Shape Feature Abstraction in Knowledge-based Analysis of Manufactured Products

Rajit Gadh and Friedrich B. Prinz

Carnegie Mellon University
Pittsburgh, PA 15213

Abstract
Knowledge required to analyze designs for functional considerations such as manufacturability, assemblability, functionality, etc., may be based on shape features, which are physically distinguishable regions on parts, and which usually perform certain functions. The shape features required to analyze manufacturability are not always explicitly available in surface CAD models of parts, and may need to be extracted. However, it is the variation of shape features with topology and geometry that makes the problem of feature recognition difficult. The current research focuses on shape feature recognition based on the Differential Depth Perception Filter, which reduces the number of topological entities. Yet another level of operation causes the topological entities to be transformed into entities of a higher abstraction level called loops. Loops aid in the reduction of number of entities in which features need to be searched, which implies reduction in search space. Considerable success in feature recognition has been achieved for parts with varying topology and geometry and for parts with a relatively large number of topological entities.

AI Topic: Knowledge-based Analysis in Manufacturing (or Concurrent Engineering).
Domain Area: Shape Feature Recognition in Design for Manufacturability, Assemblability, etc.
Language/Tools: "C", UNIX.
Person Years: 2+.
Impact: Using the system one may automatically extract shape features from CAD models, and subsequently perform knowledge-based design for manufacturability analysis, thereby reducing concept to production time and manufacturing cost.

1. Introduction
Heuristic analysis of designs for various life cycle considerations is typically performed by manufacturing experts. One of the inputs they use are geometric shape abstractions known as shape features [Gadh90A]. Knowledge-based systems may be made to mimic the reasoning of a human expert, and therefore also require knowledge of such features. Typically three types of knowledge are required with respect to shape features: 1. Feature parameters, e.g. height or width of a feature, 2. Feature relationships, e.g. distance between two features, and 3. Feature interactions, e.g. recursion between two features. In general, there are two classes of approaches to obtain such information: 1. Designing with Features, and 2. Feature Recognition. The current approach to obtaining features is through feature recognition.

1.1 Designing with Features
In a Design With Features approach features are used as building blocks to create the part geometry ([Dixon88A], [Libardi86]). Designing with features is typically performed by selecting features from a features library, and using boolean operations on them.

In using a Design with Features approach, when a design-with feature is added to a part, the manufacturability of the part may be checked with respect to attributes of the placed feature. In a pure design with features approach, manufacturability analysis of secondary features, which are combinations of primary features, will not occur unless recognition of such features is performed. While a feature may take on an infinite set of geometric shapes, a design-with-features system typically contains a finite number of geometric features with which the part may be created. Therefore a pure design-with-features system may not be able to handle variability of shape within features.

1.2 Feature Recognition
Feature recognition involves obtaining feature parameters, feature relations, and feature interactions, given the geometric CAD data. Once feature recognition is performed, knowledge of features, feature locations on the part, feature parameters, feature relationships and feature interactions between features is obtained. To represent relationships between features we define a network called the Feature Relationships Network, which is similar in spirit to the feature graph [Gadh89] but is somewhat richer in content. In a feature graph an arc represents connectivity, whereas in the feature relationships network, it contains not only the fact that two features are connected, but also the exact
form of connectivity. For example, for the part shown in figure 1.1, the form of connectivity between the features boss and rib is recursion. Such a connectivity is represented by a directed arc on the network, and knowledge of the type of connection, i.e., recursion.

Two examples of relationships in this network are sharing-face and sharing-edge which represent sharing of face and edge respectively. Another example of a relationship on an arc is adjacent-to in which two features touch each other. Using such an enhanced form of feature graph a better understanding of shape feature relationships exists.

