Local Feature Extraction and Matching Partial Objects

Dmitriy Bespalov†  William C. Regli†‡  Ali Shokoufandeh†

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

March 20, 2006

Abstract

A primary shortcoming of existing techniques for 3D model matching is the reliance on global information of model’s structure. Models are matched in their entirety, depending on overall topology and geometry information. A current open challenge is how to perform partial matching. Partial matching is important for finding similarities across part models with different global shape properties and for segmentation and matching of data acquired from 3D scanners.

This paper presents a Scale-Space feature extraction technique based on recursive decomposition of polyhedral surfaces into surface patches. The experimental results presented in this paper suggest that this technique can potentially be used to perform matching based on local model structure. In our previous work, Scale-Space decomposition has been successfully used to extract features from mechanical artifacts. Scale-Space techniques can be parameterized to generate decompositions that correspond to manufacturing, assembly or surface features relevant to mechanical design. One application of these technique is to support matching and content-based retrieval of solid models.

This paper shows how a Scale-Space technique can extract features that are invariant with respect to the global structure of the model as well as small perturbations that 3D laser scanning process introduce. In order to accomplish this, we introduce a new distance function defined on triangles instead of points. Believe this technique offers a new way to control the feature decomposition process, which results in extraction of features that are more meaningful from an engineering view point.

The new technique is computationally practical for use in indexing large models. Examples are provided that demonstrate effective feature extraction on 3D laser scanned models. In addition, a simple sub-graph isomorphism algorithm was used to show that the feature adjacency graphs obtained through feature extraction, are meaningful descriptors of 3D CAD objects.

All of the data used in the experiments for this work is freely available at: http://www.designrepository.org/datasets/.

1 Introduction

In order to perform content-based indexing and retrieval of 3D objects, each model must be converted into some collection of features. Previous research on model matching and retrieval has drawn on feature definitions from mechanical design, computer graphics and computer vision literature. Many of these
feature-based techniques ultimately use vertex-labeled graphs, whose nodes represent 3D features (or their abstractions) and whose edges represent spatial relations or constraints, between the features. Retrieval and matching is done using some variation of graph matching to assign a numerical value describing the distance between two models.

It is common in engineering communities for the term feature to be used to refer to machining features (i.e., holes, pockets, slots) or other local geometric or topological characteristics of interest, depending on the domain (i.e., assembly surfaces, molding lines, etc). In the context of this work, feature will be used as an intrinsic property of the 3D shape which may encompass local geometry and topology. Depending on the choice of function to parameterize the Scale-Space decomposition, these local features could correspond to design or manufacturing operations, machining features or assembly surfaces, etc. The notion of features in this paper draws from the computer vision literature [1]; hence, the features are designed for object classification.

There are numerous surveys of feature recognition techniques for CAD [2, 3]; and similarity assessment of 3D models using feature extraction has been addressed by several efforts [4, 5, 6]. These techniques assume the exact representation, as is obtained from a CAD system (i.e., a 3D, watertight boundary representation). However, these representations are proprietary, and their internals vary from system to system. Feature-based descriptions of models also vary by system. Hence, CAD search tools that can perform semantically effective searches using “the lowest common denominator” (e.g., shape) representation are widely applicable.

Matching 3D shape representations has been widely studied in graphics [7], computer vision [8] and engineering [9]. When shape representations are used for CAD data, there are two major shortcomings with existing work. First, the current generation of matching techniques have difficulty handling the approximate representations (i.e., polyhedral mesh, point cloud, etc) that are needed to find sub-patterns in objects or handle data created by 3D scanners. With a few notable exceptions, most researchers assume watertight VRML or shape models. Second, and more importantly, the current generation of search techniques almost exclusively focus on gross or overall shape. In the context of CAD, local features and feature patterns contribute considerably to manufacturing cost, selection of manufacturing processes, producibility and functional parameters of 3D objects. Many objects with similar gross shape can have vastly different functions, costs or manufacturing process specifications.

**Relationship to prior art.** This paper builds on the work in [10], where it was shown how Scale-Space decompositions could be used to extract features from 3D models in a polyhedral representation. It develops a parameterizable feature decomposition method that can be tuned to extract local feature configurations of engineering significance. Lastly, it introduces the problem of partial matching in the context of acquired data and presents an end-to-end methodology for evaluation.

The approach put forth in this paper is motivated by several open problems in 3D shape matching and indexing of CAD models for useful engineering purposes:

**Parameterizable Decompositions:** Scale-Space decomposition promises to decompose a 3D model into structurally relevant components automatically. These decompositions can be parameterized with a measure function, resulting in different components and features. This allows us to perform efficient comparisons (using well studied tree-matching algorithms) of 3D models in terms of the similarity of underlying features [10]. Different measure functions can be created to tailor decompositions toward feature sets tuned to answer specific questions (i.e., cost, manufacturing process, shape, etc).

**Performance Consistency on Scanned Data:** Most existing work on shape and solid matching assumes an ideal world with watertight models and no inaccuracies or perturbations (normally intro-
duced through scanning). The reality is that objects may not be perfect, this is especially true when trying to examine 3D models acquired by laser scanning or other means (i.e., image, inspection probes, etc). In this context, one needs to be able to compare the noisy acquired data to other noisy data or to the exact geometry data that exists in a database of CAD models. The Scale-Space technique has shown consistency in its performance on scanned data, both with synthetic perturbations (simulation of scanning process) as well as with respect to the actual inaccuracy introduced through laser scanning. Hence, models can be effectively matched across representations.

**Partial Matching**: Partial matching is a major open problem in 3D indexing and matching. It manifests itself in several ways. Most evident is that acquired data is rarely complete. For example, occlusion prevents scanners from getting interior points of holes and other features. In addition, obtaining a “complete” scan is time consuming, requiring manual re-positioning of artifacts on the scanning apparatus and (quite often) manual registration of the point set data acquired by these scans. In the most basic case, the scanned data may consist of only one “view” of the model—resulting in a set of points on the surfaces of the object and not a 3D shape.

**From Scanned Point Cloud to Database Query**: The ability to handle partial data, as well as to perform consistently on scanned data, are essential in a system that can go from scanned data input directly to a database query with minimal human intervention. With even the best of present technology, it is difficult to get complete watertight solids from scanned data that precisely match the scanned artifact. In the case of CAD objects, most of which have high genus and many occluded surfaces, obtaining a complete scan that evenly samples points over the surfaces is simply impossible. Hence, matching and query mechanisms must be able to operate from limited information, i.e., point data or portions of surfaces.

