

Scale-Space Representation of 3D Models and Topological Matching

Dmitriy Bespalov[†] Ali Shokoufandeh[†] William C. Regli^{†‡} Wei Sun[‡]

Department of Computer Science[†]
Department of Mechanical Engineering & Mechanics[‡]
College of Engineering
Drexel University
3141 Chestnut Street
Philadelphia, PA 19104

ABSTRACT

Reeb graphs have been shown to be effective for topology matching of 3D objects. Their effectiveness breaks down, however, when the individual models become very geometrically and topologically detailed—as is the case for complex machined parts. The result is that Reeb graph techniques, as developed for matching general shape and computer graphics models, produce poor results when directly applied to create engineering databases.

This paper presents a framework for shape matching through scale-space decomposition of 3D models. The algorithm is based on recent developments in efficient hierarchical decomposition of metric data using its spectral properties. Through spectral decomposition, we reduce the problem of matching to that of computing a mapping and distance measure between vertex-labeled rooted trees. We use a dynamic programming scheme to compute distances between trees corresponding to solid models. Empirical evaluation of the algorithm on an extensive set of 3D matching trials demonstrates both robustness and efficiency of the overall approach.

Categories and Subject Descriptors

F.2.2 [Non-numerical Algorithms and Problems]: Pattern matching; I.3.5 [Computer Graphics]: Computational Geometry and Object Modeling; J.6 [Computer-Aided Engineering]: Computer-aided design (CAD); H.3.3 [Information Search and Retrieval]: Clustering.

General Terms

Algorithms Design Theory

Keywords

Solid modelling, Scale-space, Matching.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, to republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.

SM'03, June 16–20, 2003, Seattle, Washington, USA.
Copyright 2003 ACM 1-58113-706-0/03/0006 ...\$5.00.

1. INTRODUCTION

The problem of 3D object recognition is often formulated as that of matching configurations of features. Such configurations are often represented as vertex-labeled graphs, whose nodes represent 3D features (or their abstractions), and whose edges represent spatial relations (or constraints) between the features. The relations are typically geometric or hierarchical, but can include other types of information. To match two 3D models means to establish correspondences between their constituting features. In this context, *features* are intrinsic properties of the 3D shape which may encompass local geometry and topology related to design or manufacturing operations. To evaluate the quality of a match, one defines an overall distance measure, whose value depends on similarities of their graph representations.

Due to the importance of the 3D matching problem in a number of disciplines, there has been a growing interest in developing efficient algorithms for matching vertex-labeled graphs. Previous work on 3D matching (see Section 2) has typically focused on the problem of finding a correspondence between the vertices of two graphs. However, the assumption of direct graph matching is a very restrictive one, for it assumes that the primitive features (nodes) in the two graphs are easy to extract and are invariant in their level of abstraction. If there is a lesson to be learned from the feature recognition community [13, 22], it is that features are not easy to extract and that they are not invariant across different domains. This paper presents a wholly new techniques that goes beyond existing feature-based indexing.

Several existing approaches to the problem of constructing the graph representations of 3D models suffer from computational inefficiency and/or from an inability to handle perturbations in models. This paper seeks a solution to this problem while addressing drawbacks of existing approaches. Drawing on recently-developed techniques from the domain of spectral clustering, we have explored an *efficient* method for mapping a 3D model to a rooted tree. This mapping not only simplifies the original 3D model representation, but it *retains important information* about both local (features) as well as global structure of the model.

Matching of 3D models can now be reduced to the much simpler problem of matching rooted trees using a *recursive structural* similarity framework. We consider one such similarity measure, based on a dynamic programming framework, and show that the framework realizes the desired matching between components of the original 3D models. The result is a more efficient and more stable approach to 3D matching.

2. RELATED WORK

Our research aims to bring information retrieval to CAD databases, enabling them to have indexing and query mechanisms like those beginning to found in multimedia databases and knowledge management systems.

2.1 Comparing Solid Models

The brief literature in this area consists of results from engineering, computer science and, in particular, computer vision communities. Elinson et al. [9], and Cicirello and Regli [21, 5, 6] examined how to develop graph-based data structures to capture feature relationships and create heuristic similarity measures among artifacts. More recent work in [4] examined manufacturing feature-based similarity measurement. Elsewhere, automatic detection of part families [20] and topological similarity assessment of polyhedral models [28] has been examined.

Historically GT coding was the way of indexing of parts and part families [27], this facilitated process planning and cell-based manufacturing by imposing a classification scheme (a human-assigned alphanumeric string) on individual machined parts. While there have been a number of efforts to automate the generation of GT codes [2, 23, 3, 14, 12], none have been fully transitioned to commercial practice.

2.2 Comparing Shape Models

The computer vision and computer graphics research communities has typically viewed shape matching as a problem in 2D. This has changed in the past several years with the ready availability of 3D models (usually meshes or point clouds) generated from range and sensor data [1]. A considerable body of work has emerged to interrogate acquired datasets; we briefly review some of this work.

Thompson et al. [29, 32] reverse engineered designs by generating surface and machining feature information of range data collected from machined parts. Hilaga et al. [15] present a method for matching 3D topological models using Multiresolutional Reeb graphs. A Reeb graph is a skeletal structure defined by a continuous scalar function operating on an object. It shares properties similar to medial axes (in 2D) and medial surfaces (in 3D), however Reeb graphs avoid the degeneracies that minor surface perturbations can cause to medial structures. In this work, the Reeb graphs are used to assess distances among a set of pre-categorized shape models (i.e., airplanes, cars, animals, etc). Other beneficial properties of their approach include invariance under certain transformations and robustness under variations in the quality of the models.

Osada et al. [18] developed a method that creates an abstraction the 3D models as probability distribution of samples from a shape function acting on the model. Specifically, the measure the similarity between two models by measuring the similarity between their shape distributions. Shape distributions are generated by random sampling of points on the surface of the model. In their paper, they empirically study five different shape functions and conclude (experimentally) that a function they call $D2$ (which measures the distance between two random points on the surface of a model) results in the best shape classification method. Their database is a set of 130 shape models in VRML format gathered from the Internet. Elad et al. [8] introduced both iterative and interactive approach to the problem of searching a database of 3D models in VRML format.