The feature relationships network may be used for analysis pertaining to life cycle considerations such as manufacturability, assemblability, cost estimation etc. Upon traversal of the feature graph, feature parameters, feature relationships, and feature interactions may be deduced, which are needed by such analysis. Figure 1.1 shows a part before and after feature recognition with the corresponding feature network. The feature boss sits on the rib and this may be deduced by travelling down the arc from boss to the rib. The arrow direction in the directed arc points to the parent feature.

Figure 1.1: Feature relationships net

1.3 Literature Survey
A reasonable depth and breadth of literature exists in the field the use of shape features in design for manufacturing and their recognition from surface models ([deFloriani89], [Henderson84], [Joshi87], [Kung84], [Kyprianou80], [Marafat90], [Pinilla89], [Finger90], [Sakurai88]). The commonly found approaches to shape feature recognition are using Syntactic Approaches (or Formal Grammars), and Subgraph Matching.

Syntactic Grammars are based on the formal definition of features in terms of a language grammar. A grammar is defined by \( \{V, T, P, S\} \) where the symbols \( V \), \( T \), \( P \), \( S \) are Variables, Terminals, Productions and Start symbols [Hopcroft69]. Grammars have traditionally been used for 2D pattern recognition as well as 3D shape recognition [Lee87] [Finger90]. Grammars based on graphs of 3D BREP models are called graph grammars [Pinilla89], in which the terminals are nodes, edges, and faces.

Kyprianou [80] utilizes a syntactic approach using feature grammars to extract features from the BREP model of an object. An augmented topology graph grammar approach has been proposed by Pinilla, Finger, and Prinz [Pinilla89]. Features are defined in terms of graph grammars that contains topology as well as geometry. The features considered in both these approaches are rib, boss, hole, slot, pocket, etc.

Graph grammars have effectively been able to describe simple shapes such as slots, rectangular ribs, etc. In reality the topological and geometric variation of features in manufactured parts is so substantial, that most grammars become simply too cumbersome when describing all the instances of features.

Another type of approach, uses Subgraph Matching, in which features are defined in terms of an F-E-V graph which is searched for in the BREP graph of the part. Sakurai and Gossard [Sakurai88] use subgraph matching for feature recognition of user definable features. Once features are found, volumetric removal is performed. The process is repeated until no more features are found.

One of the limitations of subgraph matching is that for each geometric or topological variation of a shape feature, a new subgraph needs to be defined. The larger the amount of variation in shapes within a feature type, the greater the number of subgraphs required. Handling each instance of a shape with an independent subgraph may easily result in the number of subgraphs becoming undesirably large. Yet another drawback of subgraph matching, as has been mentioned, is that it is in general combinatorially explosive.

The above classes of feature recognition approaches and other similar ones are more appropriate for features with a reasonably small number of variations in shape. Variation in topology, variation in geometry, and complexity due to combinatorics, cause a substantial barrier to feature recognition, which such approaches may not be able to handle effectively.

2. Differential Depth Perception Filter
The current approach to feature recognition, which is motivated by such difficulties, is based on a filtering scheme that reduces the number of topological entities on the surface model of a part. The approach may be considered as being analogous to the manner in which humans perceive shapes. The human eye is able to distinguish shape features with relative ease. Research has shown that humans recognize scenes by identifying shapes using one or more of the following classes of information: Stereo information [Marr77], Shading [Ikeuchi81], Texture [Rosenfeld71], Shadow [Shafer83], and Contour [Marr77].

199
The contour is essentially the outline of an object, which we call a silhouette. The Differential Depth Perception Filter which is the basis of the current feature recognition approach, is motivated by the fact that contours are one of the inputs used by humans to recognize features. The filter is a mathematical operator, that when applied to the surface model of the part produces a silhouette-like output, and results in a substantial reduction of the size of the F-E-V graph that describes the object, thereby reducing the number of topological entities in which features are searched for.

