Scale Space Representation for Matching of 3D Models

A Thesis
Submitted to the Faculty
of
Drexel University
by
Dmitriy Bespalov
in partial fulfillment of the
requirements for the degree
of
Master of Science in Computer Science
January 2005
Dedications
Acknowledgements
Table of Contents

List of Tables ................................................................. vi
List of Figures ................................................................. vii
Abstract ................................................................. ix
1 Introduction ............................................................... 1
  1.1 Organization .......................................................... 5
2 Background ................................................................. 6
  2.1 Related Work .......................................................... 6
    2.1.1 Object Recognition and Matching ................................. 6
    2.1.2 Matching 3D Objects .............................................. 8
  2.2 Open Problem ......................................................... 11
3 Approach ................................................................. 14
  3.1 Feature Decomposition ............................................... 14
    3.1.1 Decomposition Algorithm ....................................... 16
  3.2 Variations of Feature Decomposition ................................ 20
    3.2.1 Global Distance Measure ....................................... 23
    3.2.2 Local Distance Measure ....................................... 23
  3.3 Controlling Decomposition Process ................................. 25
    3.3.1 Controlling Decomposition Process for Local Distance Measure . 25
    3.3.2 Controlling Decomposition Process for Global Distance Measure . 26
  3.4 Scale-Space Matching ................................................ 27
    3.4.1 Matching for Global Distance Measure .......................... 28
3.4.2 Matching for Local Distance Measure ............................................. 32

4 Experimental Studies ................................................................. 34

4.1 Experimental Results for Global Scale-Space Approach ................. 35

  4.1.1 Synthetic Datasets .............................................................. 36

  4.1.2 Actual Artifacts Dataset ...................................................... 38

  4.1.3 Experimental Protocol ....................................................... 44

  4.1.4 Experimental Results ....................................................... 47

4.2 Experimental Results For Local Scale-Space Approach ................. 50

  4.2.1 Feature Decomposition on CAD Data ..................................... 53

  4.2.2 Feature Decomposition on Noisy Data ................................... 53

  4.2.3 Feature Decomposition on Acquired Models ............................ 58

  4.2.4 Matching ................................................................. 61

5 Contribution and Future Work ....................................................... 62

  5.1 Discussion and Future Work .................................................. 62

  5.2 Contribution ................................................................. 64

Bibliography ................................................................. 66
List of Tables

4.1 Statistics of the Primitive Dataset ........................................ 37
4.2 Statistics of the Minor Topological Variation Dataset ............... 38
4.3 Statistics of the Manufacturing Classification Dataset ............... 41
4.4 Statistics of Functional Dataset ........................................... 43
4.5 Statistics of the LEGO© Dataset .......................................... 44
List of Figures

2.1 The finest level of features extracted using Reeb Graph technique. Two views (front and back) of Boeing model are provided. Each feature is colored in one color. ....................................................... 12

3.1 Illustration of various distance functions. ...................................................... 20

3.2 Results of recursively applying Feature-Decomposition(ℳ, k) to a model using different distance functions ∆ for k = 2. .......................... 22

3.3 Results of matching similar models between each other with matched regions having similar colors. Most significant features are obtained for each model using Feature-Decomposition(ℳ, k) and are used to construct binary trees T1 and T2. Match-Models(T1, T2) then generates a set of matched features which have been colored similarly for this figure. ........ 29

4.1 Examples of the models from the primitive models dataset. ............... 37

4.2 Examples of the models from the minor topological variation dataset. .... 39

4.3 Examples of the models from the National Design Repository. ............. 40

4.4 Examples of the models from the manufacturing classification dataset. ... 42

4.5 Examples of the models from the functional classification dataset. ......... 43

4.6 Examples of the LEGO® dataset. .............................................................. 44

4.7 Precision-Recall graphs on synthetic datasets. (PLOTS ARE IN COLOR.) 49

4.8 Precision-Recall graphs on actual artifacts datasets. (PLOTS ARE IN COLOR.) 51

4.9 Feature extraction process. ................................................................. 52

4.10 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. ....................................................... 55
4.11 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. ................................................................. 56

4.12 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. ................................................................. 57

4.13 Illustration of acquisition process. ................................................................. 59

4.14 Correspondence of selected extracted features for the full scan models, single scan models and models obtained from exact representation (ACIS model). ................................................................. 60

4.15 Precision-Recall graphs for Functional Classification Dataset. .............. 61

5.1 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. ................................................................. 63
Abstract
Scale Space Representation for Matching of 3D Models
Dmitriy Bespalov
William C. Regli, Ph. D., Ali Shokoufandeh, Ph. D.
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 work draws from the computer vision literature [35]; hence, the features are designed for object classification.

There are numerous surveys of feature recognition techniques for CAD [20, 47]; and similarity assessment of 3D models using feature extraction has been addressed by several efforts [10, 11]. 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 [25], computer vision [62] 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.

This work develops a parameterizable feature decomposition method that can be tuned to extract local feature configurations of engineering significance. The approach put forth here 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 to perform efficient comparisons (using well studied tree-matching algorithms) of 3D models in terms of the similarity of underlying features [5]. Different measure functions can be created to tailor decompositions toward feature sets tuned to answer specific questions (i.e., cost, manufacturing process, shape, etc).

Robustness in Presence of Noise: Most existing work on shape and solid matching assumes an ideal world with watertight models and no noise. 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 under noise, both with synthetic noise as well as with respect to the actual noise in data from laser scans. 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 noise and partial 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 document is consistent across these different media types; however, the exact boundaries of the segmentations may vary depending on the quality of the data, noise 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.
1.1 Organization

The rest of this work is structured as follows. The related work and the discussion of the open problem is presented in Chapter 2. The general Scale-Space decomposition approach is introduced in Section 3.1. Section 3.2 talks about the discussion on possible variations of the approach based on the distance measure and introduces two possible variations of the distance function – one measure is defined using global information of a model while the other one is tuned to be more sensitive to the local structure of a model. Matching techniques that could be utilized when a specific distance measure is used (global or local) are described in Section 3.4. The experimental results for both approaches are presented in Chapter 4. Finally, contributions as well as discussion of the future directions of this work can be found in Chapter 5.
2. Background

2.1 Related Work

This work draws on concepts from several areas of computer science and engineering. Some of this background is reviewed below.

2.1.1 Object Recognition and Matching

The term feature is used in this document 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 [35].

The computer vision research community has typically viewed shape matching as a problem in 2D [49, 33, 65, 34]. 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 my work, the review of several notable efforts is presented. Thompson et al. [60, 64] reverse engineered designs by generating surfaces and machining feature information from range data collected from machined parts. Jain et al. [18] indexed CAD data based on the creation of “feature vectors” from 2D images. Sipe, Casasent and Talukder [58, 53] 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 [35, 52].

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 analyses 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. To be brief, only a few examples of how they are being applied to 3D object recognition and matching are presented. Nayar and Murase extended this work to general 3D objects where a dense set of views was acquired for each object [38]. 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 [55], 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 [45].