**Basis for Solution by Many-to-Many Matching**: Many-to-Many matching aligns the corresponding decomposition from one medium (e.g., the native CAD object) with that of other media (e.g., scanned data) and similar, but slightly different, CAD objects. The decomposition process presented in this paper is consistent across these different media types; however, the exact boundaries of the segmentations may vary depending on the quality of the data, perturbations due to 3D laser scanning or differences in the underlying geometric representation. This creates a many-to-many matching problem in which subsets of segments from one object must be paired to subsets of the segments resulting from the decomposition of the other object.

### 2 Related Work

The work in the paper draws on concepts from several areas of computer science and engineering. We review some of this background below.

**2.1 Object Recognition and Matching**

This paper uses the term *feature* to refer to an intrinsic property of the 3D shape which may encompass local geometry and topology related to design or manufacturing operations, but may not have direct correspondence to any explicit manufacturing features. In this sense the notion of a *feature* draws from the computer vision literature [1].
The computer vision research community has typically viewed shape matching as a problem in 2D [11, 12, 13, 14]. These efforts address a different aspect of the general geometric/solid model matching problem—one in which the main technical challenge is the construction of the models to be matched from image data obtained by cameras.

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. While a complete survey of this area is beyond the scope of this paper, we review several notable efforts. Thompson et al. [15, 16] reverse engineered designs by generating surfaces and machining feature information from range data collected from machined parts. Jain et al. [17] indexed CAD data based on the creation of “feature vectors” from 2D images. Sipe, Casasent and Talukder [18, 19] used acquired 2D image data to correlate real machined parts with CAD models and performed classification and pose estimation. Scale-Space decomposition is very popular in Computer Vision for extracting spatially coherent features. Most of the work in this community has focused on the Scale-Space features of 2D images using wavelets or Gaussian filters [1, 20].

Once objects are recognized, they can be segmented, decomposed and matched. Matching is frequently accomplished by encoding objects and their decompositions as a graph and doing analysis across different graph structures to identify similarity. Graphs and their generalizations are among the most common and best studied combinatorial structures in computer science, due in large part to the number of areas of research in which they are applicable. Due to space constraints, we cite only a few examples of how they are being applied to 3D object recognition and matching. Nayar and Murase extended this work to general 3D objects where a dense set of views was acquired for each object [21]. Eigen-based representations have been widely used in many areas for information retrieval and matching as they offer greater potential for generic shape description and matching. In an attempt to index into a database of graphs, Sossa and Horaud use a small subset of the coefficients of the $d_2$-polynomial corresponding to the Laplacian matrix associated with a graph [22], while a spectral graph decomposition was reported by Sengupta and Boyer for the partitioning of a database of 3D models, in which nodes in a graph represent 3D surface patches [23].

Graph matching has a long history in pattern recognition. Shapiro and Haralick’s use of weighted graphs for the structural description of objects was among the first in the vision community [24]. Eshera and Fu [25] used attributed relation graphs to describe parametric information as the basis of a general image understanding system to find inexact matches. Recently, Pelillo et al. [26] introduced a matching algorithm which extends the detection of maximum cliques in association graphs to hierarchically organized tree structures. Tirthapura et al. present an alternative use of shock graphs for shape matching [27]. The edit distance approach for finding matching in rooted trees has been studied by Zhang, Wang, and Shasha [28]. Their dynamic programming approach for degree-2 distance, when applied to unordered trees, is a restricted form of the constrained distance previously reported in [29].

### 2.2 Matching 3D Objects

With the ready availability of 3D models from graphics programs and CAD systems, there has been a substantial amount of activity on 3D object recognition and matching in the past 20 years. This body of relevant work is too large to survey in detail in this paper. Interested readers are referred to several recent survey papers [8, 9, 7].

#### 2.2.1 Comparing Shape Models.

Shape-based approaches usually work on a low-level point cloud, mesh or polyhedral model data, such as that produced by digital animation tools or acquired by 3D range scanners. Approaches based on faceted
representations include that of Osada et al. [30], which creates an abstraction of the 3D model as a probability distribution of samples from a shape function acting on the model. Hilaga et al. [31] present a method for matching 3D topological models using multi-resolution Reeb graphs. A variant on this is proposed in [32]. A current trend, being pursued by several groups, is the use of different types of shape descriptors (harmonics, Zernike, etc.) to capture shape invariants [33, 34, 35, 36].

The Princeton 3D shape database [37] contains mainly models from 3D graphics and rendering; none of these models are specifically engineering, solid modeling or mechanical CAD oriented.

In general, however, shape matching-based approaches only operate on the gross-shape of a single part and do not operate directly on 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.

2.2.2 Comparing Solid Models.

Unlike shape models, for which only approximate geometry and topology is available, solid models produced by CAD systems are represented by precise boundary representations. When comparing solid models of 3D CAD data, there are two basic types of approaches for content-based matching and retrieval: (1) feature-based techniques and (2) shape-based techniques. The feature-based techniques [38, 2, 39, 3, 40], going back at least as far as 1980 [41], extract engineering features (machining features, form features, etc.) from a solid model of a mechanical part for use in database storage, automated group technology (GT) part coding, etc. The shape-based techniques are more recent, owing to research contributions from computational geometry, vision and computer graphics. These techniques leverage the ready availability of 3D models on the Internet.

Feature-Based Approaches. Historically Group Technology (GT) coding was the way to index of parts and part families [42]. This facilitated process planning and cell-based manufacturing by imposing a classification scheme (a human-assigned alphanumeric string) to individual machined parts. While there have been a number of attempts to automate the generation of GT codes [43, 44, 45], transition to commercial practice has been limited.

The idea of similarity assessment of 3D models using feature extraction techniques has been discussed in [2, 3]. These techniques assume the exact representation (i.e., boundary representation or “B-rep”) for the input models and therefore cannot be used if only an approximate representation (i.e., polyhedral mesh) is available. This is a major shortcoming, especially in designing an archival system, where one may require partial and inexact matching.

There has been recent work on partial matching in the context of 3D data. For instance, Funkhouser et al. successfully employed shape-based search in [46] for 3D models with parts of those models matching a query. In addition, Cornea et al. used approach for many-to-many matching of skeletons of 3D objects in [47] to perform retrieval on those objects.

Elinson et al. [48] used feature-based reasoning for retrieval of solid models for use in variant process planning. Cicirello and Regli [49, 5, 4] examined how to develop graph-based data structures and create heuristic similarity measures among artifacts based on manufacturing features. McWherter et al. [50] integrated these ideas with database techniques to enable indexing and clustering of CAD models based on shape and engineering properties. Other work from the engineering community includes techniques for automatic detection of part families [51] and topological similarity assessment of polyhedral models [52].
Shape-Based Approaches. Comparing CAD models based on their boundary representations can be difficult due to variability in the underlying feature-based representations. Additional complications are created by differences among the boundary representations used by systems (i.e., some may use all NURBS, some may use a mix of surface types, etc). Using a shape-based approach on voxels, meshes or polyhedral models generated from native CAD representations is one way of reducing these problems.