It may be intuitively inferred that a silhouette consists only of convex edges of a surface model. A convex edge is that in which angle between the two edge faces inside the solid is less than 180°. Concave edges are those which have the same angle greater than 180°. We define a viewing direction, \( V_j \) which is the viewing direction for the filter in \( R^3 \) space. Consider an infinitely large number of rays cast parallel to \( V_j \) originating from a plane, \( P \), perpendicular to \( V_j \) and striking the object at various points. We now define two rays of light \( R_{i,1} \) and \( R_{i,2} \) to belong to a topological edge \( E_i \) if the rays pass close to the edge, one on each side of the edge at a distance of \( \epsilon \), where \( \epsilon \) is a quantity much smaller than the length of the smallest edge on the model. We define the two faces on edge \( E_i \) to be \( F_{i,1} \) and \( F_{i,2} \). There are three possible orientation of the viewing direction w.r.t. to the convex edge \( E_i \). Figure 2.1 shows these orientations in a cross section profile of an object at the edge which are as follows:

1. \( R_{i,1} \) pierces one of \( F_{i,1} \) and \( F_{i,2} \) from outside the solid; \( R_{i,2} \) pierces the other from outside (Fig. 2.1-a).
2. \( R_{i,1} \) pieces one of \( F_{i,1} \) and \( F_{i,2} \) from inside; \( R_{i,2} \) pierces the other from inside (Fig. 2.1-b).
3. Either \( R_{i,1} \) or \( R_{i,2} \) pieces one of \( F_{i,1} \) and \( F_{i,2} \) from outside, and the other does not intersect \( F_{i,1} \) or \( F_{i,2} \) (Fig. 2.1-c).

From among the set of edges \( E_i, i = 1, N_e \), the ones that satisfy criterion 3 for a given direction of viewing are considered to pass through the filter, and are called filtered edges. This filter will be called the Positive Differential Depth Perception Filter. The filter obtained by viewing the part from a single direction is called a Uni-directional Filter.

The algorithm used for obtaining the Uni-directional positive filter is linear in the number of model edges, and is as follows:

1. **Establish a viewing direction**: \( V_j \);
2. **For each edge**, \( E_i \) in the model {
   - if (\( E_i \) is convex) {
     - Let \( F_{i,1} \) and \( F_{i,2} \) be two faces on \( E_i \);
     - Let \( N_{i,1} \) and \( N_{i,2} \) be their normals;
     - if Not \((N_{i,1}.V_j < 0 \& N_{i,2}.V_j < 0)\) OR \((N_{i,1}.V_j > 0 \& N_{i,2}.V_j > 0))\) {
       - Mark Edge \( E_i \) as filtered;
     } } }

where \( A.B \) represents the dot product of two vectors.

2.1 **Positive and Negative Filters**

To demonstrate the feature recognition approach using the filter, we use a part with a boss, a rib, a through hole and a blind hole. The positive filter of the part is shown in figure 2.3-a, in which \( V_j (j=1) \) is perpendicular to the plane of the paper, looking into the paper. Subsequently, the original object is filtered again by performing the filtering operation from a number of other directions and a union of the these filtered parts is performed. This filter is called the Multi-directional Positive Filter (shown in Figure 2.3-b). The selection of the viewing directions, \( V_j, j=1, N_v \), is explained later.

![Figure 2.2 Example part](image)

Subsequently, a new type of filter, the Negative Differential Depth Perception Filter is defined the positive filter for the solid complement of the object. The solid complement of an object is defined as the region in space that is not inside the solid.

2.3 **Positive Filter** a. 1 View, b. Many Views

We represent the positive filter as:

\[
F_{P,V_j}(E) = G_{P,V_j}(M(F, E, N)), j=1, N_v
\]

and the negative filter as:

\[
F_{n,V_j}(E) = G_{n,V_j}(M(F, E, N)), j=1, N_v
\]
where $M$ is the BREP model of the part which consists of model faces ($F$ is the set of model faces, of which there are $N_F$), model edges ($E$ is the set of model edges of which there are $N_E$) and model nodes ($N$ is the set of model nodes of which there are $N_N$); $V_j (j=1, N_V)$ is the viewing direction; $G_p, V_j$ is the positive filter operator viewing from direction $V_j$; $G_n, V_j$ is the negative filter operator viewing from direction $V_j$. $F_p, V_j$ and $F_n, V_j$ are the uni-directional filtered outputs for the part for the viewing direction, $V_j$. When the part is viewed from a number of directions, the multi-directional filtered model is obtained.