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 [50]. Eshera and Fu [16] 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. [42] 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 [56]. The edit distance approach for finding matching in rooted trees has been studied by Zhang, Wang, and Shasha [66]. Their dynamic programming approach for degree-2 distance, when applied to unordered trees, is a restricted form of the constrained distance previously reported in [67].

2.1.2 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 work. Interested readers are referred to several recent survey papers [62, 9, 25].

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. [40], 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. [21] present a method for matching 3D topological models using multi-resolution Reeb graphs. A variant on this is proposed in [61]. A current trend, being pursued by several groups, is the use of different types of shape descriptors (harmonics, Zernike, etc.) to capture shape invariants [30, 39, 29, 31].
The Princeton 3D shape database [51] that has been used in a number of these studies [21, 40] 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.

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 [48, 20, 28, 47, 1], going back at least as far as 1980 [32], extract engineering features (machining features, form features, etc.) from a solid model of a mechanical part for use in the 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 [54]. 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 [2, 46, 19], transition to commercial practice has been limited.

The idea of similarity assessment of 3D models using feature extraction techniques has
been discussed in [20, 47]. These techniques assume the exact representation (i.e. Brep)
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.

Elinson et al. [15] used feature-based reasoning for retrieval of solid models for use
in variant process planning. Cicirello and Regli [44, 11, 10] examined how to develop
graph-based data structures and create heuristic similarity measures among artifacts based
on manufacturing features. McWherter et al. [37] integrated these ideas with the database
techniques to enable indexing and clustering of CAD models based on shape and engi-
eering properties. Other work from the engineering community includes techniques for
automatic detection of part families [43] and topological similarity assessment of polyhe-
dral models [57].

**Shape-Based Approaches.** Comparing CAD models based on their boundary representa-
tions 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 [12, 14] 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 [6, 8, 7, 24].

Work out of Purdue [36, 27, 26] has improved on the voxel methods of [12, 14], 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.

2.2 Open Problem

There are some retrieval techniques that use a collection of features as a signature for a 3D polyhedral model. For instance, Hilaga et al. in [21] introduces a Reeb Graph based retrieval technique for shape models which 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 reasonably well if overall gross shape of the models are similar. Although, the approach is not suitable for retrieval of models which are composed of similar features but do not have similar gross shapes [4]. In the context of 3D CAD data retrieval, it is essential to be able to retrieve models that have similar engineering features but do not have similar gross shapes.

The reason why the Reeb Graph technique fails to successfully assess similarity of such 3D CAD data is its underlying representation of features. Figure 2.1 shows the set of features for Boeing model from the finest hierarchical level obtained using Reeb Graph technique. Clearly, these features are not quite intuitive. In addition, each feature does
Figure 2.1: The finest level of features extracted using Reeb Graph technique. Two views (front and back) of Boeing model are provided. Each feature is colored in one color.

not capture any structurally important information about the model. This work presents a Scale-Space decomposition framework that could be tuned to extract more meaningful features which could potentially improve retrieval process of 3D CAD models.

In addition to the above problem, most of the retrieval techniques that operate on models in polyhedral representation are very sensitive to the global structure of the 3D models. In other words, small changes in the shapes of models (i.e. 2 holes instead of 1, etc) result in major changes of the underlying representations. This is a significant disadvantage because it compromises use of scanned data in database retrieval. Indeed, if a physical CAD part is scanned using a 3D digitizer, the obtained model will have two problems – noise and missing features. If a retrieval technique that is sensitive to the global structure of the model is used, the representation that will be obtained for such scanned model will not likely to be similar to its CAD counterpart (similar model without noise and missing features). As a
result, this makes querying a CAD database with scanned models inapplicable. This work shows how Scale-Space decomposition framework can be extended to successfully extract meaningful features from the scanned data. Moreover, the proposed framework enables one to extract features on partial data, which is an essential step to perform partial matching.
3. Approach

This chapter introduces the feature extraction process based on Scale-Space decomposition. The type of features that can be extracted using this approach can be tuned through the distance measure used by the technique. This work provides two ways to define distance measure – Global (Section 3.2.1) and Local (Section 3.2.2) distance measures. Please note that the use of the feature extraction technique is not limited by these two distance functions and could potentially be extended by using other distance measures.

3.1 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 $\mathcal{M}$, one may be interested in partitioning $\mathcal{M}$, into $k$ features $\mathcal{M}_1, \ldots, \mathcal{M}_k$, with $\mathcal{M}_i \cap \mathcal{M}_j = \emptyset$, for $1 \leq i < j \leq k$, and
\( \mathcal{M} = \bigcup_i \mathcal{M}_i \) 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 \( j = 1, \ldots, k_i \) 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 decomposition process, \( \ell \). 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 \( \mathcal{A}_1, \ldots, \mathcal{A}_j \), 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 \( f(\mathcal{A}) \leq f(\mathcal{A}_1) + f(\mathcal{A}_2) + \ldots + f(\mathcal{A}_j) \). 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 \( \mathcal{A}_1, \ldots, \mathcal{A}_j \).

The 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, a scale-space decomposition would allow to 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 \( f_2(\cdot) \).
The problem is tackled in two steps. First, one has to identify 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, the alignment of the two scale-space feature decompositions is introduced, 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 $\Psi$ between the corresponding scale-space representations.

### 3.1.1 Decomposition Algorithm

A 3D model $\mathcal{M}$ is given in polyhedral representation (in the experiments models in VRML format were used). Before one can proceed with the scale-space decomposition of model $\mathcal{M}$, a suitable distance function to capture the affinity structure of $\mathcal{M}$ must be chosen, i.e., one must define a distance between any two 3D points in $\mathcal{M}$. Let $\{v_1, \ldots, v_n\}$ denote the set of points in the 3D model $\mathcal{M}$. $D$ is said to be a metric function for $\mathcal{M}$ if, for any three points $v_1, v_2, v_3 \in \mathcal{M}$, $D(v_1, v_2) = D(v_2, v_1) > 0$, where $v_1 \neq v_2$, $D(v_i, v_i) = 0$, and $D(v_1, v_3) \leq D(v_1, v_2) + D(v_2, v_3)$. In general, there are many ways to define metric distances on a 3D model. The discussion of various distance measures used in the implementation of the decomposition algorithm is presented in Section 3.2.

The problem of decomposing model $\mathcal{M}$ into $k$ most significant features $\mathcal{M}_1, \ldots, \mathcal{M}_k$ is closely related to $k$-dimensional subspace clustering ($k$-DSC). In $k$-DSC, a set of distance vectors $p_1, \ldots, p_n$ is given, and the objective is to find a $k$-dimensional subspace $\mathcal{S}$ that
minimizes the quantity:

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

where \( d(p_i, S) \) corresponds to the smallest distance between \( p_i \) and any member of \( S \). In practice, if \( S \) is given, then \( M_1, ..., M_k \) can be computed using the principle components \( \{c_1, ..., c_k\} \) of \( k \)-dimensional subspace \( \mathcal{S} \) [59]. Observe that, these \( k \) vectors will also form a basis for \( \mathcal{S} \). Specifically, \( p_i \) will belong to the feature \( M_j \) if the angle between \( p_i \) and \( c_j \) is the smallest among all basis vectors in \( \{c_1, ..., c_k\} \), i.e., the point \( p_i \) will belong to the feature vector \( M_j \) iff the angle between these vectors is the smallest compared to all other basis vectors.