Cyr and Kimia [7] proposed a similarity measure for comparing two projected 2D views of 3D objects. In fact, they used the same 2D measure to decompose the viewing sphere of 3D objects in terms of groups of similar views that in turn can be used to generate the aspect graph characterization of a 3D objects. They showed

the performance of their system for the task of 3D object recognition in computer vision, where the main objective is to identify the 3D pose of an object using a set of characteristic 2D views (view-based 3D object recognition). It is not clear how this framework can be generalized to measure the similarity of two distinct objects with similar components.

In computer vision community, the problem of object recognition is often reformulated as that of matching feature graphs. Several researchers have developed algorithms that find one-to-one correspondences between graph nodes. Shapiro and Haralick [24] proposed a matching algorithm based on comparing weighted primitives (weighted attributes and weighted relation tuples) using a normalized distance for each primitive property that is inexactly matched. Kim and Kak [16] used a combination of discrete relaxation and bipartite matching in model-based 3-D object recognition. Pellilo et al. [19] devised a quadratic programming framework for matching association graphs using a maximal clique reformulation, while Gold and Rangarajan [10] used graduated assignment for matching graphs derived from feature points and image curves. Siddiqi et al. combined a bipartite matching framework with a spectral decomposition of graph structure to match shock graphs [26], while Shokoufandeh et al. [25] extended this framework to directed acyclic graphs that arise in multi-scale image representations. Hancock and his colleagues have also proposed numerous frameworks for graph matching, including [17].

For a recent survey on vision and graphics-based matching techniques, and their limitations, readers are referred to [30]. In general, shape matching-based approaches operate only on the gross-shapes of single parts (not applicable to considering assembly structures) and does not operate directly on the solid models or consider semantically meaningful engineering information (i.e., manufacturing or design features, tolerances). Retrieval strategies are usually based on a query-by-example or query-by-sketch paradigm. The Princeton 3D shape database that has been used in a number of these studies [15, 18] contains mainly models from 3D graphics and rendering, and not any models that are specifically engineering, solid modeling or mechanical CAD oriented.

3. FEATURE DECOMPOSITION

During the last decade, hierarchical segmentation has become recognized as a very powerful tool for designing efficient algorithms. The most common form of such hierarchical segmentations is the scale-space decomposition in computer vision. Intuitively, an inherent property of real-world objects is that they only exist as meaningful entities over certain ranges of scale. This fact, that objects in the world appear in different ways depending on the scale of observation, has important implications if one aims at describing them. Specifically, the need for multi-scale representation arises when designing methods for automatically analyzing and deriving information from real-world measurements.

In the context of solid models, the notion of scale can be simplified in terms of the levels for the 3D features. The notion of *feature* in this sense draws from the computer vision literature rather than the CAD literature. Namely, given an object \mathcal{M} , we are interested in partitioning \mathcal{M} , into k features $\mathcal{M}_1, \dots, \mathcal{M}_k$, with $\mathcal{M}_i \cap \mathcal{M}_j = \emptyset$, for $1 \leq i < j \leq k$, and $\mathcal{M} = \cup_i \mathcal{M}_i$ subject to maximization of some similarity measure, $f(\mathcal{M}_i)$, defined on the 3D elements forming each \mathcal{M}_i . In a finer scale, each feature \mathcal{M}_i will be in turn decomposed into $j = 1, \dots, k_i$ sub-features, subject to maximization of some similarity measures.

There are three central components in aforementioned process, the number of components at each scale of decomposition, k , the feature similarity function $f(\cdot)$, and the number of scales of de-

composition process, ℓ . In most pattern recognition applications, the value of k is a control parameter. Namely, if models \mathcal{M} and \mathcal{M}' are topologically similar, the k major components at every scale should also be similar. The similarity function $f(\mathcal{A})$ will assign an overall metric to the quality of 3D elements participating in the construction of feature \mathcal{A} . Finally, the depth of decomposition will be controlled depending on the quality of a feature in comparison to all its sub-features. Specifically, assume \mathcal{A} represents a feature at scale i , and $\mathcal{A}_1, \dots, \mathcal{A}_j$, for $j \leq k$ represents its sub-features at scale $i+1$, then the decomposition process should proceed to scale $i+1$ with respect to feature \mathcal{A} if and only if $f(\mathcal{A}) \leq f(\mathcal{A}_1) + f(\mathcal{A}_2) + \dots + f(\mathcal{A}_j)$. This simple criteria for expansion of scale-space at every feature has its roots in information theory. It is in fact motivated by the entropy of feature \mathcal{A} as oppose to its sub-features $\mathcal{A}_1, \dots, \mathcal{A}_j$.

Our interest in scale-space feature decomposition is motivated by its ability to transform the problem of matching of 3D models to hierarchical matching problems in rooted trees. Specifically, let $\mathcal{M}_1, \mathcal{M}_2$ denote two 3D models, under similarity metrics $f_1(\cdot)$ and $f_2(\cdot)$, respectively. Ideally, we seek a scale-space decomposition that can map each 3D model to the set of primitive features, in which the two feature decomposition can be directly compared. However, in general, this is not possible without introducing unacceptable discrepancies in choosing $f_1(\cdot)$ and $f_2(\cdot)$.

We will therefore tackle the problem in two steps. First, we will seek a systematic scale-space decomposition of \mathcal{M}_1 and \mathcal{M}_2 that maps the elements in 3D models to correlated sub-features across a coarse to fine hierarchy of 3D features. Next, we will align the two scale-space feature decompositions, so their hierarchical representations can be directly compared using a hierarchical matching framework. Using these procedure, the problem of measuring similarity between \mathcal{M}_1 and \mathcal{M}_2 will be reduced to that of computing a mapping \mathfrak{F} between the corresponding scale-space representations.

