SM38: Removal of Blends from Boundary Representation Models

Sashikumar Venkataraman[1]
Geometric Software Solutions Ltd.
Plant 14, Pirojshanagar,
Vikhroli, Mumbai-400079, India
91-22-5960982
sashiv@geometricsoftware.com

Milind Sohoni
Dept. of Computer Science and Engg.
Indian Institute of Technology, Powai
Mumbai-400076, India
91-22-5767729
sohoni@cse.iitb.ac.in

Rahul Rajadhyaksha
Geometric Software Solutions Ltd.
Plant 14, Pirojshanagar,
Vikhroli, Mumbai-400079, India
91-22-5960947
rahulr@geometricsoftware.com

ABSTRACT

This paper reports an algorithm for deletion of blends (or fillets) from Boundary Representation (B-rep) solid models. Blend deletion is usually performed as the first step in feature recognition since it simplifies the model for recognition of volumetric features. The algorithm handles several blend types that include face-face, face-edge and vertex blends. It also handles interactions of blends with other blends and/or volumetric features. Another unique feature of our algorithm is the recreation of new faces in certain situations of blend deletion. Unlike some earlier face-deletion approaches that use geometric heuristics to determine the final topology, our algorithm uses the underlying blend information to directly predict the final topology and geometry. Hence, our algorithm is more efficient and predictable for suppressing large blend networks.

Keywords

Blend recognition, blend deletion, face-deletion algorithms, euler operators, blend chains, feature recognition.

1. INTRODUCTION

Feature recognition is an area of active research in the past two decades [1-5]. Several different approaches have been developed for recognizing features from CAD data. For a comprehensive review of feature recognition, the reader is referred to Refs [4] [5]. These techniques are usually focused towards recognition of volumetric features such as holes, pockets or slots. However, it has been observed that the algorithms presented in the literature do not yet apply to many industrial parts. Most of the examples presented in academic research are usually simple prismatic parts with few feature interactions. It seems a challenge to develop an efficient feature recognition system that works on complex cases containing many curved surfaces and many feature interactions.

Another common occurrence in most industrial parts is the presence of complex blend features (or fillets) that are introduced in the design stage to smoothen the sharp edges of the part. This is mainly done to improve its strength, the aesthetics, and also to ensure the manufacturability of the part. Blend features usually manifest as complex networks that can distort the neighboring topology considerably. In presence of such blend features, most current recognition algorithms fail to recognize volumetric features. Even if volumetric features are recognized, their parameterization is difficult in presence of blends. For example, since the edges of the feature faces are also modified by blends, they cannot be directly used as profiles to create feature volumes. To overcome these problems, feature recognition techniques usually assume that blend features are removed prior to the recognition of volumetric features. However, no comprehensive algorithm has yet been presented to remove blends from a part.

Apart from feature recognition, blend removal is also useful as a local operator for editing and modification of models. For example in applications involving Finite Element Analysis (FEA), blends below a certain threshold radius are removed prior to analysis.

The first step towards blend removal is to recognize the blends in the model. In [7], the authors had proposed an algorithm for recognizing all the blends in a model. The algorithm also deduces the sequence in which the blends were created in the model, and thereby helps in understanding the design intent of the model. The reverse sequence can then be used for blend suppression ensuring valid intermediate models at each step. In that paper, the exact step of blend suppression was performed using a separate function (referred as *delete faces*) that removed a set of faces from a solid by extending/shrinking neighbor faces of the blends. Such functionality is provided as part of many geometry kernels such as ACIS [12] and Parasolid [13], and also as standalone libraries such as by GSSL [14].

Several algorithms have been presented in the literature to delete a set of faces by extending the adjacent faces [8,9,10]. These algorithms are *geometric* in nature and use several heuristics to arrive at the final topology. Usually these approaches work for simple and specific kinds of face-sets, and can fail in complex situations. Especially in case of blends, the number of neighbor faces can be large since the blend chain may run across several faces. This increases the likelihood of failure of a particular fixed heuristic in these algorithms. Further, face deletion algorithms usually involve several intersections between surfaces. In case of blend networks, these intersections could be between tangential (or smooth) surfaces that are unstable. Moreover, in many situations, blend deletion needs to be accompanied by recreation of new faces that got removed during making of the blend. This is not within the scope of current face deletion algorithms. Due to these reasons, many kinds of blends cannot be robustly removed with existing face deletion algorithms.

---

[1] Sashikumar V. is also pursuing a Ph.D. in the Dept of Computer Science and Engg. at the Indian Institute of Technology, Powai.

In this paper, we present a *topological* approach for removing (or suppressing) blend chains from a solid model. This is referred as the *blend suppression algorithm*. The algorithm has several merits when compared with other face deletion algorithms. The main difference is that while other face deletion algorithms use geometry information and heuristics to arrive at the final topology, our algorithm uses the underlying blend structure to directly predict the final topology. The geometry is later computed based on the final topology. The blend structure is determined initially using a blend recognition module as described in [7]. Hence, for many examples involving large blend chains, our algorithm is able to predict the final topology and geometry completely without using any heuristics. In more complex situations wherein the blend features interact with other features, a geometric face deletion algorithm (akin to other face-extension approaches) is used to determine the local topology around the interaction. Even in such cases, the regions needing the face deletion algorithm are smaller and less complex than the entire blend chain and therefore less prone to failure.

Another feature of our algorithm is the ability to avoid tangential intersections during suppression of blend networks. This is done by postponing the geometry computations of intermediate topological entities to the end after suppressing the entire blend network. Furthermore, our algorithm is capable of recreating new faces in certain situations, which is usually not within the scope of other face-deletion algorithms.