The 3D-Base Project [53, 54] used CAD models in a voxel representation, which were then used to perform comparisons using geometric moments and other features. The recent work by the authors covers several areas including shape classification, Scale-Space decomposition and classification learning [55, 56, 57, 58].

Work out of Purdue [59, 60, 61] has improved on the voxel methods of [53, 54], augmenting them with skeletal structures akin to medial axes or shock graphs. The main accomplishment of the Purdue group is getting these shape-only techniques in a system for query by example.

3 Sub-Space Clustering and Scale-Space Decomposition

During the last decade, hierarchical segmentation has become recognized as a 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. The 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. rather than the CAD literature. Namely, given an object \( M \), we are interested in partitioning \( M \) into \( k \) features \( M_1,\ldots,M_k \) with \( M_i \cap M_j = \emptyset \), for \( 1 \leq i < j \leq k \), and \( M = \bigcup_i M_i \) subject to maximization of some coherence measure, \( f(M_i) \), defined on the 3D elements forming each \( M_i \). At a finer scale, each feature \( M_i \) will be decomposed into \( j = 1,\ldots,k \) sub-features, subject to the maximization of some coherence measures.

There are three central components in the aforementioned process: the number of components at each scale of decomposition, \( k \); the feature coherence function \( f(\cdot) \); and the number of scales of decomposition process, \( \ell \). In most pattern recognition applications, \( k \) is a control parameter. If models \( M \) and \( M' \) are topologically similar, the \( k \) major components at every scale should also be similar. The coherence function \( f(M') \) will assign an overall metric to the quality of 3D elements participating in the construction of feature \( M' \). Finally, the depth of decomposition will be controlled depending on the quality of a feature in comparison to all its sub-features. Specifically, assume \( M' \) represents a feature at scale \( i \), and \( M_1^{i+1},\ldots,M_j^{i+1} \), for \( j \leq k \) represent its sub-features at scale \( i + 1 \). The decomposition process should proceed to scale \( i + 1 \) with respect to feature \( M' \) if and only if \( f(M'^{i+1}) \leq f(M_1^{i+1}) + f(M_2^{i+1}) + \ldots + f(M_j^{i+1}) \). This simple criteria for expansion of scale-space at every feature has its roots in information theory. It is in fact motivated by linear form similar to entropy of feature \( M' \) as opposed to its sub-features \( M_1^{i+1},\ldots,M_j^{i+1} \). In the end, a set of the leaf nodes in a decomposition tree would correspond to the final features of a given model.

3.1 Distance Function

A 3D model \( M \) is given in polyhedral representation (models in VRML format were used in the experiments). We are interested in decomposing this model into \( k \) sub-features using Scale-Space decomposition.
The application of Scale-Space decomposition requires some distance function \( D(\cdot, \cdot) \) that captures the affine structure of a model \( \mathcal{M} \). The shortest-path metric \( \delta(\cdot, \cdot) \) (geodesic distance [62]) on the triangulation of \( \mathcal{M} \) with respect to points \( \{v_1, \ldots, v_n\} \), is one such function. In this case, the distance function \( D(u, v) = \delta(u, v) \) would be the shortest path distance on the triangulated surface between \( u \) and \( v \) for all \( u, v \in \mathcal{M} \).

In our previous work [10] metric \( \delta(\cdot, \cdot) \) was successfully used for Scale-Space decomposition. The experimental results showed that the measure \( \delta(\cdot, \cdot) \) successfully captures affine structure of the model \( \mathcal{M} \) and produces meaningful decompositions. The problem with such a distance measure is that it captures global information of the model. Even small perturbations of the model \( \mathcal{M} \) may cause the distance function \( D(\cdot, \cdot) \) to vary significantly, which in its turn, changes extracted features. Further, using geodesic distance as distance measure for decomposition does not tolerate small perturbations (i.e., laser-scanned data) very well. Figure 1 illustrates several distance functions that could be used in Scale-Space decomposition. Specifically, Figure 1(a) shows a geodesic distance function (weight of the shortest path between points) between points \( p_1 \) and \( p_2 \); Figure 1(b) illustrates angular shortest path (weight of the shortest path computed using an angular measure) distance between faces \( t_1 \) and \( t_2 \); Figure 1(c) shows maximum angle on angular shortest path distance function (described below) between faces \( t_1 \) and \( t_2 \).

Due to the above shortcomings of the geodesic distance measure that a new distance function is introduced for use in the Scale-Space decomposition process. The new distance function is computed with respect to the triangular faces of the model \( \mathcal{M} \) \( \{t_1, \ldots, t_n\} \). Here and in the rest of the paper \( n \) denotes the number of triangles in the model. The angular shortest path between two triangular faces \( t_i \) and \( t_j \) is defined to be the shortest path on the surface of the model which is computed in terms of angular difference between faces.

Figure 1(c) shows a maximum angle on angular shortest path between the faces \( t_i \) and \( t_j \). Specifically, let \( t_i \leadsto t_j \) denote the angular shortest path \( (t_i, t_m, t_i, \ldots, t_j) \) between faces \( t_i \) and \( t_j \). And let \( t_m \to t_i \in t_i \leadsto t_j \) denote two adjacent triangular faces \( t_m \) and \( t_i \) on the angular shortest path \( t_i \leadsto t_j \). Then, the distance function used in this work is defined as

\[
D(t_i, t_j) = \max_{t_m \to t_i \in t_i \leadsto t_j} \angle(t_m, t_i).
\] (1)

Intuitively, distance \( D(t_i, t_j) \) is the maximum angle between adjacent faces on the angular shortest path between \( t_i \) and \( t_j \). The rationale behind such measure is to quantify the smoothness of the surface – small angle between adjacent faces correspond to smooth surface.
Observe that by construction the matrix $D_M = [D(t_i, t_j)]_{n \times n}$ is symmetric. Also note that distance measure $D$ is not a metric function, but it captures the geometric structure of the model $M$. It is important to stress that such angular distance measure has the same properties as geodesic distance function used in the previous work. As a result, introduction of the new distance measure for decomposition does not violate any statements made in [10] and all of the theorems would still hold.