3.1 Decomposition Algorithm

We are given a 3D model \mathcal{M} in polyhedral representation (in our experiments we used models in VRML format). Before we can proceed with scale-space decomposition of model \mathcal{M} , we must choose a suitable metric to capture the affinity structure of \mathcal{M} , i.e., we must define a distance between any two 3D points in \mathcal{M} . Let $\{p_1, \dots, p_n\}$ denote the set of points in 3D model \mathcal{M} . We will say that \mathcal{D} is a metric function for \mathcal{M} if, for any three points $p_1, p_2, p_3 \in \mathcal{M}$, $\mathcal{D}(p_1, p_2) = \mathcal{D}(p_2, p_1) \geq 0$, $\mathcal{D}(p_i, p_i) = 0$, and $\mathcal{D}(p_1, p_3) \leq \mathcal{D}(p_1, p_2) + \mathcal{D}(p_2, p_3)$. In general, there are many ways to define metric distances on a 3D model. One of the best-known metric functions is the shortest-path metric $\delta(\cdot, \cdot)$ (geodesic distance) on the triangulation of \mathcal{M} with respect to points $\{p_1, \dots, p_n\}$, i.e., $\mathcal{D}(p, q) = \delta(p, q)$, the shortest path distance between p and q for all $p, q \in \mathcal{M}$ (see Figure 1). Observe that by construction the matrix $\mathcal{D}_{\mathcal{M}} = [\mathcal{D}(p_i, p_j)]_{n \times n}$ is symmetric, and the i^{th} row (or column) in \mathcal{D} is an n -dimensional vector, characterizing the metric structure of point p_i in model \mathcal{M} .

The problem of decomposing model \mathcal{M} into k most significant features $\mathcal{M}_1, \dots, \mathcal{M}_k$ is closely related to k -dimensional subspace clustering (k -DSC). In k -DSC, we are given a set of points $p_1, \dots, p_n \in \mathbb{R}^n$, and we would like to find a k -dimensional subspace \mathcal{S} that minimizes the quantity:

$$\sqrt{\sum_{1 \leq i \leq n} d(p_i, \mathcal{S})},$$

where $d(p_i, \mathcal{S})$ corresponds to smallest distance between point p_i and any member of \mathcal{S} . In practice, if \mathcal{S} is given, then the con-

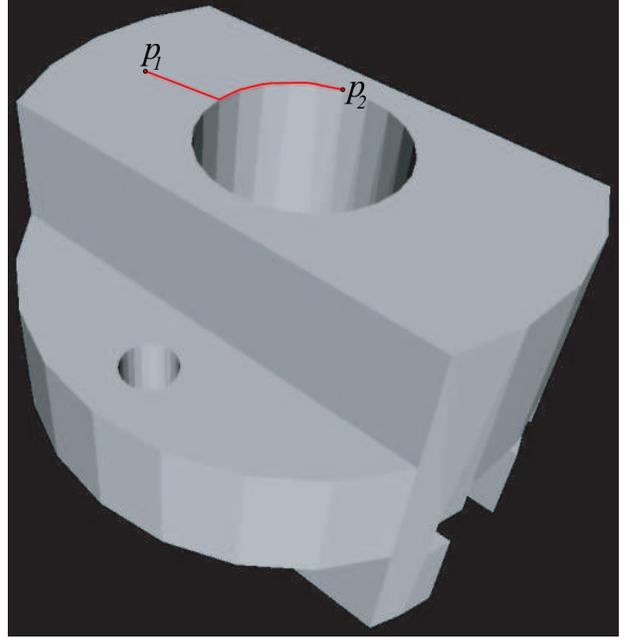


Figure 1: Distance Metric for two points of a GOOD-COUPLING.

struction of $\mathcal{M}_1, \dots, \mathcal{M}_k$ is fairly straightforward. Let $\{c_1, \dots, c_k\}$ denote the k basis vectors of \mathcal{S} , then p_i will belong to feature \mathcal{M}_j iff

$$d(p_i, \mathcal{S}) = \frac{p_i \cdot c_j}{\|p_i\| \cdot \|c_j\|}, \quad (1)$$

where $p_i \cdot c_j$ is the inner product of vectors p_i and c_j , and $\|\cdot\|$ is the length function. In other words, p_i will belong to feature vector \mathcal{M}_j iff the angle between vector p_i and basis vector c_j is the smallest among all other basis vectors. To construct the subspace \mathcal{S} , the optimal solution of k -DSC, we will use the techniques commonly known as singular value decomposition (SVD) clustering. First, observe that the symmetric matrix $\mathcal{D} \in \mathbb{R}^{n \times n}$ has an SVD-decomposition of the form

$$\mathcal{D} = U \Sigma V^T, \quad (2)$$

where $U, V \in \mathbb{R}^{n \times n}$ are orthogonal matrices and

$$\Sigma = \text{Diag}(\sigma_1, \sigma_2, \dots, \sigma_n), \quad (3)$$

with $\sigma_1 \geq \sigma_2 \geq \dots \geq \sigma_n \geq 0$. Let us define the order k compression matrix $\mathcal{D}^{(k)}$ for \mathcal{D} as:

$$\mathcal{D}^{(k)} = U \text{Diag}(\sigma_1, \dots, \sigma_k, 0, \dots, 0) V^T, \quad (4)$$

then

THEOREM 1. [Eckart-Young].

$$\|\mathcal{D} - \mathcal{D}^{(k)}\|_2 = \min_{\text{rank}(H)=k} \|\mathcal{D} - H\|_2. \quad (5)$$

That is matrix $\mathcal{D}^{(k)}$ is the best approximation to \mathcal{D} among all matrices of rank k . In fact, this result can be generalized to many other norms including *Forbenius* norm:

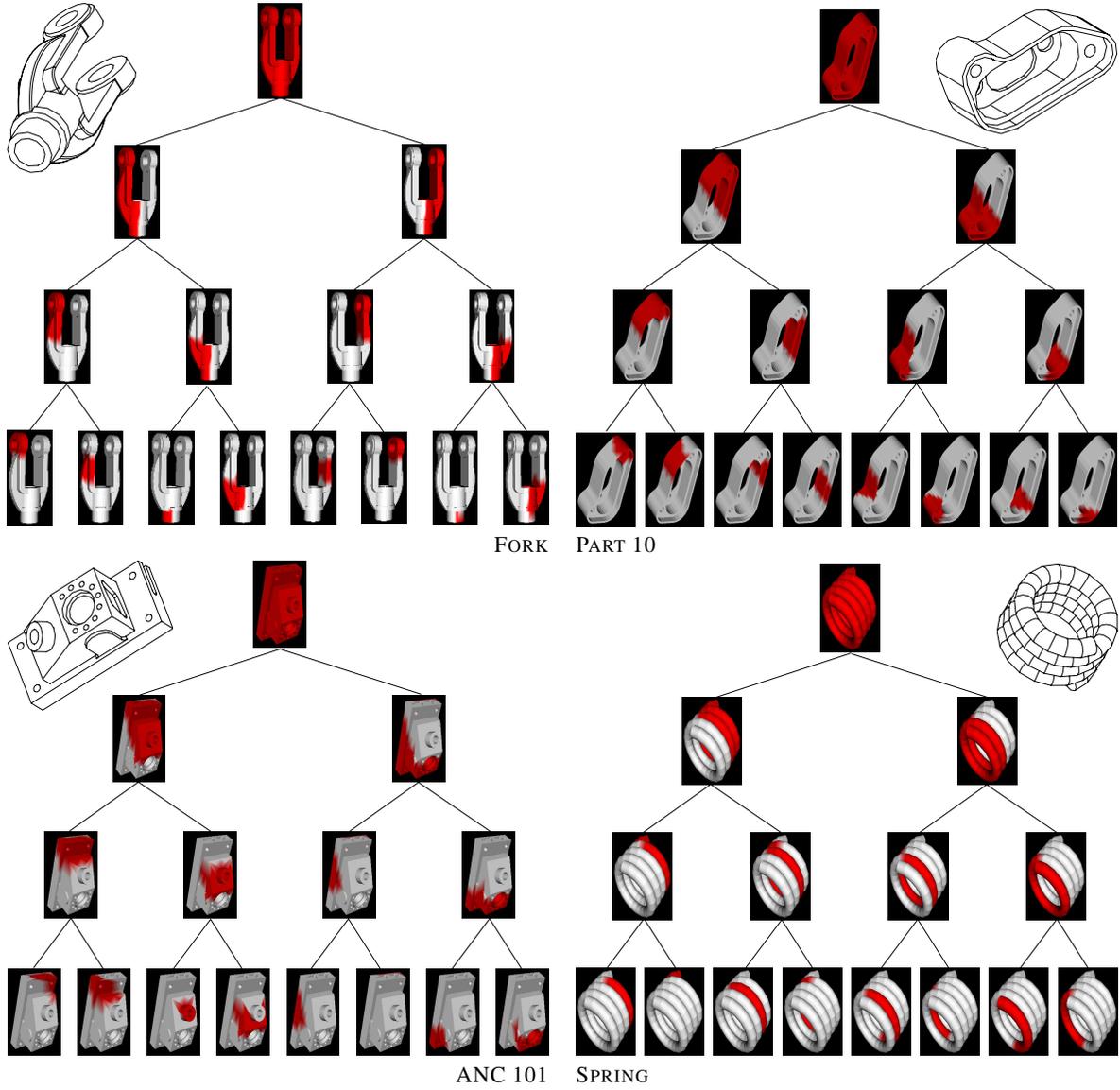


Figure 2: Scale-space bisection of the sample models. Binary trees were obtained by recursively applying FEATURE-DECOMPOSITION(\mathcal{M}, k). Red-colored regions of child nodes represent partitions that result in the bisection of the parent.

COROLLARY 2. For $A \in \mathbb{R}^{n \times n}$, let

$$\|A\|_F = \left(\sum_{i,j} A_{i,j}^2 \right)^{1/2}, \quad (6)$$

then

$$\|\mathcal{D} - \mathcal{D}^{(k)}\|_F = \min_{\text{rank}(H)=k} \|\mathcal{D} - H\|_F. \quad (7)$$

Next, assume \mathcal{S} is the range of matrix $\mathcal{D}^{(k)}$ and let c_j , for $1 \leq j \leq k$, denote the j^{th} column of $\mathcal{D}^{(k)}$. Let $\mathcal{S}' \neq \mathcal{S}$ be any k -dimensional subspace of \mathbb{R}^n . For every $p_i \in \mathcal{M}$ let $q_i \in \mathcal{S}'$ be the closest point in \mathcal{S}' to p_i . Define $\mathcal{Q} \in \mathbb{R}^{n \times n}$ with i^{th} column equal

to q_i . Clearly, $\text{rank} \mathcal{Q} \leq k$. Using Corollary 2 we have:

$$\begin{aligned} \sum_{i=1}^n d(p_i, \mathcal{S}')^2 &= \sum_{i=1}^n d(p_i, q_i)^2 \\ &= \|\mathcal{D} - \mathcal{Q}\|_F^2 \\ &\geq \|\mathcal{D} - \mathcal{D}^{(k)}\|_F^2 \\ &= \sum_{i=1}^n d(p_i, c_i)^2 \\ &\geq \sum_{i=1}^n d(p_i, \mathcal{S})^2. \end{aligned}$$

Consequently;

PROPOSITION 3. . The set $\mathcal{S} = \text{range}(\mathcal{D}^{(k)})$ is the optimal solution to k -DSC problem.

Algorithm 1 FEATURE-DECOMPOSITION(\mathcal{M}, k)

- 1: Construct the distance matrix $\mathcal{D} \in \mathbb{R}^{n \times n}$.
 - 2: Compute the SVD decomposition $\mathcal{D} = U\Sigma V^T$, with $\Sigma = \text{Diag}(\sigma_1, \sigma_2, \dots, \sigma_n)$.
 - 3: Compute the order k compression matrix $\mathcal{D}^{(k)} = U\text{Diag}(\sigma_1, \dots, \sigma_k, 0, \dots, 0)V^T$.
 - 4: Let c_j denote the j^{th} column of $\mathcal{D}^{(k)}$, for $j = 1, \dots, k$, and form sub-feature \mathcal{M}_j as the union of points $p_i \in \mathcal{M}$ with $d(p_i, \mathcal{S}) = d(p_i, c_j)$.
 - 5: Return the set $\{\mathcal{M}_1, \dots, \mathcal{M}_k\}$.
-

Algorithm 1 summarizes one phase of scale-space decomposition of \mathcal{M} into its k most significant features, $\mathcal{M}_1, \dots, \mathcal{M}_k$. Algorithm 1 returns partitioning of \mathcal{M} by placing each point p_i in \mathcal{M} to one of the partitions \mathcal{M}_j , such that the angle between vector p_i and basis vector c_j corresponding to the partition \mathcal{M}_j is the smallest. The results of scale-space decomposition for several 3D models using the recursive application of FEATURE-DECOMPOSITION(\mathcal{M}, k), for $k = 2$ is presented in Figure 2.