The paper is structured as follows. Section 2 discusses some blending concepts that are used in the paper. Section 3 is a review of the blend recognition algorithm that was presented in [7]. The blend recognition algorithm is run prior to blend suppression in order to determine the underlying structure of the blend network. Section 4 presents the blend suppression algorithm and is the main contribution of the paper. Section 5 discusses finer details of the algorithm and some complex situations. Section 6 describes the implementation of the algorithm using local operations. Section 7 presents several examples that are handled by our algorithm. Finally, section 8 presents a conclusion to the paper.

2. BLENDING CONCEPTS

The most common blend is the so-called face-face blend, which replaces a sharp edge by a face tangent to the two faces adjoining the sharp edge. When many blended edges meet at a common vertex, a vertex blend is formed that connects all the neighboring blends smoothly. In special cases, blending can also take place between a face and an edge. Such blends are called face-edge blends or *cliff* blends, and occur as boundary cases of face-face blends. Figure 1 shows an example that contains a few kinds of blend surfaces created while blending a simple part. Further details on the various kinds of blends and blending options can be found in the discussion by Braid [6].

The blend geometry of a face-face blend is computed using an imaginary *rolling ball* that maintains contact with the surfaces to be blended. The blend surface can be visualized as the envelope of this ball as it rolls along the edge. Blending may either add material to or remove material from a model, depending on the convexity of the blended edge. A blend on a convex edge *removes* material from the model, while a blend on a concave edge *adds* material to the model. The locus of points traced by the rolling ball center is called the *spine curve* (which is imaginary); the two edges of contact are called *spring edges*. The side faces of the blend are called the *support faces* since they support the rolling ball. Often during blending, a single blend operation generates many blend faces connected smoothly to each other. Such sets of blend faces created in a single blending operation are referred to as *chains*. The edges that connect adjacent faces in a blend chain are called *cross* edges. Cross edges are usually generated when any of the adjoining support faces of the blend changes during blending. Figure 2 illustrates the spring edge and cross edges for two blend chains. In this figure, the concave blend chain was created after the convex blend. Frequently in real parts, a number of blends interact with each other forming a complex *blend network*. Figure 3 shows an example of a blend network formed in multiple blending operations.

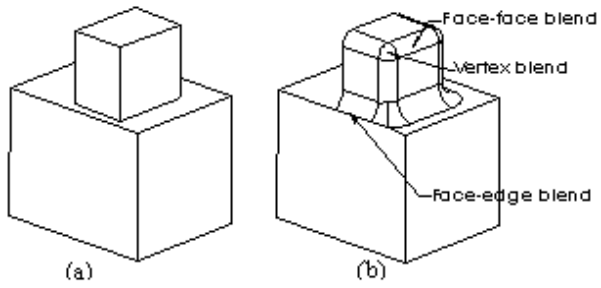

Figure 1 Example of a part before and after blending. The blended part shows the formation of the common kinds of blends.

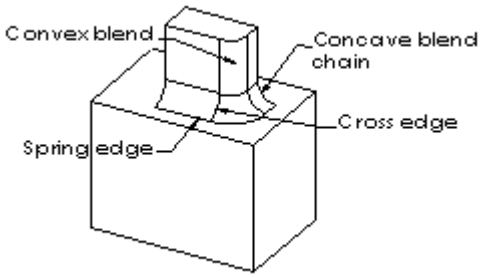

Figure 2 Example showing spring edges and cross edges.

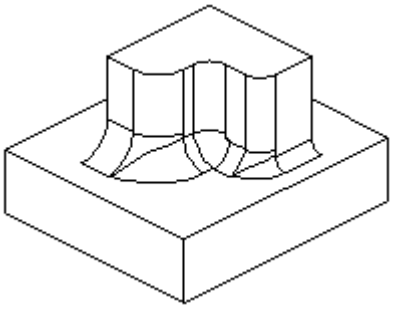

Figure 3 Blends interacting with other blends forming a complex blend network.