To construct the subspace \( S \), the optimal solution of \( k \)-DSC, the technique commonly known as singular value decomposition (SVD) clustering [59] is used. First, observe that the symmetric matrix \( D \in \mathbb{R}^{n \times n} \) has a SVD-decomposition of the form

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

(3.1)

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

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

(3.2)

with \( \sigma_1 \geq \sigma_2 \geq \ldots \geq \sigma'_{n} > 0 \), \( \sigma'_{n+1} = \ldots = \sigma_{n} = 0, n' \leq n \). Define the order \( k \) compression matrix \( D^{(k)} \) of \( D \), for \( k \leq n' \) as:
\( \mathcal{D}^{(k)} = U \text{Diag}(\sigma_1, \ldots, \sigma_k, 0, \ldots, 0) V^T. \)  

(3.3)

Then,

**Theorem 1** [Eckart-Young].

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

(3.4)

That is, matrix \( D^{(k)} \) 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 *Frobenius* norm:

**Corollary 2** For \( A \in \mathbb{R}^{n \times n} \), let

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

(3.5)

then,

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

(3.6)

Next, assume \( \mathcal{I} \) is the range of matrix \( \mathcal{D}^{(k)} \) (the subspace spanned by the columns 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 \( \mathcal{I}' \neq \mathcal{I} \) 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 obtain:

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

Consequently;

**Proposition 3.** The set $\mathcal{S} = \text{range}(\mathcal{Q}^{(k)})$ 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 $\Sigma = \text{Diag}(\sigma_1, \sigma_2, \ldots, \sigma_n)$.
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 points $p_i \in \mathcal{M}$ with $d(p_i, \mathcal{S}) = d(p_i, c_j)$.
5: Return the set $\{\mathcal{M}_1, \ldots, \mathcal{M}_k\}$.

Algorithm 1 summarizes one phase of scale-space decomposition of $\mathcal{M}$ into its $k$ most significant features, $\mathcal{M}_1, \ldots, \mathcal{M}_k$. Algorithm 1 returns the partitioning of $\mathcal{M}$ by placing each point $p_i$ in $\mathcal{M}$ into 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 minimized.
The bottleneck of Algorithm 1 is the $O(n^3)$ SVD decomposition, for an $n \times n$ matrix. Polyhedral representation of a model provides with planar graph of a 2D manifold. If only neighboring vertices are considered in the construction of the distance matrix $D$, the number of non-zero entries in $D$ would be at most $6n$ (due to planarity of the graph). Computing SVD decomposition for sparse matrices is much faster and takes $O(mn) + O(mM(n))$ [17]. Where $m$ is the maximum number of matrix-vector computations required and $M(n)$ is the cost of matrix-vector computations of the form $Dx$. Since $M$ is a planner map and $D$ is a sparse matrix, $M(n) = O(n)$ and $m = O(n)$.

### 3.2 Variations of Feature Decomposition

The features that can be extracted using Algorithm 1 can significantly vary depending on the distance measure used in the process. For the implementation of the feature extraction technique two distance measures were chosen. Figure 3.1 illustrates three distance functions:

- **(a) Geodesic Distance Function** (described in Section 3.2.1)
- **(b) Angular Shortest Path**
- **(c) Max-Angle on Angular Shortest Path** (described in Section 3.2.2)

![Figure 3.1: Illustration of various distance functions.](image)
functions (two of them are used in the implementations) that could be used for feature decomposition. Figure 3.2 presents results of scale-space decomposition for a model using different distance functions.
Figure 3.2: Results of recursively applying FEATURE-DECOMPOSITION(\(\mathcal{M}, k\)) to a model using different distance functions \(\mathcal{D}\) for \(k = 2\).
3.2.1 Global Distance Measure

One of the best-known metric functions is the shortest-path metric $\delta(.,.)$ (geodesic distance) on the triangulation of $\mathcal{M}$ with respect to points $\{v_1,\ldots,v_n\}$; i.e., $\mathcal{D}(u,v) = \delta(u,v)$, the shortest path distance on the triangulated surface between $u$ and $v$ for all $u,v \in \mathcal{M}$ (see Fig. 3.1(a)). Observe that by construction the matrix $\mathcal{D}_\mathcal{M} = [\mathcal{D}(v_i,v_j)]_{n \times n}$ is symmetric, and the $i^{th}$ row (or column) in $\mathcal{D}$, $p_i$, is an $n$-dimensional vector, characterizing the distance structure of point $v_i$ in model $\mathcal{M}$. All Pairs Shortest Path algorithm [13] is used for $\mathcal{D}_\mathcal{M}$ construction in the experiments.

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 $\mathcal{D}(.,.)$ to vary significantly, which in its turn, changes extracted features. Further, using geodesic distance as distance measure for decomposition does not tolerate noise (i.e. laser-scanned data) very well.

3.2.2 Local Distance Measure

Due to the above shortcomings of the geodesic distance measure 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 $\mathcal{M} \{t_1,\ldots,t_n\}$. For such distance measure, assume that $n$ denotes the number of triangles in the model ($n$ denotes number of points in the model $\mathcal{M}$ for the case where geodesic distance measure is used). 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 3.1(c) shows a maximum angle on angular shortest path between the faces \( t_1 \) and \( t_2 \). Specifically, let \( t_i \leadsto t_j \) denote the angular shortest path \((t_i, t_m, t_l, \ldots, t_j)\) between faces \( t_i \) and \( t_j \). And let \( t_m \rightarrow t_l \in t_i \leadsto t_j \) denote two adjacent triangular faces \( t_m \) and \( t_l \) on the angular shortest path \( t_i \leadsto t_j \). Then, the distance function used in this work is defined as

\[
\mathcal{D}(t_i, t_j) = \max_{t_m \rightarrow t_l \in t_i \leadsto t_j} \angle(t_m, t_l).
\] (3.7)

Intuitively, distance \( \mathcal{D}(t_i, t_j) \) 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 \( \mathcal{D}^{M} = [\mathcal{D}(t_i, t_j)]_{n \times n} \) is symmetric. Also note that distance measure \( \mathcal{D} \) 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 described in Section 3.2.1. As a result, introduction of the new distance measure for decomposition does not violate any statements made in Section 3.1.1 (except that now instead of points, faces are used as primitive entities) and all of the theorems would still hold.
3.3 Controlling Decomposition Process

The decomposition process as presented in Section 3.1.1 does not allow for an explicit mechanism to stop indefinite break up of a feature into a point-cloud. Clearly, one could use a constant 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 prescribed depth. This section presents a mechanism that will control the feature decomposition through constant measurement of coherence through out the decomposition paths. The use of various distance measures requires to use specialized control mechanisms for each distance measure. For each distance function (global and local) a specialized control mechanism is proposed. Intuitively, the use of this control mechanism will terminate the decomposition process only when all significant features are extracted.