The bottleneck of Algorithm 1 is computation of SVD decomposition. Normally, it takes $O(n^3)$ for $n \times n$ matrix. Polyhedral representation of a model provides us with the graph that is 2D manifold. If we consider only neighboring vertices in the construction of the distance matrix \mathcal{D} , the number of non-zero entries in \mathcal{D} would be at most $6n$ (since the model is a planar graph). Computing SVD decomposition for sparse matrices is much faster and takes $O(mn) + O(mM(n))$ [11]. Where m is the maximum number of matrix-vector computations required and $M(n)$ is the cost of matrix-vector computations of the form $\mathcal{D}x$. Since M is a planner map and \mathcal{D} is a sparse matrix, $M(n) = O(n)$ and $m = O(n)$.

4. SCALE-SPACE MATCHING

The scale-space decomposition of a 3D model \mathcal{M} will give raise to a rooted undirected tree $T_{\mathcal{M}} = (V, E)$ in a natural way. The vertex set of this graph corresponds to the set of features produced by recursive application of Algorithm 1 to model \mathcal{M} and the edges of $T_{\mathcal{M}}$ will capture the decomposition relation between a feature and all its sub-features. Figure 2 represent several examples of degree-2 trees corresponding to 3D models. Using this construction the problem of comparing two 3D models \mathcal{M}_1 and \mathcal{M}_2 can be reformulated as computing a matching among the corresponding rooted trees T_1 and T_2 .

Given 3D models \mathcal{M}_1 and \mathcal{M}_2 , and their corresponding scale-space trees (SST's) T_1 and T_2 , we will propose a matching algorithm based on the dynamic programming framework proposed by Zhang and Shasha [31]. Intuitively, the similarity between two features $u \in \mathcal{M}_1$ and $v \in \mathcal{M}_2$ is closely related to similarity of sub-features forming u and v . Specifically, the similarity of features u and v should be measured in terms of the similarity of subtrees $T_1(u)$ and $T_2(v)$ rooted at u and v .

The cost of matching $T_1(u)$ and $T_2(v)$ can be characterized as:

$$C(T_1(u), T_2(v)) = C(F_1(u), F_2(v)) + \gamma(u, v). \quad (8)$$

That is, the cost of matching trees rooted at features u and v is equal to the distance between the two features u and v ($\gamma(u, v)$) plus the cost of matching the forests $F_1(u)$ and $F_2(v)$, obtained from $T_1(u)$ and $T_2(v)$, after removing u and v , respectively. In order to deal with degenerate case, $u = \emptyset$ (and/or $v = \emptyset$) we will use:

$$C(T_1(u), \emptyset) = \gamma(u, \emptyset) \quad (9)$$

$$C(F_1(u), \emptyset) = \sum_y \gamma(y, \emptyset), \quad (10)$$

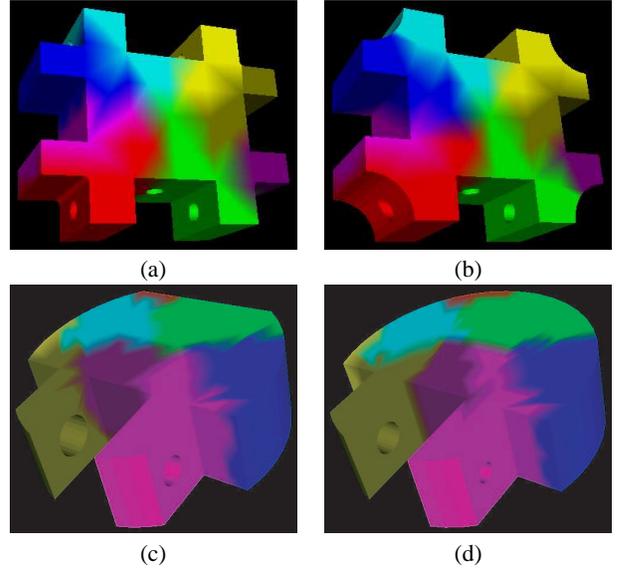


Figure 3: Results of matching between two GOODPARTS (a, b) and two SWIVELS (c, d), with matched regions having similar colors. Most significant features are obtained for each model using FEATURE-DECOMPOSITION(\mathcal{M}, k) and are used to construct binary trees T_1 and T_2 . Match-Models(T_1, T_2) then generates a set of matched features which have been colored similarly for this figure.

where the sum in (10) is over the roots of all trees in $F_1(u)$. Observe that for two features $u \in T_1$ and $v \in T_2$ be matched, at some level some of their sub-features in $F_1(u)$ and $F_2(v)$ should have been matched. This phenomenon is captured in our formulation of $C(T_1(u), T_2(v))$ in (8), using term $C(F_1(u), F_2(v))$. To compute the cost of matching the two forests $F_1(u)$ and $F_2(v)$, we need to compute a maximum similarity matching among the roots of trees in these two forests. To this end, we will form a complete bipartite graph among the sub-features forming the roots of $F_1(u)$ and $F_2(v)$. Let $x \in F_1(u)$ and $y \in F_2(v)$ denote two such vertices, we will associate the following as the cost of matching x and y in aforementioned bipartite graph:

$$w(x, y) = C(x, \emptyset) + C(\emptyset, y) + C(T_1(x), T_2(y)). \quad (11)$$

Observe that the term $C(T_1(x), T_2(y))$ is the basis of the recursive dynamic-programming framework. The chain of recursive calls will terminate at the primitive features forming the leaves of T_1 and T_2 . Consequently, the cost of matching the forests $F_1(u)$ and $F_2(v)$ can be restated as

$$C(F_1(u), F_2(v)) = \frac{1}{2} \left(\sum_x C(x, \emptyset) + \sum_y C(\emptyset, y) \right) + \sum_{(x,y) \in \mathfrak{P}(u,v)} w(x, y). \quad (12)$$

The first two sums in (12) run over the roots of trees in $F_1(u)$ and $F_2(v)$, respectively, and the third sum runs over all matched pairs in bipartite matching of sub-features of u and v , $\mathfrak{P}(u, v)$.