### 3.2 Decomposition Algorithm

Let $v_i$ be the $i^{th}$ row (or column) in $D$. Then $v_i$ is an $n$-dimensional vector characterizing the distance structure of face $t_i$ in model $M$. The problem of decomposing model $M$ into its $k$ most significant features $M_1, ..., M_k$ is closely related to $k$-dimensional subspace clustering ($k$-DSC). $k$-DSC gives a set of distance
vectors \( v_1, \ldots, v_n \), where the objective is to find a \( k \)-dimensional subspace \( S \) that minimizes the quantity:

\[
\sqrt{\sum_{1 \leq i \leq n} d(v_i, S)^2},
\]

(2)

where \( d(v_i, S) \) corresponds to the smallest distance between \( v_i \) and any member of \( S \). In practice, if \( S \) is given, then \( M_1, \ldots, M_k \) can be computed using the principle components \( \{ c_1, \ldots, c_k \} \) of the \( k \)-dimensional subspace \( S \) [63]. Observe that these \( k \) vectors will also form a basis for \( S \). Specifically, \( t_i \) will belong to the feature \( M_j \) if the angle between \( v_i \) and \( c_j \) is the smallest among all basis vectors in \( \{ c_1, \ldots, c_k \} \), i.e., , the triangular face \( t_i \) that corresponds to the vector \( v_i \) will belong to the feature vector \( M_j \) iff the angle between \( v_i \) and \( c_j \) vectors is the smallest compared to all other basis vectors.

Singular value decomposition (SVD) clustering [63] is used to construct the subspace \( S \), which is the optimal solution of \( k \)-DSC. First, observe that the symmetric matrix \( D \in \mathbb{R}^{n \times n} \) has a SVD-decomposition of the form

\[
D = U\Sigma V^T,
\]

(3)

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

\[
\Sigma = \text{Diag}(\sigma_1, \sigma_2, \ldots, \sigma_n),
\]

(4)

with \( \sigma_1 \geq \sigma_2 \geq \ldots \geq \sigma_{n'} > 0, \sigma_{n'+1} = \ldots = \sigma_n = 0, n' \leq n \). Let us define the order \( k \) compression matrix \( D^{(k)} \) of \( D \), for \( k \leq n' \) as:

\[
D^{(k)} = U\text{Diag}(\sigma_1, \ldots, \sigma_k, 0, \ldots, 0)V^T.
\]

(5)

Then,

**Theorem 1** [follows from Eckart-Young Theorem [64]].

\[
\|D - D^{(k)}\|_2 = \min_{\text{rank}(H)=k} \|D - H\|_2.
\]

(6)

This states that matrix \( D^{(k)} \) is the best approximation to \( D \) among all matrices of rank \( k \). In fact, this result can be generalized to many other norms, including *Forbenius* norm. [10] showed that the set \( S = \text{range}(D^{(k)}) \) (\( S \) is the range of matrix \( D^{(k)} \), the subspace spanned by the columns of matrix \( D^{(k)} \)) is the optimal solution to \( k \)-DSC problem.

Algorithm 1 summarizes one phase of the Scale-Space decomposition of \( M \) into its \( k \) most significant features, \( M_1, \ldots, M_k \). Algorithm 1 returns the partitioning of \( M \) by placing each face \( t_i \) in \( M \) into one of the partitions \( M_j \), such that the angle between vector \( t_i \) and basis vector \( c_j \) corresponding to the partition \( M_j \) is minimized. Figure 2 shows three decomposition trees of the model – using geodesic distance, angular shortest path and maximum angle on angular shortest path measures. Note that the presented decomposition trees are not full and the leaf nodes of the trees may not correspond to the actual final features extracted from this model.
**Algorithm 1** \textsc{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 \(\Sigma = \text{Diag}(\sigma_1, \sigma_2, \ldots, \sigma_n)\).
3: Compute the order \(k\) compression matrix \(\mathcal{D}^{(k)} = U \text{Diag}(\sigma_1, \ldots, \sigma_k, 0, \ldots, 0)V^T\).
4: Let \(c_j\) denote the \(j\)th column of \(\mathcal{D}^{(k)}\), for \(j = 1, \ldots, k\), and form sub-feature \(\mathcal{M}_j\) as the union of faces \(t_i \in \mathcal{M}\) with \(d(t_i, S) = d(t_i, c_j)\).
5: Return the set \(\{\mathcal{M}_1, \ldots, \mathcal{M}_k\}\).

---

(a) Decomposition tree is obtained using \textsc{Feature-Decomposition}(\(\mathcal{M}, k\)) algorithm.

(b) Leaf nodes of the tree correspond to the features.

Figure 3: Feature extraction process.

The bottleneck of Algorithm 1 is the \(O(n^3)\) SVD decomposition, for an \(n \times n\) matrix. The polyhedral representation of a model provides a planar graph of a 2D manifold. If only neighboring vertices are considered in the construction of the distance matrix \(\mathcal{D}\), the number of non-zero entries in \(\mathcal{D}\) would be at most \(3n\) (due to planarity of the graph). Computing the SVD decomposition for sparse matrices is much faster and takes \(O(mn) + O(mM(n))\) \cite{65}. 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 \(m = O(n)\).

### 3.3 Controlling Decomposition Process

The decomposition process as presented in Section 3.1 does not allow for an explicit mechanism to stop the indefinite subdivision of a feature. Clearly, one could use a prescribed value to control the decomposition depth of the feature trees, i.e., decomposition process will be stopped when a root branch in feature decomposition tree reaches a given depth. This section provides an overview of a mechanism that will control the feature decomposition. Intuitively, the use of this control mechanism will terminate the decomposition process only when all coherent features are extracted. Figure 4 shows results of decomposition process on selected CAD models, including partial ones.
Let $\mathcal{M}$ be the original model’s face set. Assume in the decomposition process a feature $\mathcal{M}_1$ in $\mathcal{M}$ can be decomposed into sub-features $\mathcal{M}_2$ and $\mathcal{M}_3$ (e.g., without loss of generality assume that feature $\mathcal{M}_1$ is being bisected). The decomposition of the feature $\mathcal{M}_1$ into sub-features $\mathcal{M}_2$ and $\mathcal{M}_3$ is said to be significant if the angular distance between components of $\mathcal{M}_2$ and $\mathcal{M}_3$ is large. Formally, this condition could be expressed as follows:

$$\forall t_i \in \mathcal{M}_2, \ t_j \in \mathcal{M}_3 \ \exists t_m \rightarrow t_l \ni t_i \sim t_j \text{ s.t.}$$

$$t_m \in \mathcal{M}_2 \land t_l \in \mathcal{M}_3 \land \angle(t_m, t_l) = D(t_i, t_j),$$

i.e., if the angular shortest path between $t_i \in \mathcal{M}_2$ and $t_j \in \mathcal{M}_3$ contains two faces $t_m$ and $t_l$ (from $\mathcal{M}_2$ and $\mathcal{M}_3$ respectively) with large angular distance, then $\mathcal{M}_1$ should be decomposed into $\mathcal{M}_2$ and $\mathcal{M}_3$. Intuitively, if $\mathcal{M}_1$ is smooth it should not be bisected any further. On the other hand, if discrepancy between the neighboring triangles in $\mathcal{M}_1$ is significant, $\mathcal{M}_1$ should be bisected.