3.3.1 Controlling Decomposition Process for Local Distance Measure

First, assign a measurement to each iteration of Feature-Decomposition algorithm. Specifically, Let $\mathcal{M}$ be the original model’s point set and $E$ denote the set of all edges connecting points in $\mathcal{M}$. Next, assume in the decomposition process a feature $\mathcal{M}_1$ in $\mathcal{M}$ can be decomposed into sub-features $\mathcal{M}_2$ and $\mathcal{M}_3$ after bisection (e.g., without loss of generality assume bisecting feature $\mathcal{M}_1$). The coherence measurement for $\mathcal{M}_1$ is defined as follows:

$$f(\mathcal{M}_1) = \frac{\rho(\mathcal{M}_1)}{\lambda(\mathcal{M}_1) + \rho(\mathcal{M}_1)},$$
where

\[ \rho(\mathcal{M}_1) = \sum_{(u,v) \in E} \mathcal{D}(u,v), \]

\[ (u,v) \in E, \quad u \in \mathcal{M}_2, \quad v \in \mathcal{M}_3 \]

denotes the distance of points across sub-features, and

\[ \lambda(\mathcal{M}_1) = \sum_{(u,v) \in E} \mathcal{D}(u,v) + \sum_{(u,v) \in E} \mathcal{D}(u,v), \]

\[ (u,v) \in E, \quad u \in \mathcal{M}_2, \quad u \in \mathcal{M}_3, \quad v \in \mathcal{M}_2, \quad v \in \mathcal{M}_3 \]

is the sum of distances within sub-features. Intuitively, the normalized ratio in \( f(\mathcal{M}_1) \) will account for coherence of \( \mathcal{M}_1 \) as oppose to its constituting sub-features \( \mathcal{M}_2 \) and \( \mathcal{M}_3 \). Specifically, bisection of \( \mathcal{M}_1 \) into \( \mathcal{M}_2 \) and \( \mathcal{M}_3 \) is said to be good if \( f(\mathcal{M}_1) \) is less than a prescribed threshold \( \tau \). If this threshold condition holds, the decomposition process continues for sub-features \( \mathcal{M}_2 \) and \( \mathcal{M}_3 \), otherwise it stops at \( \mathcal{M}_1 \).

### 3.3.2 Controlling Decomposition Process for Global Distance Measure

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:

$$\forall t_i \in \mathcal{M}_2, \ t_j \in \mathcal{M}_3 \ \exists t_m \rightarrow t_l \in t_i \Rightarrow t_j \text{ s.t. } t_m \in \mathcal{M}_2 \land t_l \in \mathcal{M}_3 \land \angle(t_m, t_l) = \mathcal{D}(t_i, t_j),$$

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.

### 3.4 Scale-Space Matching

The use of different distance measures in the Scale-Space decomposition technique results in different types of extracted features. Furthermore, when the Global Distance Measure is used, the resulting decomposition trees are rather stable – the significance of the extracted features correspond to the depth of this features in the decomposition trees. In other words, for two models $\mathcal{M}_1$ and $\mathcal{M}_2$ the features extracted on similar depths of the decomposition trees will have similar significance. This property allows the use of a dynamic programming algorithm to match the decomposition trees in bottom-up manner.

On the other hand, when Local Distance Measure is used, the obtained decomposition trees do not exhibit such property. As a result, the decomposition trees can not be used for matching. For Local Distance Measure matching of feature graphs is proposed. First, the
constructed decomposition trees are used to obtain final features (leaf nodes in the trees). Then, feature graphs are constructed based on the location of the extracted features within the models. Finally, a certain graph matching technique can be employed to compare these feature graphs.

3.4.1 Matching for Global Distance Measure

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 \( \mathcal{M} \). The edges of \( T_{\mathcal{M}} \) will capture the decomposition relation between a feature and all its sub-features. Figure 3.2.1 shows 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 \( T_2 \), a matching algorithm is introduced which is based on the dynamic programming framework proposed by Wang et. el. [63]. 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:

\[
C(T_1(u), T_2(v)) = C(F_1(u), F_1(v)) + \gamma(u, v). \tag{3.8}
\]
Figure 3.3: Results of matching similar models between each other with matched regions having similar colors. Most significant features are obtained for each model using \textsc{Feature-Decomposition}(\mathcal{M}, k) and are used to construct binary trees $T_1$ and $T_2$. 
\textsc{Match-Models}(T_1, T_2) then generates a set of matched features which have been colored similarly for this figure.

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 $v = \emptyset$) the following should hold:

$$C(T_1(u), \emptyset) = \gamma(u, \emptyset)$$  \hspace{1cm} (3.9)

$$C(F_1(u), \emptyset) = \sum_y \gamma(y, \emptyset),$$  \hspace{1cm} (3.10)

where the sum in Eq. (3.10) is over the roots of all trees in $F_1(u)$. Observe that for two features $u \in T_1$ and $v \in T_2$ to 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 the formulation of $C(T_1(u), T_2(v))$ in Eq. (3.8), using term $C(F_1(u), F_2(v))$. To compute the cost of matching the two forests $F_1(u)$ and $F_2(v)$, one need to compute a maximum similarity matching.
among the roots of trees in these two forests. To this end, one may use 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, associate the following as the cost of matching $x$ and $y$ in aforementioned bipartite graph:

$$w(x,y) = C(x,\emptyset) + C(\emptyset,y) + C(T_1(x),T_2(y)). \quad (3.11)$$

Observe that the term $C(T_1(x),T_2(y))$ is the basis of the recursion in this 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

$$C(F_1(u),F_2(v)) = \frac{1}{2} \left( \sum_x C(x,\emptyset) + \sum_y C(\emptyset,y) \right) + \sum_{(x,y) \in \Psi(u,v)} w(x,y). \quad (3.12)$$

The first two sums in Eq. (3.12) 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 $v$, $\Psi(u,v)$.

To state the recursive algorithm one need to specify the cost function $\gamma(u,v)$ in Eq. (3.8). In the formulation of $\gamma(.,.)$ the notion of topological similarity introduced by Hilaga et. al. in [21] is used. In fact, will set

$$\gamma(u,v) = e^{-\text{sim}(u,v)}, \quad (3.13)$$
where \( \text{sim}(u,v) \) is a convex combination of the form:

\[
\text{sim}(u,v) = w \cdot \min(\alpha(u), \alpha(v)) + (1-w) \cdot \min(\ell(u), \ell(v)), \text{ for } 0 \leq w \leq 1,
\]

for \( 0 \leq w \leq 1 \), with functions \( \alpha(.) \) and \( \ell(.) \) 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 \( \alpha(.) \) and \( \ell(.) \) see [21]).

Finally, one 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 two pairs of parts is represented in Fig. 3.3. Please note that final distance function is not metric. It satisfies symmetry and non-negativity properties but it does not satisfy triangle inequality by construction. Even though such case has never been encountered in the experiments, it is theoretically possible that matching algorithm would return similarity values for the models such that they would not satisfy triangle inequality.