To state our recursive algorithm we need to specify the cost function $\gamma(u, v)$ in 8. In our formulation of $\gamma(\cdot, \cdot)$ function we will use the notion of topological similarity introduced by Hilaga et. al.

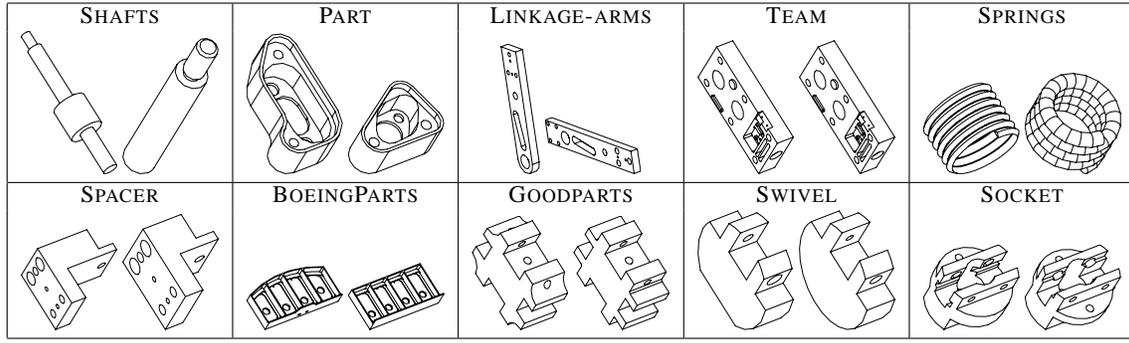


Figure 4: Sample views from 10 Groups of 3D models in our database.

in [15]. In fact we will set

$$\gamma(u, v) = e^{-sim(u, v)}, \quad (13)$$

where $sim(u, v)$ is a convex combination of the form:

$$sim(u, v) = w \cdot \min(\alpha(u), \alpha(v)) + (1 - w) \cdot \min(\ell(u), \ell(v)),$$

for $0 \leq w \leq 1$, with functions $\alpha(\cdot)$ and $\ell(\cdot)$ representing the ratios of the area and the length of the corresponding features in the whole 3D model, respectively (for further details on the description of $\alpha(\cdot)$ and $\ell(\cdot)$ see [15]).

Finally, we can state the matching algorithm for two scale-space decompositions T_1 and T_2 corresponding to 3D models \mathcal{M}_1 and \mathcal{M}_2 . The result of applying Algorithm 2 to two GOODPARTS parts is represented in Figure 3.

Algorithm 2 MATCH-MODELS(T_1, T_2)

- 1: $C(\emptyset, \emptyset) \leftarrow 0$.
- 2: $\forall u \in T_1$ COMPUTE $C(T_1(u), \emptyset)$ and $C(F_1(u), \emptyset)$ using (9) and (10).
- 3: $\forall v \in T_2$ COMPUTE $C(T_2(v), \emptyset)$ and $C(F_2(v), \emptyset)$ using (9) and (10).
- 4: $\forall u \in T_1$ and $\forall v \in T_2$ do
 COMPUTE $C(F_1(u), F_2(v))$ as in (12)
 COMPUTE $C(T_1(u), T_2(v))$ as in (8)
- 5: Return the set of matched vertices:

$$\cup_{(u, v)} M(u, v),$$

and the total cost of matching T_1 and T_2 :

$$C(\text{root}(T_1), \text{root}(T_2)).$$

We use degree-2 trees in Algorithm 2. Then, according to the time bound proposed by Zhang and Shasha in [31], the running time of our algorithm is $O(n_1 n_2 \sqrt{2} \log 2)$, where n_1 and n_2 are the number of vertices in T_1 and T_2 respectively. Please note that n_1 and n_2 are much smaller than the initial sizes of \mathcal{M}_1 and \mathcal{M}_2 . Also, in order for our algorithm to meet the time bound, we pre-compute values of $\alpha(u)$ and $\ell(u) \forall u \in T$. Complexities for computations of $\alpha(u)$ and $\ell(u)$ is linear in terms of the number of points in partition u .

5. EXPERIMENTS

To demonstrate our approach to scale-space matching, we performed a set of matching experiments using a database of 3D mod-

els. For a given 3D model, its scale-space feature decomposition is first represented by a rooted, vertex-labeled tree, in which nodes represent *features* (obtained by recursive application of Algorithm 1) and edges capture the relationships between a feature and its sub-features. We will assume that each point u in this tree is labeled by two attributes, values of the functions $\alpha(u)$ and $\ell(u)$, defined in Section 4. These attributes were used in the Algorithm 2 to compute the distances between pairs of nodes.

Models with many features and parts lead to trees with many nodes. To reduce the size of the tree, we will impose a fixed bound on the maximum number k used in the Algorithm 1 for decomposition purposes. In practice, one can optimize on parameter k via an independent study of individual models. For the experiments, we compute the scale-space tree of every model using seven layers of decomposition (depth of the tree) with branching parameter $k = 2$. This procedure results in a database of hierarchical trees, each representing a 3D model.

We tested our scale-space matching algorithm on a database of 40 3D solid models. The database consists of 10 different group of objects. A set of representative 2D views of 3D models is shown in Figure 4. These models cover several non-trivial classes of mechanical artifacts and include parts from several industry and government sources. These models are available for download from the National Design Repository at URL <http://www.designrepository.org/SM03>.

To test our approach on the resulting database, we have computed distances between every 3D model in the database and each of the remaining database entries (the distance between a view and itself is always zero). These object distances are summarized in Figure 5. The magnitudes of the distances are denoted by shades of gray, with black and white representing the smallest and largest distances, respectively.

Inter-group distances, shown along the main diagonal, are very close to zero. Also, distances between models from different groups are significantly larger than inter-group distances.

Based on the overall matching statistics, we observed that in almost 15% of the experiments, the closest match selected by our algorithm was not a member of the same group. We expect that with a more appropriate $\gamma(\cdot, \cdot)$ function, ensuring that for each model there exists a similar inter-group model, this error rate would decrease significantly. Finally, it should be noted that both the feature decomposition and matching procedures can accommodate sampling noise. This is due to the fact that the structure of trees corresponding to model are unaffected by sampling noise.