4 Empirical Results

The feature extraction process was performed on a number of CAD models in polyhedral representation. These models were converted from ACIS format, which is exact representation format. As a result, all of the models have nice structure (i.e., no missing faces).

In the experiments we would like to examine the qualities of features extracted using the Feature- Decomposition($\mathcal{M}, k$) algorithm. To these ends, Feature-Decomposition($\mathcal{M}, k$) is recursively applied to each model for $k = 2$. Once a decomposition tree is obtained, the last layer of the decomposition tree (leaf nodes) is said to be a set of extracted features. Note, that the union of the features (leaf nodes) is equivalent to the surface of the entire model. Refer to Figure 3 for an illustration of the feature extraction process. For illustrative purposes, only a subset of extracted features is shown in Figure 3(b); the features shown in Figure 3(a) do not correspond to the leaf nodes in Figure 3(b). The actual decomposition tree is quite large for this model.

4.1 Feature Decomposition on CAD Data

Figure 4 shows extracted features for several models. These images are presented in order to illustrate the type of features the technique can extract. Observe that each feature corresponds to a relatively smooth surface on the model. If there is a significant angular difference on the surface, then it gets decomposed into separate features. Any closed smooth surfaces (i.e., hole) are decomposed into two (i.e., hole) or more (i.e., surface is concave) features. We plan to address the problem of how to use the decomposition trees for matching in the future work.

In addition, partial data from these models was created. Each model was intersected with several planes and only a part of the model (on one side of the plane) was saved. As a result, a number of partial objects was obtained which enabled us to see how the Feature-Decomposition($\mathcal{M}, k$) algorithm performs on the models where only partial data is available. Illustrations of extracted features\footnote{Please note: for a typical CAD model Scale-Space feature decomposition with max-angle distance measure produces over 150 features. Pictures of entire feature sets were omitted for the sake of space.} could be found in Figure 4.
4.2 Feature Decomposition on Noisy Data

In order to simulate small perturbations produced from capturing the object using 3D laser scan, Gaussian noise was applied to each point of the models presented in Section 4.1. Gaussian Noise with standard deviation of 1% and 2% from the standard deviation of all points in the model was used. Then the features were extracted using FEATURE-DECOMPOSITION($\mathcal{M}, k$) algorithm. The illustrations of the extracted features can be found in Figures 5 and 6. Similar to the CAD models presented above, partial models for this dataset were generated. Note that it is possible for separate features to be assigned visually similar colors, making them appear to be the same features. The names for the CAD models were assigned by the organizations that provided the files to us. We chose to use such names for the purpose of referencing the models within this paper.

4.3 Feature Decomposition on Acquired Models

We have established that the feature extraction procedure allows to obtain relevant subsets of a model that reflect the complexity of its 3D structure. The next experiment was aimed at assessing whether the technique is capable of handling models that were obtained using a 3D digitizer – full 3D view (Figure 7(b)) and partial 3D view (Figure 7(a)) of 3D objects. Such data is known to be very noisy, often with broken connectivity and missing faces. Ideally, one would like to be able to take a single scan of a 3D CAD model, decompose it into features, and select models from the database that contain the same feature arrangements. Three CAD parts were used to create six 3D models – full and partial (one scan) for each CAD part. Once the point clouds were obtained, they were faceted, and features were extracted using the FEATURE-DECOMPOSITION($\mathcal{M}, k$) algorithm.

Figure 8 shows correspondence between extracted features for fully and partially scanned models as well as models obtained from exact representation. Note that in some cases one feature from one model (i.e., full scan) can correspond to multiple features from another model (i.e., single scan).

The performance of the technique on noisy data is certainly not as remarkable as on the CAD dataset (Section 4.1). Although, we believe that in most cases the extracted features are meaningful and reflect the structure of the models. In addition, it is clear that there are similarities between feature decompositions of fully and partially scanned models and 3D CAD models from our database. The scanned models used for this experiment are freely available at:

http://www.designrepository.org/datasets/Scanned.tar.bz
<table>
<thead>
<tr>
<th></th>
<th>Full Model</th>
<th>Partial Models</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>CIMPLEX</strong></td>
<td><img src="image" alt="CIMPLEX Full Model" /></td>
<td><img src="image" alt="CIMPLEX Part 1" /></td>
</tr>
<tr>
<td><strong>SIMPLE BOEING</strong></td>
<td><img src="image" alt="SIMPLE BOEING Full Model" /></td>
<td><img src="image" alt="SIMPLE BOEING Part 1" /></td>
</tr>
<tr>
<td><strong>PART 9</strong></td>
<td><img src="image" alt="PART 9 Full Model" /></td>
<td><img src="image" alt="PART 9 Part 1" /></td>
</tr>
<tr>
<td><strong>PART 10</strong></td>
<td><img src="image" alt="PART 10 Full Model" /></td>
<td><img src="image" alt="PART 10 Part 1" /></td>
</tr>
</tbody>
</table>

Figure 4: Extracted features from CAD models. Each extracted feature is assigned a separate color. Full models as well as partial models are presented in this figure.
<table>
<thead>
<tr>
<th></th>
<th>Full Model</th>
<th>Partial Models</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>CIMPLEX</strong></td>
<td><img src="image1.png" alt="Image" /></td>
<td><img src="image2.png" alt="Image" /></td>
</tr>
<tr>
<td></td>
<td><img src="image3.png" alt="Image" /></td>
<td><img src="image4.png" alt="Image" /></td>
</tr>
<tr>
<td><strong>Simple Boeing</strong></td>
<td><img src="image5.png" alt="Image" /></td>
<td><img src="image6.png" alt="Image" /></td>
</tr>
<tr>
<td><strong>Part 9</strong></td>
<td><img src="image7.png" alt="Image" /></td>
<td><img src="image8.png" alt="Image" /></td>
</tr>
<tr>
<td><strong>Part 10</strong></td>
<td><img src="image9.png" alt="Image" /></td>
<td><img src="image10.png" alt="Image" /></td>
</tr>
</tbody>
</table>

Figure 5: 1% Gaussian Noise. Extracted features from CAD models with Gaussian noise applied to each point of the model. Each extracted feature is assigned a separate color. Full models as well as partial models are presented in this figure.
<table>
<thead>
<tr>
<th></th>
<th>Full Model</th>
<th>Partial Models</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>CIMPLEX</strong></td>
<td><img src="image1.png" alt="Image" /></td>
<td><img src="image2.png" alt="Image" /></td>
</tr>
<tr>
<td><strong>SIMPLE BOEING</strong></td>
<td><img src="image4.png" alt="Image" /></td>
<td><img src="image5.png" alt="Image" /></td>
</tr>
<tr>
<td><strong>PART 9</strong></td>
<td><img src="image7.png" alt="Image" /></td>
<td><img src="image8.png" alt="Image" /></td>
</tr>
<tr>
<td><strong>PART 10</strong></td>
<td><img src="image10.png" alt="Image" /></td>
<td><img src="image11.png" alt="Image" /></td>
</tr>
</tbody>
</table>

Figure 6: 2% Gaussian Noise. Extracted features from CAD models with Gaussian noise applied to each point of the model. Each extracted feature is assigned a separate color. Full models as well as partial models are presented in this figure.
Figure 7: Illustration of acquisition process. Case (a) is substantially more difficult than Case (b).