Based on time bound proposed by Zhang and Shasha in [63] for this framework, the number of iterations of Algorithm 2 algorithm is \( O(n_1 n_2 \sqrt{2 \log d}) \), where \( d \) is the maximum degree of trees obtained in decomposition process, and \( n_1 \) and \( n_2 \) are the number of vertices in \( T_1 \) and \( T_2 \), respectively. Observe that \( n_1 \) and \( n_2 \) are much smaller than the initial sizes of \( \mathcal{M}_1 \) and \( \mathcal{M}_2 \). In the computations of running time, there will also be an additional multiplicative factor \( \phi(\mathcal{M}_1, \mathcal{M}_2) \) that accounts for the complexity of vanishing any feature in models \( \mathcal{M}_1 \) and \( \mathcal{M}_2 \). Also, in order for the algorithm to meet the time bound, the values of \( \alpha(u) \) and \( \ell(u) \forall u \in T \) are precomputed. Complexities for computations of \( \alpha(u) \) and \( \ell(u) \)
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 Eq. (3.9) and Eq. (3.10).
3: \(\forall v \in T_2\) COMPUTE \(C(T_2(v), \emptyset)\) and \(C(F_2(v), \emptyset)\) using Eq. (3.9) and Eq. (3.10).
4: \(\forall u \in T_1\) and \(\forall v \in T_2\) do
   - COMPUTE \(C(F_1(u), F_2(v))\) as in Eq. (3.12)
   - COMPUTE \(C(T_1(u), T_2(v))\) as in Eq. (3.8)
5: Return the set of matched vertices:
   \[\bigcup_{(u,v)} M(u,v),\]
   and the total cost of matching \(T_1\) and \(T_2\):
   \[C(\text{root}(T_1), \text{root}(T_2)).\]

are linear in terms of the number of points in partition \(u\).

3.4.2 Matching for Local Distance Measure

In contrast to global distance measure, when angular-based local distance measure is used in Scale-Space decomposition, the rooted undirected tree which is constructed during the decomposition process, can not be used for matching purposes. When local distance measure is used on two models \(M_1\) and \(M_2\), the features in these models that are similar (i.e. those that should be matched) may not appear on the same depth levels of the decomposition trees. It is reasonable to suggest that such behavior is due to the fact that only relative local information is used in distance computation. When global distance measure is used, the structures of decomposition trees do not exhibit such behavior and one can successfully use these decomposition trees for matching.

For the case of local distance measure, a different approach to matching should be
taken. When decomposition process is finished, the leaf nodes in decompositoin tree correspond to extracted features. These features can be used to construct feature graph for decomposed model: extracted features become nodes of the graph, edges are constructed based on the position of each feature within the model. Further, matching such feature graphs can be used for matching of 3D models.

For the sake of this work, a simple sub-graph isomorphism is used to assess similarity of the feature graphs. Hill-climbing algorithm with random restarts 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 local 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. Experimental Studies

The purpose of this work is to introduce a parameterizable feature extraction framework that could be used on models in polyhedral representation. The experimental results presented in this chapter suggest that the approach is capable of extracting features that can be used to classify 3D CAD models. Section 4.1 provides the results of benchmarking Global Scale-Space Approach (Local Distance Measure is used), as well as several other retrieval techniques, on a predefined dataset of CAD models. The main purpose of these experiments is to show that the Global Scale-Space Approach can successfully be used for retrieval of CAD models.

For the Local Scale-Space Approach (Local Distance Measure is used) a set of different experiments is performed in Section 4.2. The purpose of these experiments is to study the features that the Local Scale-Space Approach can extract. In addition, the effects that the presence of noise and incomplete model information produce are studied in the experiments in Section 4.2. The experiments show that the feature extraction technique have potential applications in partial matching and retrieval of scanned (i.e. noisy) models. Finally, it is shown that even when a well known sub-graph isomorphism method is used to match feature graphs of the models, the retrieval performance of the Local Scale-Space Approach is quite acceptable.

As it is shown in Section 4.2, the use of Local Distance Measure in the feature extraction process sophisticates the problem of matching. It would take additional research efforts to create a matching technique that could successfully match the feature graphs of
the models. Note that the current graph matching technique used in Local Scale-Space Approach performs on the graphs without any additional node (feature) information. Intuitively, matching part of the approach could significantly improve its retrieval abilities if feature information is properly used during matching.

4.1 Experimental Results for Global Scale-Space Approach

In order to assess performance of Global Scale-Space matching technique, a number of experiments were performed. The CAD datasets of 3D models was used, which was introduced in [3]. A number of different retrieval techniques were benchmarked using this dataset. This was done in order to better understand how well Global Scale-Space approach when compared to the other retrieval techniques.

Synthetic models and models of actual artifacts are provided in the CAD dataset. To assist classification, datasets are designed to test the sensitivity with regards to geometry, topology, relevant attributes to appearance, and manufacturing classification. Synthetic models are artificial models tailored for testing behaviors of classification systems to a particular geometry or topology. Actual artifacts are sampled and classified from the National Design Repository. This dataset consists of manufacturing and functional classified models. Furthermore, a LEGO© dataset presents an example of a homogeneous model set. All models are provided in ACIS SAT, STEP, and VRML formats. Each dataset will be presented with a sample view and statistics showing the average size of a model in the dataset under different file formats, average face counts for solid representations (SAT and STEP), as well as, average polygon counts for shape representations (VRML).
4.1.1 Synthetic Datasets

Synthetic models are provided to test the behaviors of model retrieval systems towards specific topological attributes in the interest of CAD/CAM. These synthetic models are created by the ACIS solid modeler in both SAT and STEP formats. The corresponding VRML shape models are then faceted by Sat2VRML.

Primitive Dataset

Cubes, cylinders, tori, and spheres with various deformations are created to test retrieval systems’ behavior on topological and geometrical classifications among the same models. To distort just the geometry, but retaining topology, unit primitives are blended and scaled in $x, y, z$ directions to create 296 models. The set consists of 101 cubes, 141 cylinders, 29 tori, 29 spheres. Two different classifications are produced:

- Group models by their types (Topologically similar groupings, e.g. groups: Cubes, Cylinders, Tori, and Spheres)
- Group models by their deformations (Geometrically similar groupings. e.g. groups: $1 \times 1 \times 1$, $2 \times 1 \times 2$ and $1 \times 1 \times 4$)

Figure 4.1 gives a sample view of this dataset, and table 4.1 shows a brief summary of this dataset it is available at:


\[1\text{http://gicl.cs.drexel.edu/sat2vrml/}\]
### Table 4.1: Statistics of the Primitive Dataset

<table>
<thead>
<tr>
<th></th>
<th>#Models</th>
<th>Avg. #Faces</th>
<th>Avg. #Polygons</th>
</tr>
</thead>
<tbody>
<tr>
<td>Tori</td>
<td>29</td>
<td>1</td>
<td>2356</td>
</tr>
<tr>
<td>Spheres</td>
<td>29</td>
<td>1</td>
<td>1146</td>
</tr>
<tr>
<td>Cylinders</td>
<td>141</td>
<td>5</td>
<td>1000</td>
</tr>
<tr>
<td>Cubes</td>
<td>101</td>
<td>22</td>
<td>848</td>
</tr>
<tr>
<td>Total</td>
<td>300</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>Tori</td>
<td>23KB</td>
<td>37KB</td>
<td>144KB</td>
</tr>
<tr>
<td>Spheres</td>
<td>12KB</td>
<td>20KB</td>
<td>72KB</td>
</tr>
<tr>
<td>Cylinders</td>
<td>21KB</td>
<td>47KB</td>
<td>48KB</td>
</tr>
<tr>
<td>Cubes</td>
<td>27KB</td>
<td>39KB</td>
<td>40KB</td>
</tr>
</tbody>
</table>

Groupings

\[1 \times 1 \times 1\]

\[2 \times 1 \times 1\]

Figure 4.1: Examples of the models from the primitive models dataset.