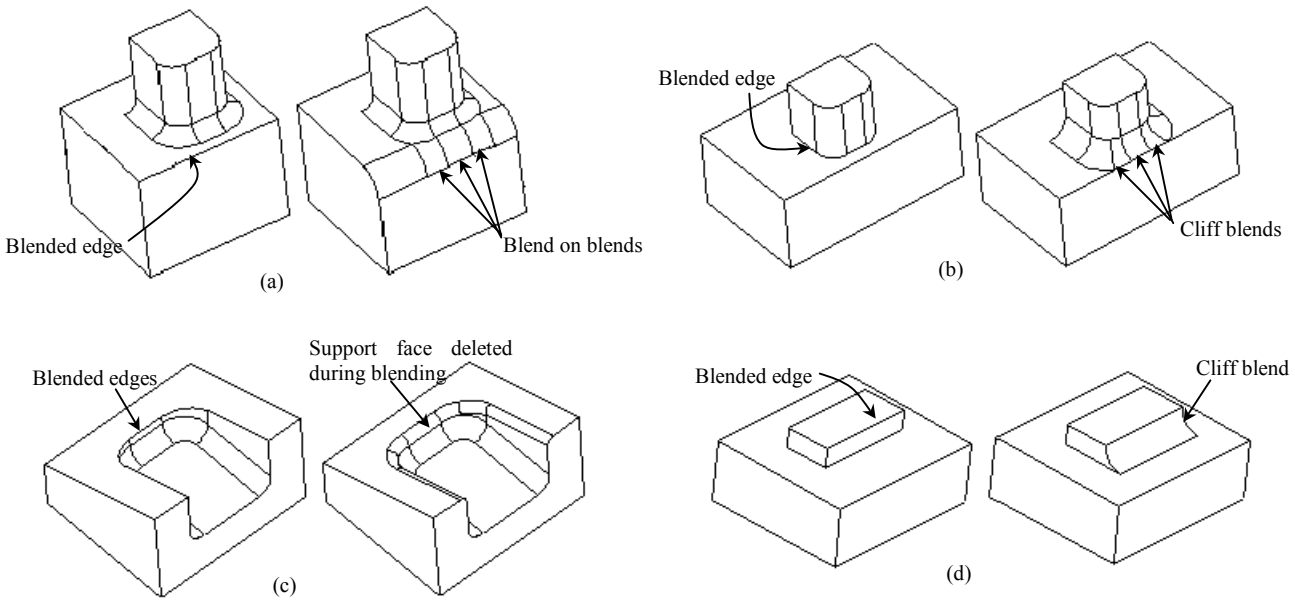

Figure 4 Types of blends based on interactions with neighboring faces. (a) Blend chain interacting with neighboring blends; (b) Cliff blends occurring as part of a blend chain; (c) Support faces lost during blending; (d) Support face lost during creation of a cliff blend.

Based on the nature of the support faces, blends are classified into several categories. When the support faces of the blend are same as the adjoining faces of the blended edge, the corresponding blend is termed as a normal blend. All the blends shown in Figure 2 are examples of normal blends.

In certain kinds of blends, the support face of the blend is not an adjoining face of the blended edge. In such cases, there are two common occurrences that we mainly consider in the paper. In the first case, the blend interacts with another blend chain. Such blends are termed as *blend on blend faces.* The spring edges of the blend on blends are termed as *blend on blend edges*. Figure 4(a) shows the creation of a blend chain got by blending a single edge. Three of the blends shown are blend on blend faces, while the remaining two faces are normal blends.

In the second case, a portion of the blend chain rolls along an edge causing intermediate face-edge blends. These blends have a single support face on one side and an edge curve on the other side. These blends are termed as *cliff blends* and the corresponding edges that support the blends are termed as *cliff edges*. Figure 4(b) shows cliff blends that occur in a part of a blend chain.

In certain situations, blend on blends and cliff blends can cause removal of the support faces of the neighboring blends. Figure 4(c) shows an example of a blend on blend that removes a support face of a neighboring blend during blending. Figure 4(d) shows an example of a cliff blend that removes a face during blending. In such cases, the removed faces need to be reconstructed during the course of blend suppression.

The edges of the blend faces can also be classified into different categories. A blend face typically has a collection of smooth and sharp edges. Spring edges are the smooth edges between the blend and support faces, while cross edges are the smooth edges between the blend face and other blend faces in the same chain. When a blend face is at the end of a blend chain, sharp edges get created. When the blend face terminates on a single face, a single sharp edge is created which is termed as an *isolated terminating edge*. In the example shown in Figure 2, the concave blend chain has two isolated terminating edges, while the convex blend chain has a single isolated terminating edge. When the blend face terminates in multiple faces, multiple sharp edges are formed. Sharp edges also get created due to interaction of the blend with other features. Examples of such sharp edges are shown in Figure 5 below.

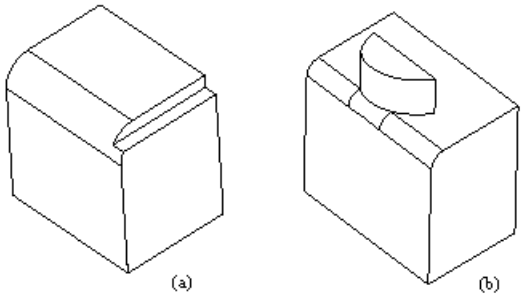

Figure 5 Sharp edges around blend. (a) Blend ending on multiple faces; (b) Boss feature interacting with a blend.

3. BLEND RECOGNITION REVIEW

One of the key features of the blend suppression algorithm is the usage of the internal structure of the blend chain in order to guide the suppression operation. This structure is typically found prior to suppression by a blend recognizer. In this section, we briefly review the blend recognition algorithm that has been presented in [7].

The input to the blend recognition algorithm is typically a B-Rep model. The output is a list of blend chains in the solid along with the corresponding blend parameters. Each blend chain consists of blend faces that could have been created in the same blending operation. The chains are sequenced in an order that denotes a possible sequence that could have been used to create the blend network. The blend faces in a single blend chain are further classified into different *leaf chains*. Leaf chains represent minimal groups of blend faces that need to be necessarily created in one blending operation. Separate leaf chains in the same chain may or may not be created in a single blending operation. Leaf chains usually terminate at sharp edges or vertex blends, and chains are formed by grouping leaf chains across vertex blends.

Figure 6 shows a simple model with some blended edges and the results of the blend recognizer. In this example, Chain 1 consists 3 leaf chains, each of which terminates on one side on vertex blend 7. Chain 2 consists of a single leaf chain that consists of faces 4, 5, and 6. The chain order denotes a possible sequence in which they were created originally in the model, i.e., Chain 1 created first and Chain 2 created later.