Figure 8: Correspondence of two selected extracted features for the full scan models, single scan models and models obtained from exact representation (ACIS model). Note the many-to-many case for the Socket (top right example).
4.4 Matching Experiment

In order to test whether features extracted using Scale-Space technique with max-angle distance measure can be used in retrieval of solid models, the following matching experiment was conducted. Three retrieval techniques were used in evaluation: Reeb Graph, original Scale-Space and max-angle Scale-Space (described in this paper).

Reeb Graph based technique introduced by Hilaga et al. in [31]. This technique was designed for shape models and is based on identifying certain regions of a model (i.e., feature) and combining them into hierarchical graph structure. Then, a graph matching technique is used to obtain similarity values for corresponding models. This approach performs very well if overall gross shape of the models are similar.

Original Scale-Space technique was introduced in [10]. This approach use geodesic distance for distance function. By “max-angle Scale-Space” we refer to the feature extraction approach described in this paper. Although This approach does not have matching technique specifically designed for it, a simple sub-graph isomorphism approach was employed to assess similarity of constructed feature graphs (see Section 4.4.1 for more information).

All three technique were evaluated on one dataset of solid models which is described in Section 4.4.2. In order to illustrate results of the experiment, precision-recall plots were constructed. Refer to Section 4.4.3 for more information on precision-recall measures.

4.4.1 Matching Approach

For simplicity, a variation on a classical sub-graph isomorphism algorithm is used to asses similarity of the feature adjacency graphs: leaf nodes (features) in decomposition tree become nodes in the graph, edges indicate adjacency of the features on the surface of the model. Hill-climbing algorithm with random restarts was used in the implementation of the sub-graph isomorphism technique. Largest Common Subgraph algorithm described in [6, 4, 5, 66] was used in the implementation of the sub-graph isomorphism technique. This well-known approach to graph matching was used to simply show that the feature graphs constructed using max-angle distance measure carry relevant information about the structure of the models and could be used to assess similarities between 3D CAD models. In reality, more sophisticated graph matching algorithm should be used to yield even higher accuracy in matching. As the experimental results suggest, such graph matching algorithm should be able to allow many-to-many matching of the nodes within the feature graphs.

4.4.2 Dataset

The dataset used in this experiment consists of seven groups of models. Seventy (70) models are hand classified by their role in mechanical systems. For instance, brackets are overhanging members that project from a structure and are usually designed to support a vertical load or to strengthen an angle. Linkage arms are motion transferring components from the spectrometer assembly. Nuts, Screws, and Blots are commonly used fasteners. Figure 9 shows a sample of this dataset, and table 1 shows a brief summary of this dataset, it is available at:


4.4.3 Precision-Recall Measure

The performance of various retrieval techniques could be evaluated by the $k$-nearest neighbor classification ($k$NN), and conventional recall and precision measures for evaluating information retrieval systems. The
Table 1: Statistics of Functional Dataset

<table>
<thead>
<tr>
<th></th>
<th>#Models</th>
<th>Avg. #Faces</th>
<th>Avg. #Polygons</th>
</tr>
</thead>
<tbody>
<tr>
<td>Brackets</td>
<td>9</td>
<td>45</td>
<td>911</td>
</tr>
<tr>
<td>Gears</td>
<td>12</td>
<td>169</td>
<td>4045</td>
</tr>
<tr>
<td>Housings</td>
<td>6</td>
<td>218</td>
<td>5141</td>
</tr>
<tr>
<td>Linkage Arms</td>
<td>13</td>
<td>30</td>
<td>1282</td>
</tr>
<tr>
<td>Nuts</td>
<td>7</td>
<td>8</td>
<td>518</td>
</tr>
<tr>
<td>Screws and Blots</td>
<td>18</td>
<td>15</td>
<td>431</td>
</tr>
<tr>
<td>Springs</td>
<td>5</td>
<td>161</td>
<td>7933</td>
</tr>
<tr>
<td><strong>Total</strong></td>
<td>70</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th></th>
<th>Avg. SAT size</th>
<th>Avg. STEP size</th>
<th>Avg. VRML size</th>
</tr>
</thead>
<tbody>
<tr>
<td>Brackets</td>
<td>56KB</td>
<td>100KB</td>
<td>41KB</td>
</tr>
<tr>
<td>Gears</td>
<td>458KB</td>
<td>525KB</td>
<td>191KB</td>
</tr>
<tr>
<td>Housings</td>
<td>300KB</td>
<td>450KB</td>
<td>250KB</td>
</tr>
<tr>
<td>Linkage Arms</td>
<td>62KB</td>
<td>100KB</td>
<td>57KB</td>
</tr>
<tr>
<td>Nuts</td>
<td>13KB</td>
<td>19KB</td>
<td>31KB</td>
</tr>
<tr>
<td>Screws and Blots</td>
<td>18KB</td>
<td>30KB</td>
<td>21KB</td>
</tr>
<tr>
<td>Springs</td>
<td>620KB</td>
<td>960KB</td>
<td>440KB</td>
</tr>
</tbody>
</table>

Figure 9: Examples of the models from the functional classification dataset.

Recall and precision values are computed at different thresholds values of parameter $k$ using the following formulas:

\[
\text{recall} = \frac{\text{Retrieved and Relevant models}}{\text{Relevant models}}
\]

\[
\text{precision} = \frac{\text{Retrieved and Relevant models}}{\text{Retrieved models}}
\]

The $k$NN classification labels a query model with the categories of its $k$ closest neighbors, where $k$ is
the threshold for classification. The numbers of labeled categories potentially increase and decrease with respect to $k$.

Under this experimental setting, the factors of recall and precision computation become:

- **Relevant models**: The number of models that fall in to same category as the query model.
- **Retrieved models**: The number of models returned by a query.
- **Retrieved and Relevant models**: The number of models returned and that fell into the same category as the query model.

Recall and precision values were first computed per model at different $k$ values. For each $k$, the arithmetic mean of the recall and precision across all models in a dataset was used as a representative value. To illustrate the results, precision is plotted against recall on different datasets and comparison techniques.