**Minor Topological Variation Dataset**

This dataset consists of rectangular boxes with a differing numbers of holes. They are designed to test the behavior of retrieval systems under minor topological variations. It evaluates the effect of varying simple features, such as holes on rectangular boxes.

Figure 4.2 gives a sample view of the dataset and table 4.2 shows a brief summary of this dataset. It is available at:

http://www.designrepository.org/datasets/bricks.tar.gz
Cubes-Holes  Sixteen (16) cubes were modeled with a different numbers of holes (1, 2, 3, or 4 holes). Holes were made with a different radii, in addition, each model is constructed with holes with the same radii. The models are organized into four groups by the number of holes in each model. Figure 4.2(a) shows an example of a cube model from the dataset.

Brick-Holes  Eleven (11) rectangular box models with zero to four holes of the same size in different locations were created: one model with no holes, four models with one hole, three models with two holes, two models with three holes, and one model with four holes, as shown in Figure 4.2(b). Similar to the previous dataset, the models were grouped by their respective number of holes.

<table>
<thead>
<tr>
<th>#Models</th>
<th>Avg. #Faces</th>
<th>Avg. #Polygons</th>
</tr>
</thead>
<tbody>
<tr>
<td>Bricks</td>
<td>11</td>
<td>8</td>
</tr>
<tr>
<td>Cubes</td>
<td>16</td>
<td>9</td>
</tr>
<tr>
<td>Total</td>
<td>27</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Avg SAT size</th>
<th>Avg STEP size</th>
<th>Avg VRML size</th>
</tr>
</thead>
<tbody>
<tr>
<td>Bricks</td>
<td>6KB</td>
<td>17KB</td>
</tr>
<tr>
<td>Cubes</td>
<td>6KB</td>
<td>19KB</td>
</tr>
</tbody>
</table>

4.1.2 Actual Artifacts Dataset

In addition to synthetic models, models of actual artifacts are also provided, namely, mechanical engineering parts available from the National Design Repository and LEGO©

pieces. The National Design Repository models were sampled from industrial CAD data and grouped under multiple classification schemes. LEGO® parts and assemblies are provided as an example of homogeneous part families with repeating features.

**The National Design Repository Dataset**

CAD models in this dataset are collected from industry, and can be obtained through the publicly available National Design Repository. Two sets of models are hand classified under two classification schemes: (1) Manufacturing classification, a binary classification for prismatic machined or cast-then-machined parts. (2) Functional classification, a multi-category classification of brackets, gears, screws, springs, nuts, housing, and linkage arms. A sample view of the National Design Repository CAD models is shown in Figure 4.3.
Manufacturing Classification Dataset  This dataset was classified by hand into (1) prismatic machined parts and (2) parts that are first cast and then have their finishing features machined. The engineering rationale in this classification is that parts that are exclusively machined are usually high-precision parts, or parts made in small batches (i.e., for custom jobs). Cast-then-machined parts are typically from larger production runs and generally have much looser tolerance considerations for the non-machined surfaces of the object. In this case the investment of the physical plant is larger, as is the manufacturing production plan (i.e., one needs to machine a mold with which to do casting). Figure 4.4 shows a
sample of this dataset, and table 4.3 shows a brief summary of this dataset it is available at:

http://www.designrepository.org/datasets/machined.tar.bz2

and


<table>
<thead>
<tr>
<th></th>
<th>#Models</th>
<th>Avg. #Faces</th>
<th>Avg. #Polygons</th>
</tr>
</thead>
<tbody>
<tr>
<td>Prismatic Machined</td>
<td>56</td>
<td>106</td>
<td>3600</td>
</tr>
<tr>
<td>Casted-then-Machined</td>
<td>54</td>
<td>80</td>
<td>3447</td>
</tr>
<tr>
<td><strong>Total</strong></td>
<td><strong>110</strong></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>Prismatic Machined</td>
<td>146KB</td>
<td>233KB</td>
<td>162KB</td>
</tr>
<tr>
<td>Casted-then-Machined</td>
<td>277KB</td>
<td>314KB</td>
<td>159KB</td>
</tr>
</tbody>
</table>

**Functional Classification Dataset**  This dataset 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 4.5 shows a sample of this dataset, and table 4.4 shows a brief summary of this dataset it is available at:


**LEGO® Dataset**

The LEGO® dataset aims to provide a benchmark for a part family composed of homogeneous features. This dataset consists of LEGO® pieces from the popular LEGO®
Mindstorms® robotics kit. The remarkable characteristic of this dataset is that all LEGO® components are composed with a fixed set of features. In addition, these features exhibit explicit interactions between one another. For instance, a pin on the top of the plate can fit in a hole on another piece. Forty seven (47) LEGO® components were modeled in ACIS and classified into four categories according to their appearance. The groups are named as follows: plates, wheels and gears, cylindrical parts and X-shape axles. Figure 4.6 gives a sample view of the dataset, and table 4.5 shows a brief summary of this dataset it is available at:


<table>
<thead>
<tr>
<th>PRISMATIC MACHINED</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td><img src="image1.png" alt="Image" /></td>
<td><img src="image2.png" alt="Image" /></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>CAST-THEN-MACHINED</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td><img src="image3.png" alt="Image" /></td>
<td><img src="image4.png" alt="Image" /></td>
</tr>
</tbody>
</table>

Figure 4.4: Examples of the models from the manufacturing classification dataset.
Table 4.4: Statistics of Functional Dataset

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

<table>
<thead>
<tr>
<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>
</tr>
<tr>
<td>Gears</td>
<td>458KB</td>
<td>525KB</td>
</tr>
<tr>
<td>Housings</td>
<td>300KB</td>
<td>450KB</td>
</tr>
<tr>
<td>Linkage Arms</td>
<td>62KB</td>
<td>100KB</td>
</tr>
<tr>
<td>Nuts</td>
<td>13KB</td>
<td>19KB</td>
</tr>
<tr>
<td>Screws and Blots</td>
<td>18KB</td>
<td>30KB</td>
</tr>
<tr>
<td>Springs</td>
<td>620KB</td>
<td>960KB</td>
</tr>
</tbody>
</table>