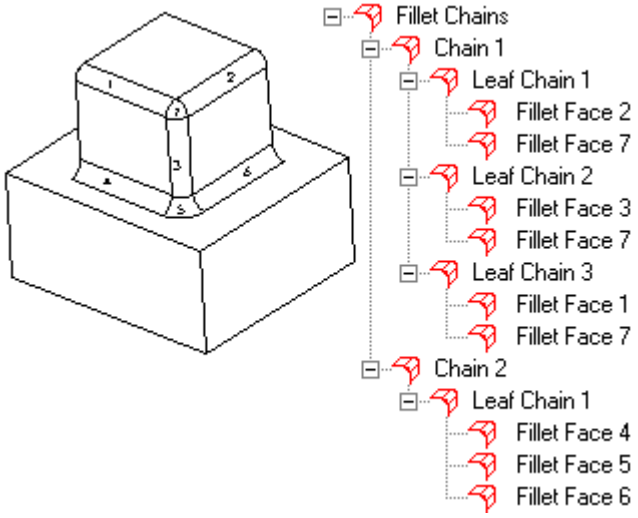

Figure 6 The output of blend recognition for a simple model.

The blend recognition algorithm proceeds in two stages. In the first stage, all the blend faces are detected using local clues. Smooth edges are used as the main clues to classify blend faces. Spring edges and cross edges are distinguished based on surface curvatures evaluated along the edges. Face-face blends have two sets of spring edges running roughly parallel along the face, while face-edge blends have a single spring edge. Further heuristics are used to detect blend on blend edges and cliff edges. Using the classification of edges, it is possible to distinguish normal blends, blend on blends, and cliff blends.

In the second stage, the blend faces are grouped into blend chains. The basic idea in chaining is to group blend faces that are linked via cross edges. Leaf chains are first created that terminate at sharp edges or vertex blends. Chains are then created by grouping leaf chains as per the application's requirement. Finally, using the spring and cross edge relationships, these chains are sequenced in the order in which they got created in the model.

4. BLEND SUPPRESSION ALGORITHM

The blend chains that have been recognized by the blend recognizer are passed to the blend suppressor for deletion. These chains are suppressed in the reverse order of creation so that they result in valid intermediate solid models. At each step, the blend suppression algorithm takes in a single blend chain and suppresses it from the model. For suppression of each blend chain, the blend suppression algorithm first predicts the final topology based on the underlying blend structure, and then computes the geometry for the associated topological entities. This is the key difference from other face deletion algorithms that use several geometry heuristics to arrive at the final topology.

Before presenting the main algorithm, we first illustrate it's working on a simple example. Figure 7 shows a part with a single blend. The blend recognition algorithm is first used to detect the blend chain consisting of a blend face. The blend recognizer also classifies edges s_1 and s_2 as spring edges of the blend face. The edges p_1 and p_2 are classified as the isolated terminating sharp edges of the blend.

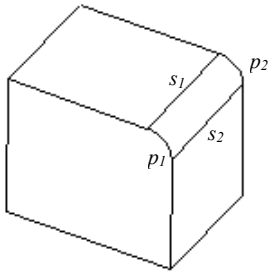

Figure 7 Blend recognition on a simple part with one blend.

The blend chain along with the edge classification is then passed to the blend suppressor for removal. The blend suppression algorithm analyzes the edges of the blend faces, and outputs a sequence of operations to modify the topology. In the example above, the sequence consists of the following two steps:

Step 1: Collapse the sharp edges p_1 and p_2 into vertices. After this step, the blend face consists of only two spring edges as shown schematically in Figure 8(a).

Step 2: The blend face consisting of two edges is collapsed into a single edge. The final topology of the model is shown in Figure 8(b).

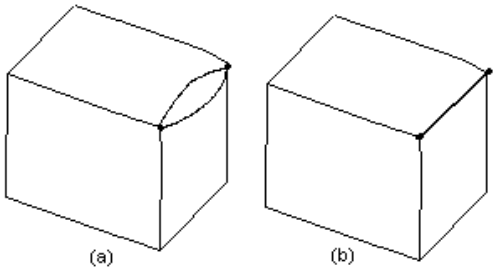

Figure 8 Intermediate topological states during blend suppression.

After fixing topology, the geometry of the edge and vertices are calculated. The edge geometry is first found by intersecting the two support faces of the blend feature. The geometry of the end vertices is then computed by intersection of the created edge curve and the neighboring surface.

The above technique can be used to suppress blend chains that consist of normal blends. Even in case of blend on blends and cliff blends, a set of rules is used for directly predicting the final topology. However, when a blend face has multiple sharp edges due to interaction with other features as in Figure 5, the final topology cannot be predicted in this manner. In such cases, the blend suppression algorithm invokes a face deletion algorithm for the region near the interaction. The face deletion algorithm uses geometry heuristics to determine the final topology around the interacting region.

The overall algorithm of blend suppression is described below.

Algorithm : Blend suppression

Input : A b-rep model and a blend chain *C*

Output : Modified model with the blend chain suppressed

Procedure DeleteBlends (Chain *C*: faces $f_1, f_2, f_3, \ldots$)

(1) Separate blend chain from neighboring blends along blend on blend edges and cliff edges if any.

(2) Recreate new support faces if necessary.

(2.1) Recreate missing faces adjoining blend on blends.

(2.2) Recreate missing faces adjoining cliff blends.

(3) Delete blend faces of the chain

(3.1) Collapse cross edges and isolated terminating edges into vertices

(3.2) Collapse each face that consists of two spring edges into an edge

(3.3) Compute geometry for created edges and vertices

(4) Solve sharp edges arising due to feature interactions and multiple end faces.

End DeleteBlend

In the above procedure, steps 1, 2, 3.1, and 3.2 modify topology, while steps 3.3 and step 4 compute geometry. The individual steps of the algorithm are explained in detail in the following sections.