Ideally, a retrieval system should retrieve as many relevant models as possible, both high precision as well as high recall are desirable. A precision-recall graph plots precision against recall. It shows the trade-off between precision and recall. Trying to increase recall, typically, introduces more irrelevant models into the retrieved set, thereby reducing precision. Rightward and upward precision-recall curves indicates a better performance.

### 4.4.4 Matching Results

The precision-recall graphs for the dataset can be found in Figure 10. Random retrieval technique was simulated by choosing all models randomly. It appears that the Reeb Graph technique performs relatively better than both Scale-Space approaches, while Scale-Space technique with max-angle distance function out-performs original Scale-Space for this dataset. The results of this experiment show that the use of Scale-Space feature extraction technique with max-angle distance function results in meaningful decomposition that could potentially be used for matching of 3D CAD data.
4.5 Fidelity Experiments

Due to the approximated nature of shape models (polyhedral representation), the fidelity of shape models depends on the granularity of the faceting process. In order to measure the effects of fidelity variations on the feature extraction technique, a set of the following experiments was performed. All of the models used for the following experiments are freely available at:


4.5.1 Variable Fidelity Dataset

A subset of models from Functional Classification Dataset (Figure 9) was chosen for the fidelity experiments. A total of 40 CAD models classified by part families were used. Each of them was faceted by ACIS for three instances with different normal tolerances (50, 15, 5), resulting in 120 models. Figure 11 shows the mesh of an example model under different fidelity settings. Lowering the normal tolerance will cause the faceting component to approximate a parametric surface with more polygons, hence increasing the fidelity of the resulting shape model. Ideally, a robust retrieval system should be indifferent to fidelity variations of meshes. Table 2 shows a brief summary of this dataset.

![Variable Fidelity Dataset](image)

(a) Low Fidelity, Normal Tolerance = 50  
(b) Normal Fidelity, Normal Tolerance = 15  
(c) High Fidelity, Normal Tolerance = 5

Figure 11: Variable Fidelity Dataset. Three Copies of the Same Model under Different Fidelity settings.

<table>
<thead>
<tr>
<th></th>
<th># Models</th>
<th>Avg. # Polygons</th>
<th>Avg VRML size</th>
</tr>
</thead>
<tbody>
<tr>
<td>High</td>
<td>40</td>
<td>18416</td>
<td>850KB</td>
</tr>
<tr>
<td>Normal</td>
<td>40</td>
<td>5908</td>
<td>275KB</td>
</tr>
<tr>
<td>Low</td>
<td>40</td>
<td>2699</td>
<td>117KB</td>
</tr>
<tr>
<td>Total</td>
<td>120</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Table 2: Statistics of Variable Fidelity Dataset.
4.5.2 Fidelity Experiment Using Original Scale-Space Technique

The precision-recall performance of the original Scale-Space technique (i.e., with a geodesic distance function) was measured on the Variable Fidelity Dataset. A precision-recall plot was constructed for each fidelity setting (Low, Normal, High). Figure 12(a) presents precision-recall plots for various fidelity settings using original Scale-Space technique.

The experimental results show that the retrieval performance of the original Scale-Space technique improved as the mesh fidelity increased. This behavior is largely due to the fact that geodesic distance measure is affected by the mesh fidelity. As the mesh fidelity increases, the distance measure is calculated more precisely and, as a result, improves “quality” of extracted features. This results in more accurate retrieval of CAD data.
4.5.3 Fidelity Experiment Using Scale-Space Technique with Max-Angle Distance Function

Just like in the experiment for the original Scale-Space, precision-recall plots for each fidelity setting was constructed using Scale-Space retrieval technique with max-angle distance function. In this example, the simple sub-graph isomorphism algorithm is used for matching. Figure 12(b) presents precision-recall plots for various fidelity settings using Scale-Space technique with max-angle distance function.

The results suggest that Scale-Space technique with max-angle distance function is relatively invariant to the mesh fidelity. This is due to the nature of the distance measure used in feature extraction. Since distance measure is angle based, the extracted features are preserved across various fidelity settings. Indeed, increasing the mesh fidelity normally affect smooth or flat surfaces, which are already being segmented as separate features. Furthermore, various fidelity settings do not affect right or rather large angles between surfaces on the mesh models, which also preserve feature extracted using Scale-Space technique with max-angle distance function.

4.6 Partial Matching

The last experiment was designed to test whether a simple sub-graph isomorphism can yield satisfactory retrieval results on scanned and partial data. A total of nine models were used as query models for this experiment. The query models correspond to three actual physical parts (displayed in Figure 13). For each physical part, three various 3D models were obtained — full scan and single scan models, and partial models in ACIS format (exact representation). The partial models obtained from exact representation were created by removing some features from the full models, such that they would resemble single scan models of the corresponding physical parts.

The database contained models from dataset used in matching experiment from Section 4.4.2 and full CAD models that correspond to the query models. All of the objects in the database were converted from the ACIS format into polyhedral representation. For each query model, \(k\) closest neighbors from the database were retrieved. The experimental results for \(k = 5\) is presented in Figure 13.

From Figure 13, one may conclude that for only one physical part (the one in the middle), the desirable (correct) model was among five returned models. Although, for this physical part, for every variation of the models, the desirable model was among the returned ones. Furthermore, if \(k\) is set to 10, then 5 (out of 9) queries returned correct model. Increasing \(k\) to 15, results in 7 (out of 9) correct queries, while \(k = 20\) return desirable model among returned ones for 9 (out of 9) queries.

The unsatisfactory performance of the max-angle Scale-Space decomposition technique with matching using sub-graph isomorphism can be explained using the following. The model that resulted in correct queries for \(k = 5\) (correct model was among returned ones) has very topologically distinctive feature graph. As a result, sub-graph isomorphism algorithm was able to pick correct model among all of the models in the database. Further, partial data result in partial feature graphs and, as a result, the distance between a partial model and a full model becomes large enough to diminish the retrieval capabilities which the technique showed in Section 4.4. Lastly, when Scale-Space decomposition is applied on scanned data, the perturbations may contribute to the extracted features (i.e., perturbations become extracted features). This issue can clearly affect performance of the sub-graph isomorphism on feature graphs.