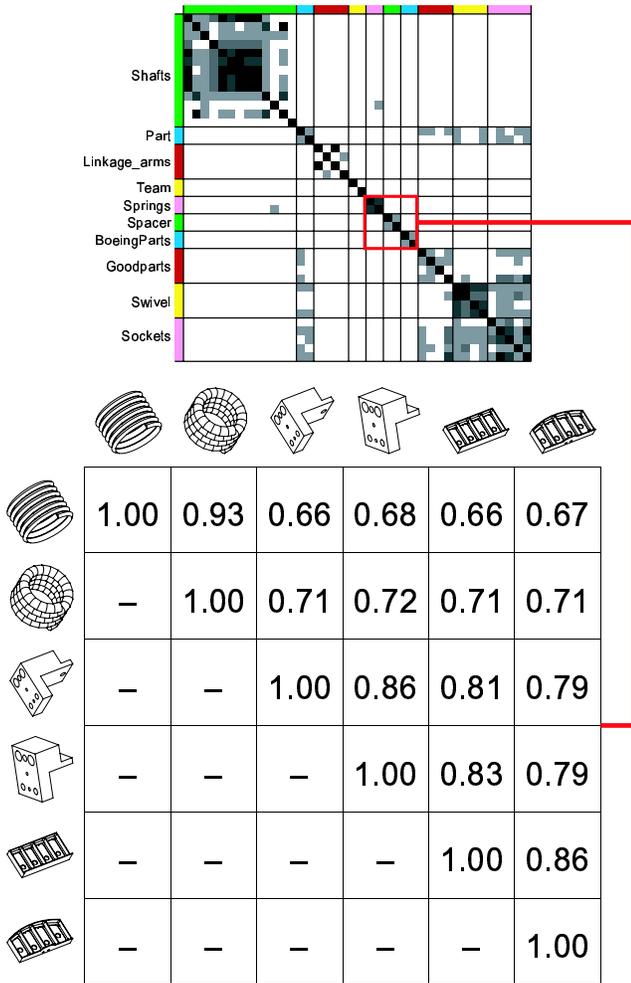


Figure 5: Distance matrix of results for the matching process. Darker regions correspond to closer matches. Actual values for matches of a subset of 6 models is shown with pictures of the models. Most of the high-valued matches occur inside the groups, although, there are several cross-group pairs that return high similarity values.

6. CONCLUDING REMARKS

We have presented a computationally efficient approach to hierarchical matching of 3D models. The approach is based on a combination of scale-space feature decomposition of 3D models, rooted tree representation of these feature decompositions, and dynamic programming matching of vertex-labeled graphs. Due to the strengths of these components, our approach is able to establish robust correspondences in the presence of sampling noise. In a series of experiments we have demonstrated the performance of our approach.

This research differs significantly from previous work on 3D model matching. The notion of feature presented here is highly tuned to the efficient identification of shape and topological categories. What we have demonstrated is that, in the context of 3D solid models of mechanical parts, the scale-space decomposition technique produces a near-exact approximation of traditional CAD features without the issues presented by traditional feature recog-

inition and mapping. This enabled us to perform highly accurate matching of 3D solid models of mechanical parts which preserved meaningful engineering classifications.

Our work is in its preliminary stages, and we plan to extend its scope in several directions: 1) to create automated recursive decomposition that would allow us to stop decomposition process when all significant features are extracted; 2) to improve the complexity of our matching algorithm; 3) improve the feature similarity metric $\gamma(\cdot, \cdot)$, to account for geometric point distribution in each feature; 4) explore techniques to add increased engineering and manufacturing data into the matching; and 5) to exploit the possibility of using the structural parameters of tree representation as signatures for indexing purposes.

At the moment, we are working on creating technique that would allow us to determine when all significant components are extracted and our decomposition algorithm should terminate. Let \mathcal{M} be the original model's point set; \mathcal{M}_1 be some partition from \mathcal{M} ; \mathcal{M}_2 and \mathcal{M}_3 denote the partitions of \mathcal{M}_1 after bisection. We define a potential function for partition \mathcal{M}_i as:

$$f(\mathcal{M}_i) = \frac{\text{Outer}(\mathcal{M}_i)}{\text{Total}(\mathcal{M}_i)},$$

where

$$\text{Outer}(\mathcal{M}_i) = \sum_{u \in \mathcal{M}_i} \sum_{v \in \mathcal{M} - \mathcal{M}_i} \mathcal{D}(u, v),$$

and

$$\text{Total}(\mathcal{M}_i) = \sum_{u \in \mathcal{M}_i} \sum_{v \in \mathcal{M}} \mathcal{D}(u, v).$$

We will continue decomposition of \mathcal{M}_1 into \mathcal{M}_2 and \mathcal{M}_3 if $f(\mathcal{M}_2) + f(\mathcal{M}_3) > f(\mathcal{M}_1)$. Currently, we are running experiments using these potential functions to control the bisection process.

Acknowledgments

This work was supported in part by National Science Foundation (NSF) CAREER Award CISE/IIS-9733545, NSF Information Technology Research Award ITR-0219176, and Office of Naval Research Grant N00014-01-1-0618. Additional support has been provided by Honeywell FM&T, Bentley Systems and Lockheed Martin, Naval Electronics and Surveillance Systems.

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 National Science Foundation or the other supporting government and corporate organizations.