4.1 Resolving blend on blend and cliff edges

The first step in the blend suppression algorithm is to resolve blend on blend edges and cliff edges. Blend on blends are topologically separated from neighboring blends as shown in Figure 9. Cliff blends are also separated in a similar manner as shown in Figure 10 below.

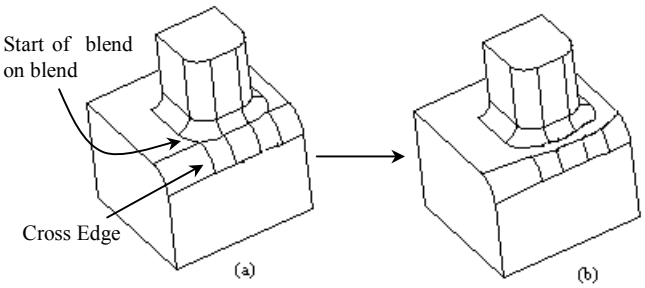

Figure 9 Separating blend on blends from neighboring blends.

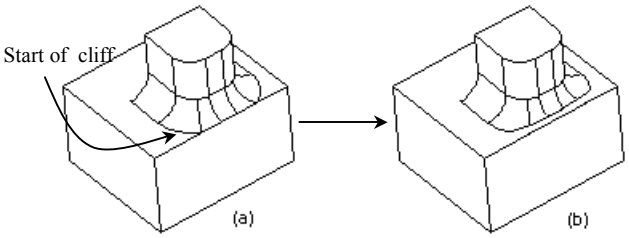

Figure 10 Separation of blend chain at cliff edges.

The vertex to begin the split on blend of blends is determined by the blend recognition module using the clue of the cross edge shown in the figure. Similarly, the start of the cliff is determined using the transition from a normal blend to a cliff blend.

4.2 Recreate support faces

In certain situations, blend on blends and cliff blends can cause removal of the support faces of the neighboring blends. Figure 4(c) shows an example of a blend on blend that removes a support face of the neighboring blend. The blending of the top edges completely removed one side face of the pocket. Suppression of the top chain must hence be accompanied by the creation of the removed face. In general, it is difficult to predict the faces that need to be recreated during blend suppression. However, in many situations, local clues aid in predicting the new faces. This is explained in detail in this section.

Figure 11 shows edges and faces adjacent to the removed faces in Figure 4(c). The bottom and top chains are first recognized by the blend recognizer. The cross edge *C* provides the clue that the top chain was introduced later, and hence should be suppressed first. Two cross edges of the bottom chain are not explained by any of the adjoining faces. This denotes that one face got removed when the top chain was introduced, and hence needs to be recreated during suppression. Further, faces adjoining to the missing faces are determined by analyzing the junction between the two chains. These faces are referred to as *trace faces*. In Figure 11, face *T* is one of the trace faces. A similar trace face is also present in the other end of the blend chain.

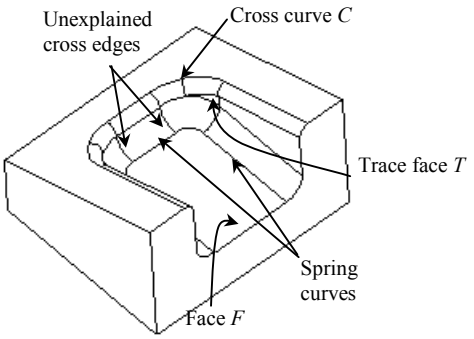

Figure 11 Faces and edges shown around missing region.

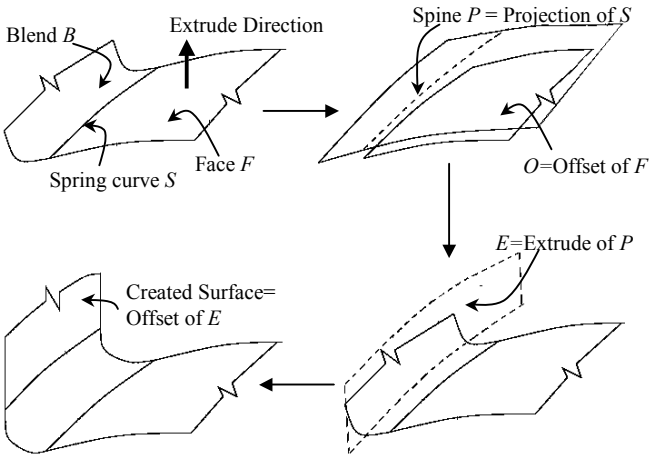

Figure 12 Steps in creation of missing extrude faces.

The basic idea in face recreation is that in certain situations the spring curves on one side of the blend and the trace faces are sufficient to predict precisely the surfaces on the other side. In the above example, the trace faces are found to be extrude surfaces in a particular direction. Hence, the missing faces are concluded to be part of the same extrude feature. The exact geometry can then be computed systematically by using the spring curves on the other side of the blend as explained below.

Let r be the blend radius of the bottom chain. The following steps are carried out for each of the spring edge between the unexplained cross edges in the bottom chain to recreate the corresponding missing face.

1. Offset the bottom support face F by r to create surface O.
2. Project spring edge S of the bottom chain to the offset surface O to create the spine P of the bottom chain.
3. Sweep the spine P in the extrude direction to create the extrude surfaces E. Note that P, in general, is not a planar curve.
4. Offset E by r to create the missing face.

Figure 12 depicts the creation of a single missing face using the above steps. The above analysis is specific for recreating missing faces that are part of an extrude feature. A similar analysis also holds when the missing faces are known to be a part of a revolve feature. In general, if the type of the missing faces can be inferred from the trace faces, the missing faces can be recreated for the corresponding paradigm.