5 Discussion

There are several areas for discussion emerging from this paper. One need only look at the results in Figure 13 to realize that the performance of retrieval techniques in the presence of acquired data is more chal-

22
<table>
<thead>
<tr>
<th>Query</th>
<th>Returned Models</th>
</tr>
</thead>
<tbody>
<tr>
<td>Full Scan</td>
<td><img src="image1" alt="Images" /></td>
</tr>
<tr>
<td>Single Scan</td>
<td><img src="image2" alt="Images" /></td>
</tr>
<tr>
<td>Partial CAD</td>
<td><img src="image3" alt="Images" /></td>
</tr>
<tr>
<td>Full Scan</td>
<td><img src="image4" alt="Images" /></td>
</tr>
<tr>
<td>Single Scan</td>
<td><img src="image5" alt="Images" /></td>
</tr>
<tr>
<td>Partial CAD</td>
<td><img src="image6" alt="Images" /></td>
</tr>
<tr>
<td>Full Scan</td>
<td><img src="image7" alt="Images" /></td>
</tr>
<tr>
<td>Single Scan</td>
<td><img src="image8" alt="Images" /></td>
</tr>
<tr>
<td>Partial CAD</td>
<td><img src="image9" alt="Images" /></td>
</tr>
</tbody>
</table>

Figure 13: Retrieval experiment on partial and scanned data. For each query, five closest models were retrieved from the database.

...lenging that more self-contained search and matching problems. Also, one would hope that the precision-recall performance found in Figures 10 and 12 could be improved. It is the belief of the authors that limitations in performance are primarily due to the difficulty of the problem and not due to inherent limitations in the Scale-Space techniques. Hence, given that the problem of partial matching from noisy, acquired data is very hard, we outline some of the lessons learned from these experiments and describe some of the key
challenge problems that are emerging.

**Handling Acquired Models.** When one scans a 3D model of a CAD artifact, or any artifact (Figure 7), many irregularities are introduced. First of all, scanning produces a point cloud; and turning such point clouds into reliable, water-tight, meshes is a very active research problem. Further, “noise” can be introduced in a number of ways. First, from the scanner accuracy, where points may be sampled in such a way as to deviate from the nominal geometry. Second, there will be gaps and voids, such as found on the inside surfaces of holes that are either occluded or are parallel to the scanning laser’s beam.

There are really two ways to approach this problem. First, one could design techniques based on the assumption that the model is completed though some automated process or using vast amounts of human editing. This could result in a water-tight model; or at least one that is suitable for visualization purposes. The second approach is to design techniques to work off of the partial data (Figure 7 (a)). Clearly, the second case is the more difficult one; but also the more realistic one in a production setting.

**Efficient Matching Algorithms.** Given that input data will not be of identical quality to the data in the database, features may not get segmented the same way across models with these different underlying representations. As shown in Figure 14 gives an, features may get divided in to several features. The way to address this is to develop algorithms for *many-to-many matching*. For instance, a matching technique with such properties could potentially be derived from many-to-many matching algorithm presented in [47]. Efficiency is also of concern, if these algorithms are to be used with the National Design Repository database ² or other interactive settings.

**Similarity Measures.** Similarity measures need to be different in the presence of partial data and many-to-many feature correspondences. Previous work [10] successfully used a distance function that was based on numerical value for each pair of features. These values were based on area and Euclidean distance measurements within features. Please see Figure 15 for a sample view of two models with matched regions. Depending on the data, it is possible that exact correspondences are not possible and that even segmentations into single features may rarely correspond with each other.

**Semantically Meaningful Features.** Other possible directions for Scale-Space work are to (1) explore techniques to extract features that more closely resemble traditional CAD features (i.e., such as those found in the ISO 10303 STEP AP 224 standard); and (2) exploit the possibility of using Scale-Space features as signatures for indexing purposes.

6 Conclusions

This paper introduces a computationally practical approach, based on Scale-Space decomposition, to automatically segment 3D models in polyhedral representation into features that could be used for indexing, classification and matching. The decomposition is based on the local surface structure of a model. As a result similar features can be extracted in the presence of partial model information and noisy data. In this way, the technique has been shown to consistently segment partial 3D views, noisy geometry and the data (both partial and noisy) acquired by 3D laser range scanners.

²http://www.designrepository.org
One of the significant contributions of this work is to unite the notion of “feature” from the computer vision and graphics literature with the “features” of CAD/CAM. The specific measurement function behind the concept of features in this paper is highly tuned to the efficient identification of shape and topological categories. In one application, features obtained using our approach could be different from traditional CAD features and used to establish partial similarities between CAD models in polyhedral representation. We argue that the Scale-Space technique can be parameterized using different measurement functions, enabling it to generate a variety of useful segmentations, including those that have semantic relevance to engineering and manufacturing properties. Through experiments, locality-based feature representation was shown to have promising capabilities that, with further research, could be employed for 3D matching purposes, including partial matchings. Furthermore, our experiments indicate that the Scale-Space decomposition technique can potentially be used on 3D models (in particular, partial models) generated from 3D data acquisition devices, such as laser range scanners.

The Scale-Space approach developed and advanced in this paper creates a foundation for creating new approaches to existing problems in feature-based manufacturing. Foremost, one can argue that Scale-Space techniques can subsume all existing approaches to feature identification by parameterizing the decomposition of the surface on a model as a distance measure function. The concept of the measure function is highly generalizable, implying that all one needs to do is identify the measure function intrinsic to the class of features of interest and provide it as a parameter to the Scale-Space algorithm. Extending and enhancing the Scale-Space technique creates several research challenges. Because it is focused on local information, Scale-Space techniques have to be extended to capture features that have been subdivided through interactions. In addition, considerable work needs to be done to develop measure functions that map to well-established engineering feature sets. Alternative approach would be to propose mappings (e.g., Brep-like schema) between engineering features and surfaces extracted through Scale-Space decomposition technique with max-angle distance measure.

Ultimately we believe that Scale-Space techniques provide important new capabilities that complement existing approaches to feature identification and shape matching. With additional research, Scale-Space technique can become part of the solution to a number of important engineering problems.

Acknowledgments

This work was supported by the Library of Congress’ National Digital Information Infrastructure and Preservation Program (NDIIPP) through the National Science Foundation (NSF) DIGARCH Program Grant IIS-0456001. Additional support includes NSF CAREER Award CISE/IIS-9733545 and Office of Naval Research (ONR) Grant N00014-01-1-0618.

Any opinions, findings, 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.

References


Figure 14: Illustration of many-to-many matching. Features from single scan model (on the left, shown in red) can not match features from full scan model (on the right, shown in red) individually, while the unions of the features (shown in green) match.

Figure 15: Results of matching between GOODPART and BADPART, with matched features (leafs of decomposition trees) having similar colors. Feature trees are obtained for each model using FEATURE-DECOMPOSITION(\mathcal{M}, k) with geodesic distance function. The technique (introduced in [10]) used to match those objects relies on global model structure.


[4] Cicirello, V., and Regli, W., 2001. “Machining feature-based comparisons of mechanical parts”. In International Conference on Shape Modeling and Applications, ACM SIGGRAPH, the Computer Graph-