**Functional Classification**

- Linkage Arms
- Housings
- Brackets
- Nuts
- Gears
- Screws
- Springs

Figure 4.5: Examples of the models from the functional classification dataset.
### Table 4.5: Statistics of the LEGO© Dataset

<table>
<thead>
<tr>
<th></th>
<th>#Models</th>
<th>Avg. #Faces</th>
<th>Avg. #Polygons</th>
</tr>
</thead>
<tbody>
<tr>
<td>Plates</td>
<td>30</td>
<td>69</td>
<td>3328</td>
</tr>
<tr>
<td>Wheels-Gears</td>
<td>4</td>
<td>256</td>
<td>2536</td>
</tr>
<tr>
<td>Cylindrical Parts</td>
<td>6</td>
<td>30</td>
<td>886</td>
</tr>
<tr>
<td>X-Shape Axles</td>
<td>7</td>
<td>10</td>
<td>204</td>
</tr>
<tr>
<td><strong>Total</strong></td>
<td><strong>47</strong></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>Plates</td>
<td>40KB</td>
<td>130KB</td>
<td>146KB</td>
</tr>
<tr>
<td>Wheels-Gears</td>
<td>40KB</td>
<td>63KB</td>
<td>14KB</td>
</tr>
<tr>
<td>Cylindrical Parts</td>
<td>43KB</td>
<td>94KB</td>
<td>57KB</td>
</tr>
<tr>
<td>X-Shape Axles</td>
<td>8KB</td>
<td>24KB</td>
<td>10KB</td>
</tr>
</tbody>
</table>

Figure 4.6: Examples of the LEGO© dataset.

The performance of retrieval systems on the models with homogeneous features can be assessed by the LEGO© dataset. This dataset is designed to be especially useful for systems employing feature extraction in the process of retrieval.

#### 4.1.3 Experimental Protocol

Nine solid and shape based comparison techniques were evaluated.
• Shape based techniques

  – Shape distributions (SD) [40]
  – Shape distributions with point pair classifications (SD-Class) [22]
  – Reeb graph comparison (Reeb) [21]
  – Shape distributions with weights learning (SD-Learn) [23]
  – Global Scale-Space comparison (Scale-Space) [6]

• Solid based techniques

  – B-Rep based techniques
    * Invariant topological vector (ITV) [37]
    * Eigenspace indexing on B-Rep graphs
      (Eigen-BRep) [41]
  – Feature based techniques
    * Model dependency graph approximate matching (MDG) [11]
    * Eigenspace indexing on machining feature interaction graphs
      (Eigen-Feat) [41]

Note that feature based techniques are only applicable to models of actual artifacts with explicit machining feature interactions. Feature based experiments were not performed on synthetic datasets nor LEGO© dataset, as they either contain no machining feature (primitive datasets) or the features do not interact (Minor topological variation dataset, LEGO© datasets).
Machining features and feature interactions of actual artifacts were extracted by Honeywell FM&T’s FBMach feature recognition system. FBMach decomposes an ACIS part into STEP AP 224 volumetric machining features. These features are typically used for process planning and for programming CNC machine tools. Feature interaction graphs and model dependency graphs used by, respectively, the Eigen-Feat and MDG techniques, are constructed by using these FBMach features. The recognized machining features map to the graph’s vertices. Interactions between the features were detected by testing intersections among the feature volumes. These interactions map to the edges and complete the respective graphs.

The performance of various techniques are evaluated by the $k$-nearest neighbor classification ($k$NN), and conventional recall and precision measures for evaluating information retrieval systems. The recall and precision values at different thresholds are computed as follows:

$$\text{recall} = \frac{\text{Retrieved and Relevant models}}{\text{Relevant models}}$$

$$\text{precision} = \frac{\text{Retrieved and Relevant models}}{\text{Retrieved models}}$$

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 the 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 indicate a better performance.

### 4.1.4 Experimental Results

Rather than having a competitive evaluation to demonstrate one retrieval technique outperforming the others, experimental results show each retrieval technique possesses different strengths producing satisfactory performance on some but not all synthetic model evaluations. Under the manufacturing classification dataset of actual artifacts, all evaluated techniques produced unsatisfactory performance, indicating there is a need for further research in the interest of CAD/CAM models retrieval.
Synthetic Datasets

**Cube-Holes and Brick-Holes Dataset** On this topologically invariant synthetic datasets, graph and solid model based ITV and Eigen-BRep performed the best on the Cube-Holes, Figure 4.7(a), and Brick-Holes, Figure 4.7(b). However, reeb graph technique performed better than the other shape based techniques. In these two datasets, models are composed with either holes in different locations or different diameters exclusively. The results demonstrated the difference in topological sensitivity in between solid, graph and shape based techniques. Solid and graph based ITV and Eigen-BRep captured better invariant topology. Among shape based techniques, the reeb graph technique produced a better precision recall than the other shape based techniques.

**Primitive Dataset** Under deformation classifications, shape-based techniques, and namely the shape distribution technique was the most effective one, as shown in Figure 4.7(d), whereas ITV and Eigen-BRep performed the best on type classification of the primitives, Figure 4.7(c).

The deformation classification grouped models based on their geometry. For instance, unit primitives were grouped together, long cylinders, and long bricks formed another group. Shape distribution technique performed best on this classification as it was sensitive to gross geometric similarities between models.

Type classification grouped models together according to their topology. For example, bricks, cylinders and ellipsoids formed different categories. ITV, Eigen-BRep capture similar topological structures and are the most effective techniques for this classification.

The primitive dataset demonstrated that the performance of retrieval techniques de-
pends on the model classification schema. Moreover, the performance of retrieval tech-
niques can vary drastically.

Figure 4.7: Precision-Recall graphs on synthetic datasets. (*PLOTS ARE IN COLOR.*)
Actual Artifacts Datasets

Manufacturing Classification Dataset  All retrieval techniques performed similarly on manufacturing classification dataset, Figure 4.8(a). However, Eigen-Feat and MDG show slightly better performance than the rest of the techniques. The precision fell to 50%, which is close to random for binary classifications, at a low recall rate, showing the techniques are not able to classify the models properly. The result questions the discrimination power of the tested techniques on prismatic machined and cast-then-machined classifications, indicating that there is a need for further work for this classification.

Functional Classification Dataset  In contrast to manufacturing classification dataset, the performance of retrieval techniques remained steady, with the exception of scale-space and MDG retrieval, Figure 4.8(b). A steep slope of the precision-recall curve shows the technique’s precision was low even under low recall settings.

LEGO Dataset  The Scale-space technique performed slightly better on the LEGO dataset than the other techniques with a high recall settings, Figure 4.8(c). The LEGO dataset provides an example of models comprised of repeating features. This special property allows the scale-space technique to extract the repeating features during its decomposition process.