In certain situations, the missing faces cannot be recreated unambiguously. Figure 13 shows an example of a cliff blend in which no "official" trace face is present. In such cases, certain heuristics are used for recreating the missing face. In this example, the extrude face T is used as a clue face, and the geometry of the missing support face is found by sweeping the sharp edge S_1 in the corresponding extrude direction.

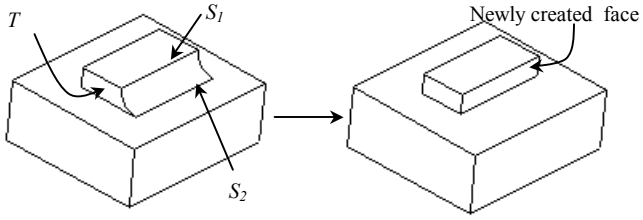

Figure 13 Recreation of faces during suppression of cliff blend

4.3 Remove blend faces

At this stage, the blend chain is isolated from neighbor blend chains and sharp edge interactions. The blend chain can then be deleted using a simple procedure. In the first step, all cross edges and isolated terminating edges are collapsed into vertices. After this step, all the blend faces become two-edged faces corresponding to the two spring edges. In the next step, the face is collapsed into an edge between the corresponding support faces of the spring edge. This operation is shown in Figure 14 for the concave blend chain of Figure 2.

After topology modification, the geometry is computed for all the created topological elements of the previous steps. The geometry of the created edges is computed by intersecting the corresponding support faces. The geometry of the created vertices is then computed by intersecting the edge adjoining the surface and a face touching the edge at the vertex. In the above example, the geometry of the vertices is obtained by intersecting the vertical edges with the horizontal planar face.

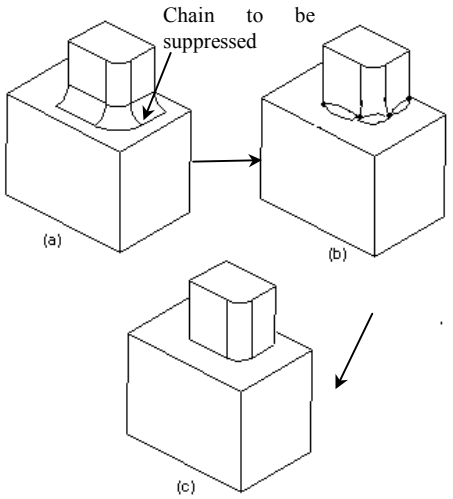

Figure 14 Stages during suppression of a blend chain.

4.4 Solving sharp edges

As discussed in the beginning, there are broadly two approaches for deleting blends, namely geometric and topological. Earlier methods such as described in [8,9,10] use the geometric approach that includes face extensions and geometric heuristics to determine the final topology. These algorithms could fail in complex situations when the number of neighboring faces becomes large. Our algorithm uses the topological approach that directly predicts the final topology based on the underlying blend structure. In the steps 1 to 3 of the algorithm, a set of rules is used to determine the final topology from the blend structure without using any heuristics. Using these steps, all blend faces that contain only spring edges, cross edges and isolated terminating edges are removed.

However, in certain situations such as in Figure 5 wherein blends interact with other features along sharp edges, the final topology around the interacting region cannot be predicted by only using the blend topology. In such cases, the blend suppressor makes a call to a geometric face-deletion algorithm to determine the final topology around the interacting region.

Frequently, one encounters a large blend chain that contains specific regions of sharp edges due to feature interactions such as in Figure 15(a). One method to delete the blend chain is to pass all the faces of the chain to the geometric face-deletion algorithm. However, the operation can be made more reliable by localizing the geometric approach to the vicinity of the interacting region. The remaining region is resolved using the topological approach (steps 4.1 – 4.3) that is more predictable in nature. For example in Figure 15(a), only a single blend face (shown in a grey color) is passed to the face deletion algorithm. The other blend faces are deleted using steps 4.1 – 4.3 of the algorithm. Figure 15(b) shows the intermediate topology after these steps and prior to calling the face deletion algorithm. Figure 15(c) shows the final topology after the face deletion algorithm.

In certain situations, it is not necessary even to pass the entire blend face to the face deletion algorithm. A portion of the face can be deleted using the topological approach whereas the remaining portion is deleted by the face deletion algorithm. An example of such a case is shown in Figure 16 where the blend ends on multiple end faces. The blend face is split into two faces by topologically inserting a "cross" edge between the two spring edges. The blend face without sharp edges is deleted using the topological rules, while the other face is deleted by the face deletion algorithm. Figure 17(a) shows the intermediate step after deleting the normal blend. Figure 17(b) shows the model after deletion of the other face by the face deletion algorithm.

Figure 18 shows the working of our geometric face deletion algorithm for the above part. The details of this algorithm are considered in a separate paper. The boundary loop of the region to be deleted is first determined. The neighboring edges touching the boundary loop are also found. These edges are referred as the *external edges* and play an important role in the face deletion algorithm. The algorithm proceeds by extension or contraction of the neighbor faces and patching up the deleted region. Using geometric techniques, the algorithm determines the topological connectivity relationships of the neighboring faces within the boundary loop in the form of a solution graph as shown in Figure 18(b).

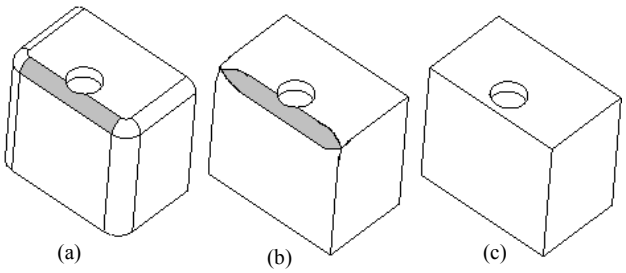

Figure 15 Blend chain with sharp edge interactions; (a) Blend face interacting with a hole; (b) Intermediate topology after suppression of the remaining blends; (c) Final topology after face deletion algorithm.