![Figure 2.4: Positive and Negative Entities](image)

### 2.2 Filter Loops

It may be observed that both the projecting features, the rib and boss, have a positive cyclic list of edges and a negative cyclic list of edges which we refer to as a loop of edges. A loop consisting only of positive filtered edges is called a positive loop; similarly one consisting only of negative filtered edges is called a negative loop. Figure 2.4 shows looped and non-looped edges on the example part.

In terms of the two loop types, positive and negative, various features may be defined in a manner whereby their recognition in a CAD model is robust. For example, the rib in Figure 2.4 has one positive and one negative loop connected by a number of non-looped edges. The loops and non-looped edges are configured in a manner so as to enclose material. Both the boss and the rib fall into the class of features known as the single connected projecting feature. This class of features will in general have one negative loop at the base of the feature and will enclose material. The second loop, the positive loop is optional.

In a similar fashion, other feature classes that may be defined are the blind depression feature, through depression feature, and bridge (or double connected projecting feature). This class of features is called the fundamental feature set. The various classes are characterized as follows:

- **Blind Depression Feature**: It has a positive loop, and does not enclose material, e.g., rectangular slot, step, through slot, blind hole (in Figure 2.4).
- **Through Depression Feature**: It has two positive loops, with non-looped filtered edges connecting them, and does not enclose material (Figure 2.4).
- **Bridge** (or double connected projecting feature): It has two negative loops, with non-looped filtered edges connecting them, and does not enclose material.

### 2.3 The Philosophy Behind Using Loops

Definition of features in terms of loops and connecting non-looped entities is more robust than definition of features in terms of F-E-V adjacency relationships. This is because, changing the number of topological entities by changing the resolution of the surface model or by changing the geometry may produce completely different models. If for example, the two ribs shown in Figure 2.5 are defined in terms of subgraphs the two subgraphs will be different. Using the loop-based feature definitions, however, the two ribs in Figure 2.5 by virtue of satisfying the characteristics of the projecting feature class, will both map to this same feature class.

![Figure 2.5: Similarity in loops among ribs](image)

Therefore, the loop represents a higher level of abstraction than faces, edges and vertices, {F, E, V}, yet a lower level of abstraction than the shape features themselves. Translation of the lowest level abstraction, {F, E, V}, to the highest level abstraction, shape feature, has been treated as a one step operation by conventional approaches. A direct search for features in terms of F-E-V adjacency relationships is known to produce combinatorial explosion during search, besides creating an arbitrarily large set of {F, E, V} configurations that could produce the same feature. The loop is a powerful intermediate abstraction which prevents combinatorial explosion of alternatives in the F-E-V space. Although in the worst case, the search space for features will be combinatorial in the number of loops, for most practical purposes, the number of loops is quite small (of the order of the number of features). Besides, it serves to define features in a uniform fashion by grouping features into classes. The loop exploits the geometric nature of features to bundle certain common shape characteristics (concavity and convexity), thereby creating a unified representation.

### 2.4 Obtaining Features from Loops

The process of mapping features based on loop configurations is called Loop-based Feature Mapping [Gadh90B]. A Boundary Loop is defined as being a loop of edges that separates the faces that belong to a feature from those that do not. A feature may have one or more than one boundary loops. For example, the boss has a single boundary loop which is the negative
loop at its base (Figure 2.4). A through hole has two boundary loops, which are the two positive loops on it (Figure 2.4). Using the boundary loop and the filtered edges in the connected component, the faces on the features may be identified from the model.

