by linear combinations of models. *IEEE PAMI*, 13(10):992–1006, 1991.

[57] S. Zhu and A. L. Yuille. Forms: a flexible object recognition and modelling system. *International Journal of Computer Vision*, 20(3):187–212, 1996.