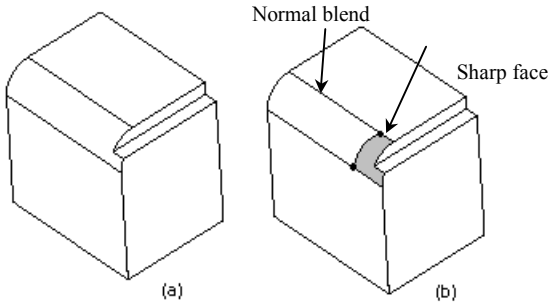

Figure 16 Isolating the set of sharp edges from the blend.

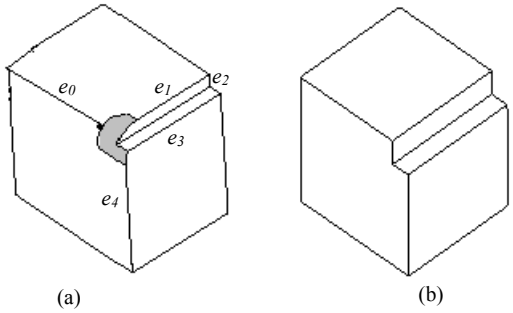

Figure 17 Creation of problem for the delete face operation for the sharp edge set. External edges shown for the face deleted.

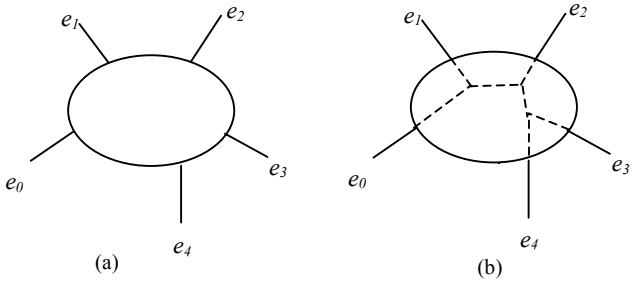

Figure 18 Solving the loop problem. (a) Solution graph obtained after delete face operation. (b) Final part obtained after deleting the blend face.

5. ADVANCED ISSUES

In this section we discuss finer details of the blend suppression algorithm and demonstrate the working of the algorithm in specific situations. We first discuss the concept of blend clustering that helps in avoiding tangential intersections during suppression. We then discuss the suppression of some special blends that were not considered in the last section. These include blend chains with vertex blends and degenerate cases.

5.1 Blend Clustering

In practice, complex blend networks are formed due to blend chains interacting with other blends, and it is usually required to remove all the blend chains. In such situations, the geometry computations associated with many intermediate topological entities can be avoided by postponing them to the end after suppressing the entire network. This grouping of interacting blend chains is termed as *blend clustering*. Apart from increasing speed, blend clustering is particularly useful when the intermediate edges are formed between faces that meet tangentially. The geometry computation for such edges involve tangential intersections that are known to be unstable. These computations are avoided using blend clustering.

For example, in Figure 19 below, chain C_1 is deleted. New edges are created for the chain C_2 that need to be associated with geometry. Moreover, these edges are between faces that intersect tangentially. The geometry computation for these edges is avoided by postponing it to the end of suppression of both the blend chains. For example in Figure 20(a), both the blend chains C_1 and C_2 are grouped in a single blend cluster. During suppression, chain C_2 is deleted immediately after chain C_1 without computing the geometry of the intermediate created edges. The final model after suppressing the blend cluster is shown in Figure 20(b).

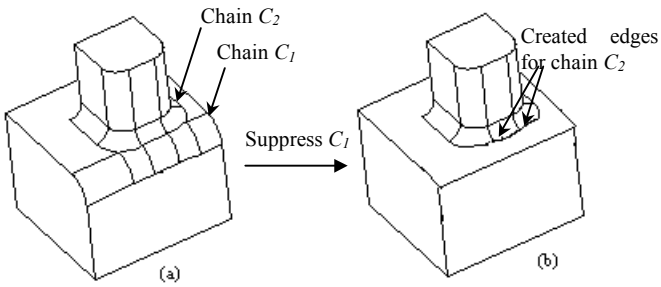

Figure 19 (a) Example showing deletion of blend on blend; (b) Created edges formed for the other blend chain.

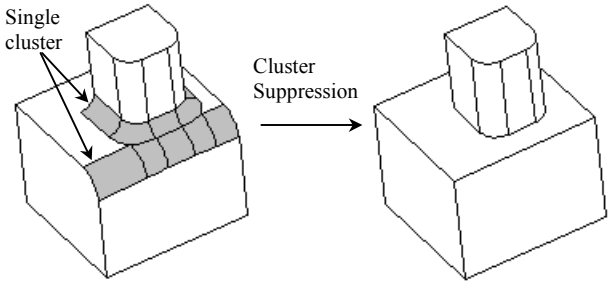

Figure 20 Suppression of blend cluster in a single step.

5.2 Vertex Blends

As discussed in section 3, a blend chain can consist of several leaf chains. Leaf chains represent minimal groups of blend faces that need to be necessarily created in one blending operation and hence denote the smallest set of faces that can be suppressed in a single blend suppression operation. Leaf chains usually terminate at sharp edges or vertex blends, and chains are formed by grouping leaf chains across vertex blends.