Figure 2.6: Interacting Features Sharing Faces

Loop-based feature definitions may be used to recognize physically separated or non-interacting features. However, when two or more fundamental features interact, a new feature, called an Interacting Feature, is produced which may not be mappable to any of the fundamental features. An example of such a feature is shown in Figure 2.6 in which a rib and a through hole interact to share a face. This interacting feature may be broken up into its primary features (a rib and a hole) using the concept of Neutral Edge Decomposition (defined in [GadhWA]).

For the part in Figure 2.6, the existence of two positive edges and one negative edge on either side of the face indicate that the face should be split up into two as shown (on the right figure). The edge used to split up this face is termed a Neutral Edge, as it is neither positive nor negative. In effect it is treated as a combination of both positive and negative types, resulting in additional loops being created, which in turn are able to define the two additional features: a through hole and a rib. For purposes of the current research three types of interacting features have been identified: Intersecting Features, Abutment Features (example shown in Figure 2.6), and Recursive Features (see [Gadh90A] for details).

3. Viewing Directions in Filtering

The number of viewing directions, $N_V$, and their direction vector values, $V_j$, $j=1, N_V$, can play an important role in the speed of the feature recognition process. To obtain an understanding of how viewing directions for a filter may be obtained, we draw an analogy of directional filtering to the process of viewing of a part by a human being. If a human being views a part once from a certain direction, the next view would be additionally useful if the relative angular orientation of the part with respect to the human’s eyes was as far away as possible from the initial view. Views should therefore have as much of an angular separation from each other as possible to capture most of the object.

Equispace $V_j$’s will satisfy this criterion, which may be used as an initial hint to determine $V_k (= \{V_j, j=1, N_V\})$.

The problem of selecting $V_j$’s uniformly spaced in $R^3$ may be studied in analogy with a similar problem, that of the location of equispaced points on a sphere, where such a point represents a viewing direction that points towards the center of the sphere. One may imagine these viewing directions as representing points at the face center of a polyhedron with $N_V$ faces that is inscribed in a unit sphere. Therefore, the problem reduces to finding symmetric polyhedra with $N_V$ faces.

However, it is a well known fact that exact symmetry is available in polyhedra for only a single class of solids known as the Platonic Solids. There are a total of five such solids, with 4, 6, 8, 12 and 20 faces, respectively known as tetrahedron, cube, octahedron, dodecahedron and icosahedron. Thus if $N_V$ was 4, 6, 8, 12, or 20, only would we be able to obtain viewing directions that are exactly equispaced.

So, what happens if $N_V$ is not equal to the above values, i.e. it is 1, 2, 3, 5 etc.? For these cases we look for inspiration towards electronic orbital distribution of atoms found in nature. The location of a viewing direction may be treated as analogous to the location of an electron in an electronic shell of an atom. Electrons are distributed spatially according to the principle of minimum energy, which causes them to distribute around the nucleus as far from each other as possible [Thomson23]. For electronic orbits with 1, 2, 3, 5, etc. electrons, the distribution of electrons is not uniform, but represents a minimization of the electronic configuration energy.

Figure 3.1: Definition of $\beta$ in $R^2$ and $R^3$

Therefore, using the electronic orbitals and the Platonic solids analogy, we can set $\Sigma$ given $N_V$. In general equispaced views may not be guaranteed. However, one question that is still unanswered is: What should we select for $N_V$? Needless to say, the number of viewing directions used should depend on the characteristics of the part’s shape features. We use the feature’s smoothness characteristic, $\beta$ (shown in Figure 3.1) to measure its complexity and also to correlate with $N_V$. In $R^2$ space, it has been shown [Gadh90A] that given $\beta$, the minimum value of $N_V$ to reveal a feature is $2\pi/\beta$. 