7. REFERENCES

- [1] The digital michelangelo project. <http://graphics.stanford.edu/projects/mich/>.
- [2] A. L. Ames. Production ready feature recognition based automatic group technology part coding. In J. Rossignac and J. Turner, editors, *Symposium on Solid Modeling Foundations and CAD/CAM Applications*, pages 161–169, New York, NY 10036, USA, Austin, TX, June 1991. ACM SIGGRAPH, ACM Press.
- [3] A. Bond and R. Jain. The formal definition and automatic extraction of group technology codes. In *ASME Design Technical Conferences, Computers in Engineering Conference*, pages 537–542, August 1988.
- [4] V. Cicirello and W. Regli. Machining feature-based comparisons of mechanical parts. In *International*

- Conference on Shape Modeling and Applications*, pages 176–187. ACM SIGGRAPH, the Computer Graphics Society and EUROGRAPHICS, IEEE Computer Society Press, Genova, Italy, May 7-11 2001.
- [5] V. Cicirello and W. C. Regli. Resolving non-uniqueness in design feature histories. In D. Anderson and W. Bronsvort, editors, *Fifth Symposium on Solid Modeling and Applications*, New York, NY, USA, June 8-11 1999. ACM, ACM Press. Ann Arbor, MI.
- [6] V. A. Cicirello. Intelligent retrieval of solid models. Master's thesis, Drexel University, Geometric and Intelligent Computing Laboratory, Department of Mathematics and Computer Science, Philadelphia, PA 19104, June 1999 1999.
- [7] C. M. Cyr and B. B. Kimia. 3d object recognition using shape similarity-based aspect graph. In *Proceedings of the Eighth International Conference On Computer Vision (ICCV-01)*, pages 254–261, 2001.
- [8] M. Elad, A. Tal, and S. Ar. Content based retrieval of vrml objects - an iterative and interactive approach. In *The Sixth Eurographics Workshop in Multimedia*, pages 97–108, 2001.
- [9] A. Elinson, D. S. Nau, and W. C. Regli. Feature-based similarity assessment of solid models. In C. Hoffman and W. Bronsvort, editors, *Fourth Symposium on Solid Modeling and Applications*, pages 297–310, New York, NY, USA, May 14-16 1997. ACM, ACM Press. Atlanta, GA.
- [10] S. Gold and A. Rangarajan. A graduated assignment algorithm for graph matching. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 18(4):377–388, 1996.
- [11] G. H. Golub and C. F. van Loan. *Matrix Computations*. John Hopkins Press, 1989.
- [12] I. Ham, D. Marion, and J. Rubinovich. Developing a group technology coding and classification scheme. *Industrial Engineering*, 18(7):90–97, 1986.
- [13] J.-H. Han, W. C. Regli, and M. J. Pratt. Algorithms for feature recognition from solid models: A status report. *IEEE Transactions on Robotics and Automation*, 16(6):782–796, December 2000.
- [14] M. Henderson and S. Musti. Automated group technology part coding from a three-dimensional cad database. *Journal of Engineering for Industry*, 110(3):278–287, 1988.
- [15] M. Hilaga, Y. Shinagawa, T. Kohmura, and T. L. Kunii. Topology matching for fully automatic similarity estimation of 3d shapes. In *SIGGRAPH*, pages 203 – 212, New York, NY, USA, August 2001. ACM, ACM Press.
- [16] W. Kim and A. C. Kak. 3D object recognition using bipartite matching embedded in discrete relaxation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 13(3):224–251, 1991.
- [17] B. Luo and E.R.Hancock. Structural matching using the em algorithm and singular value decomposition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 23:1120–1136, 2001.
- [18] R. Osada, T. Funkhouser, B. Chazelle, and D. Dobkin. Matching 3d models with shape distributions. In 154-166, editor, *International Conference on Shape Modeling and Applications*. ACM SIGGRAPH, the Computer Graphics Society and EUROGRAPHICS, IEEE Computer Society Press, Genova, Italy, May 7-11 2001.
- [19] M. Pelillo, K. Siddiqi, and S. Zucker. Matching hierarchical structures using association graphs. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 21(11):1105–1120, November 1999.
- [20] M. M. Ramesh, D. Yip-Hoi, and D. Dutta. A decomposition methodology for machining feature extraction. In *ASME Design Engineering Technical Conferences, Computers in Engineering Conference*, New York, NY, USA, September 10-13, Baltimore, Maryland 2000. American Association of Mechanical Engineers, ASME Press. DETC2000/CIE-14645.
- [21] W. C. Regli and V. Cicirello. Managing digital libraries for computer-aided design. *Computer Aided Design*, 32(2):119–132, February 2000. Special Issue on *CAD After 2000*. Mohsen Rezayat, Guest Editor.
- [22] J. Shah, D. Anderson, Y. S. Kim, , and S. Joshi. A discourse on geometric feature recognition from cad models. *ASME Transactions, the Journal of Computer and Information Science in Engineering*, 1(1):41–51, March 2001.
- [23] J. J. Shah and A. Bhatnagar. Group technology classification from feature-based geometric models. *Manufacturing Review*, 2(3):204–213, 1989.
- [24] L. G. Shapiro and R. M. Haralick. Structural descriptions and inexact matching. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 3:504–519, 1981.
- [25] A. Shokoufandeh, S. Dickinson, C. Jönsson, L. Bretzner, and T. Lindeberg. On the representation and matching of qualitative shape at multiple scales. In *Proceedings, 7th European Conference on Computer Vision*, volume 3, pages 759–775, 2002.
- [26] K. Siddiqi, A. Shokoufandeh, S. Dickinson, and S. Zucker. Shock graphs and shape matching. *International Journal of Computer Vision*, 30:1–24, 1999.
- [27] C. S. Snead. *Group Technology: Foundations for Competitive Manufacturing*. Van Nostrand Reinhold, New York, 1989.
- [28] T.-L. Sun, C.-J. Su, R. J. Mayer, and R. A. Wysk. Shape similarity assessment of mechanical parts based on solid models. In R. Gadh, editor, *ASME Design for Manufacturing Conference, Symposium on Computer Integrated Concurrent Design*, pages 953–962. ASME, Boston, MA. September 17-21. 1995.
- [29] W. B. Thompson, R. F. Riesenfeld, and J. C. Owen. Determining the similarity of geometric models. In *Proceedings of the ARPA Image Understanding Workshop*, February 1996.
- [30] R. C. Veltkmap. Shape matching: Similarity measures and algorithms. In 188-197, editor, *International Conference on Shape Modeling and Applications*. ACM SIGGRAPH, the Computer Graphics Society and EUROGRAPHICS, IEEE Computer Society Press, Genova, Italy, May 7-11 2001.
- [31] J. T. L. Wang, B. A. Shapiro, D. Shasha, K. Zhang, and K. M. Currey. An algorithm for finding the largest approximately common substructures of two trees. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 20:889–895, 1998.
- [32] W.B.Thompson, J. Owen, H. de St. Germain, S. Stark Jr., and T. Henderson. Feature-based reverse engineering of mechanical parts. *IEEE Transactions on Robotics and Automation*, 12(1):57–66, February 1999.