4.2 Experimental Results For Local Scale-Space Approach

In the experiments, the qualities of features extracted using the FEATURE-DECOMPOSITION($\mathcal{M}, k$) algorithm are studied. To these ends, FEATURE-DECOMPOSITION($\mathcal{M}, k$) is recursively ap-
Figure 4.8: Precision-Recall graphs on actual artifacts datasets. *(PLOTS ARE IN COLOR.)*

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 4.9 for an illustration of the feature extraction process. For illustrative purposes, only a subset of extracted features is shown in Figure 4.9(b); the features shown in Figure 4.9(a)
(a) Decomposition tree is obtained using \textsc{Feature-Decomposition}(M,k) algorithm. (b) Leaf nodes of the tree correspond to the features.

Figure 4.9: Feature extraction process.

do not correspond to the leaf nodes in Figure 4.9(b). The actual decomposition tree is quite large for this model.

In addition to the experiments on examining the qualities of extracted features using Local Scale-Space approach, a set of matching experiments was performed. Actual Artifacts datasets were used for this purpose. The aim of the matching experiments for Local Scale-Space was to support the claim that the Scale-Space feature extraction characterises the structure of the 3D model and can be used for matching of 3D CAD models. The matching experiments did not address thorough performance assessment due to the triviality of the actual matching technique used to compare feature graphs. In order to perform better matching of feature graphs, a specialized graph matching technique should be employed.
As the experiments suggest, such matching technique should be able to perform many-to-
many matching of the nodes (features), as well as use some kind of signature for each node
(feature) in the graph. The development of the aforementioned graph matching technique is
out of the scope for this current research work and should be addressed in the future work.

4.2.1 Feature Decomposition on CAD Data

Figure 4.10 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.

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 parial objects was obtained which enabled to see how the FEATURE-
DECOMPOSITION($\mathcal{M},k$) algorithm performs on the models where only partial data is
available. Illustrations of extracted features could be found in Figure 4.10.

4.2.2 Feature Decomposition on Noisy Data

In order to simulate the noise produced from capturing the object using 3D laser scan,
Gaussian noise was applied to each point of the models presented in Section 4.2.1. Gas-
sian 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 4.11 and 4.12. 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. Such names were chosen for the purpose of referencing the models within this work.
Figure 4.10: 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="Image" alt="Image" /></td>
<td><img src="Image" alt="Image" /></td>
</tr>
<tr>
<td><strong>SIMPLE BOEING</strong></td>
<td><img src="Image" alt="Image" /></td>
<td><img src="Image" alt="Image" /></td>
</tr>
<tr>
<td><strong>PART 9</strong></td>
<td><img src="Image" alt="Image" /></td>
<td><img src="Image" alt="Image" /></td>
</tr>
<tr>
<td><strong>PART 10</strong></td>
<td><img src="Image" alt="Image" /></td>
<td><img src="Image" alt="Image" /></td>
</tr>
</tbody>
</table>

Figure 4.11: 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.
Figure 4.12: 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.
4.2.3 Feature Decomposition on Acquired Models

It has been 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 4.13(b)) and partial 3D view (Figure 4.13(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 \textsc{Feature-Decomposition}($\mathcal{M}, k$) algorithm.

Figure 4.14 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 4.2.1). Although, 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 the database. The scanned models used for this experiment are freely available at http://edge.cs.drexel.edu/Dmitriy/Scanned.tar.bz.
(a) Single Scan – take one scan from a single view

(b) Full Scan – take multiple scans from multiple views and register these scans together

Figure 4.13: Illustration of acquisition process.
Figure 4.14: Correspondence of selected extracted features for the full scan models, single scan models and models obtained from exact representation (ACIS model).
4.2.4 Matching

For the matching experiment, Functional Classification Dataset was chosen from the Actual Artifacts Datasets. A similar procedure to the one described in Section 4.1.3 was conducted. Only three retrieval techniques were used for the sake of this experiment: Global Scale-Space, Reeb Graph [21] and Local Scale-Space techniques. The precision-recall graphs for the dataset can be found in Figure 4.15. It appears that the Reeb Graph technique performs relatively better than both Scale-Space approaches, while Local Scale-Space technique out-performs Global Scale-Space for this dataset. The results of this experiment show that the use of Local Scale-Space feature extraction technique results in meaningful decomposition that could potentially be used for matching of 3D CAD data.
5. Contribution and Future Work

5.1 Discussion and Future Work

The Global Scale-Space technique appears to be performing reasonably well when the entire 3D object is provided without presence of the noise. Further, the Global Scale-Space technique becomes unusable when applied to partial models and/or models with the presence of noise. For this kind of models, Local Scale-Space technique is more appropriate, since it is less sensitive to the overall (global) structure of the models.

Moreover, Global Scale-Space technique utilizes attributes \( \alpha (.) \) and \( \ell (.) \) for each node (feature). The use of these attributes was inspired by Reeb Graph approach. It is possible that the performance of the Global Scale-Space technique could be drastically improved by introduction of more meaningful attributes for each feature that could capture feature’s topological and/or geometrical information better. It should be pointed out that the attributes \( \alpha (.) \) and \( \ell (.) \) were successfully used in matching hierarchical graphs which were obtained using Reeb Graph technique. These hierarchical graphs normally contain more nodes (features) than the decomposition trees obtained using Global Scale-Space decomposition. As a result, one may use attributes \( \alpha (.) \) and \( \ell (.) \) for successful matching of Reeb Graph based hierarchical graphs; while such attributes may not be suitable for successful matching of feature decomposition trees. Therefore, one of the possible future directions of this work is to introduce a new attribute measure to perform node to node comparison when Global Scale-Space feature decomposition trees are matched. Such attribute measure could potentially be used to compare Local Scale-Space feature graphs.
The next objective for this work is to introduce an efficient matching algorithm for partial similarity measures of Local Scale-Space feature graphs. From the above experiments one may conclude that in order to perform successful matching, the technique must have the following properties: (1) tolerance for the noise that scanned data introduce; (2) ability to perform many-to-many matching, since it is possible that a feature could get divided into several features (Figure 5.1 gives an instance of such situation); (3) efficiency, so it can be used in the National Design Repository database.¹

Other possible directions for Scale-Space work are to (1) explore techniques to extract
features that resemble traditional CAD features; and (2) exploit the possibility of using Scale-Space features as signatures for database indexing purposes.

5.2 Contribution

This document 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 results of decomposition are based on the distance measurement function that could be defined in a number of ways. The current implementations of the feature extraction technique use global (geodesic distance) and local (max angle on angular shortest path) distance functions. The later distance function allows for similar features to 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.

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 work is highly tuned to the efficient identification of shape and topological categories. In one application, features obtained using the Scale-Space approach could be different from traditional CAD features and used to establish partial similarities between CAD models in polyhedral representation. One may argue that the Scale-Space technique can be parameterized using different measurement functions, enabling it generate a variety of useful segmentations, including those that have semantic relevance to engineering and manufacturing properties.
The locality-based feature representation can be used for 3D matching purposes, including partial matchings. Further, the Scale-Space decomposition technique is robust with respect to noise; therefore, it can be used on 3D models generated from 3D data acquisition devices, such as laser range scanners.

The Scale-Space approach developed and advanced in this work 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.

Ultimately, it is believed that Scale-Space techniques provide important new capabilities that compliment 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.
Bibliography