202
The result \( N_v > 2 \pi / \beta \) is a simple but useful result in \( \mathbb{R}^2 \). It provides us with a powerful mechanism of selecting the viewing directions, \( \mathbf{v} \), given the complexity of a feature, \( \beta \). In \( \mathbb{R}^2 \), for any value of \( N_v \), equispaced \( V_j \)'s can be obtained by dividing the 2-dimensional viewing space of \( 2\pi \) by \( N_v \) to get a number of equispaced views. It is the ability of \( \mathbb{R}^2 \) to get divided into equispaced angles that allows us to get such \( N_v \). By contrast, in \( \mathbb{R}^3 \), a selected value of \( N_v \) will provide equispaced views only if \( N_v = 4, 6, 8, 12, \) or \( 20 \). For an arbitrarily selected \( N_v \), a less than optimum distribution of \( V_j \) must be used.

4. Results of Feature Recognition

The number of approximately equispaced views, \( N_v \), required for recognition of most feature found in manufacturing processes ranges from \( N_v = 4 \) to 14.

Using Neutral Decomposition, we are able to split up most simple intersecting and abutting features into their primary features, and thus they are recognizable. For geometrically intricate interactions, a more sophisticated approach may be required.

Features with a substantial variation of geometry and topology have been recognized. The reduction in the number of topological entities for the parts of complexity shown in Figure 4.1, caused by the filter ranges from 50% to 90%. Inspite of the reduction in the data size being as low as 50% for some parts, the search for features take reasonably small computing time (a few seconds to a few minutes). This may be attributed to fact that the definition of features are in terms of loops, and not in terms of F-E-V adjacency relationships. For simpler geometries, e.g. an upright cylindrical boss on a flat base plate, the data reduction is relatively smaller (50-75%) as compared to more intricate geometries (upto 90%). The BREP model of the fan-like part shown in Figure 4.1 contains approximately 1000 faces, 2600 edges, and 1600 vertices, and requires about 20 minutes for feature recognition.

Subsequent to recognition of features, feature parameters need to be evaluated. The evaluation of feature parameters involves first a separation into the specific feature types: ribs, bosses, etc. This is because feature parameters depend on the feature type. An approach that is currently being actively pursued obtains a measurement of the principal cross dimensions of the part and then maps the feature into a rib-like feature or a boss-like feature [Gadh90A]. The approach is probabilistic in nature and in a single step classifies the feature into a rib or a boss as well as obtaining the feature parameters.

By affecting the various life-cycle considerations, features have a direct influence on cost [Gadh90A]. An example of manufacturability rule in sheet metal is Hole Diameter < X*. If this rule is not satisfied (maybe due to human error at the design stage), it could create a problem in the manufacturing stage, and cause the creation of an engineering change order (ECO) to rectify the design flaw. Propagating this ECO back to the design table could incur additional expense to the corporation. Also, in the event that the product information dictates that the Diameter be greater than X*, the manufacturing process may be altered to suit this condition by using non-standard process conditions or tooling operations, thereby also causing yet another cost to the corporation. Other ways by which features may affect costs indirectly are via tooling, tolerances, production complexity, number of operations, etc.

6. Summary

To reduce concept to manufacturing time of products, there is a need to incorporate knowledge residing in the mind of plant engineers into a system that can analyze designs early on in the design cycle. One of the principal constituents of design that affect such considerations are shape features. The current approach to obtaining features is through feature recognition. Feature recognition because features vary in geometry and topology and the feature search via graph matching is in general computationally expensive. The current
approach to feature recognition using the differential depth perception filter is able to overcome these difficulties for parts of complexity found in industry.

By defining features in terms of loops, as opposed to defining them in terms of low level topological entities, substantial geometric as well as topological variation among features is captured. The space of entities in which search for features is finally performed is loops. The reduction of this search space is first performed by the filter, and then by creating the loops from edges, both of which are significant. This assists in the reduction of combinatorial explosion.

Bibliography


204