During blend suppression involving a vertex blend, there are two cases. In the first case, all the blend chains around the vertex blend are passed for suppression. In this case, all the cross edges collapse to vertices, and the vertex blend collapses into a face with a single vertex. This face can then be removed from the part. Figure 21 shows the collapsing of a vertex blend during the suppression of a chain consisting of three leaf chains.

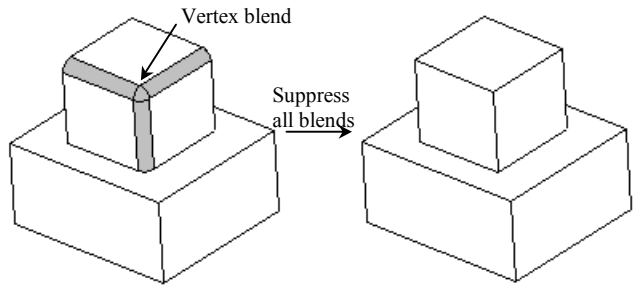

Figure 21 Example showing suppression of all leaf chains around a vertex blend. The vertex blend collapses to a point.

It is also possible to suppress a single or a partial set of leaf chains around a vertex blend. In such cases, the final topology cannot be fully predicted near the vertex blend, and the face deletion operation is used to determine the topology in the neighborhood. Figure 22 shows the model obtained by suppression of a single leaf chain. In this example, a *mitre* edge is formed by intersection of the other two adjoining blends.

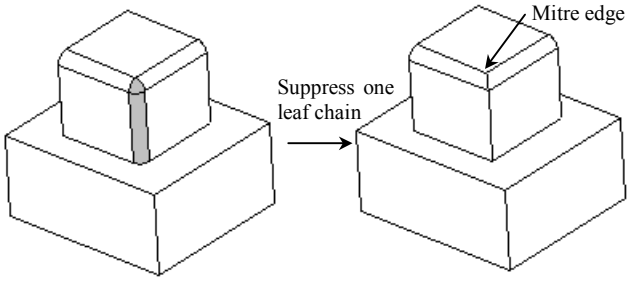

Figure 22 Example showing suppression of a single leaf chain. The vertex blend is resolved using the delete face operation.

The face deletion operation is also used to suppress leaf chains that intersect each other without the presence of a vertex blend. Figure 23 shows the suppression of such a blend chain using the face deletion operation. The intermediate model prior to the face deletion algorithm is also shown.

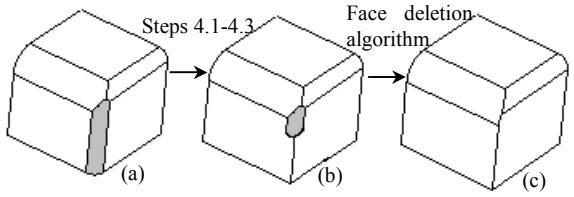

Figure 23 Suppression of a leaf chain without a vertex blend using the face deletion operation.

5.3 Degeneracy and other complex situations

Another aspect that has not been discussed so far is the treatment of degeneracy. Figure 24(a) shows a blend that has a protrusion feature touching the spring edge forming a degenerate vertex. The blend suppressor splits such degenerate vertices at the beginning of the algorithm as shown in Figure 24(b).

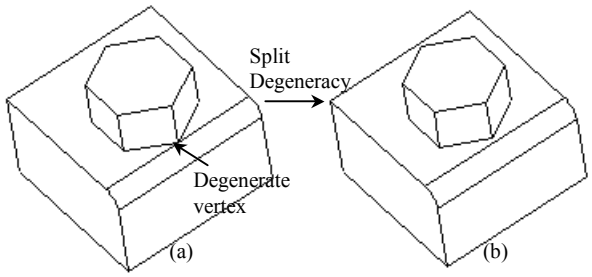

Figure 24 Splitting of degeneracy at a vertex prior to blend suppression.

In certain cases, degenerate vertices get created after suppression of a blend chain. These arise from zero length edges created during blend suppression. For example, Figure 25(a) shows a rib with two blend chains. During suppression of blend chain C_1, an edge is created between the blend face of chain C_2 and face P. This edge is detected as a zero length edge, and is deleted by the blend suppressor finally to create a degenerate vertex as shown in Figure 25(b).

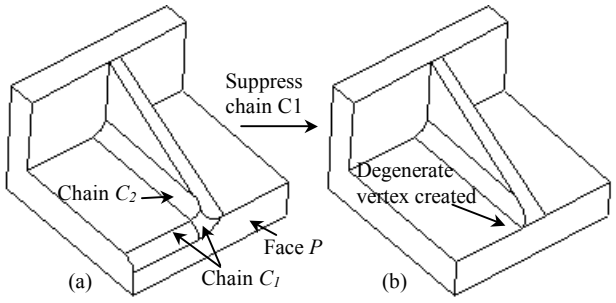

Figure 25 Creation of degenerate vertex that result after blend suppression.

An important assumption of the blend suppression algorithm is that after suppression of a blend face, an edge is created between the corresponding support faces. This assumption is usually true as long as the blend radius is small compared to the blended edge. However, in certain extreme situations where this assumption is violated, the algorithm may predict the wrong topology. Figure 26(a) shows a small blend face between faces F_3 and F_2 in which a blend radius is relatively large compared to the blended edge. The blend suppression algorithm predicts wrongly, and creates an edge between F_3 and F_2. However, in reality the edge is between F_1 and F_4 as shown in Figure 26(b). After suppression, the wrongly predicted edge is detected to overlap with the neighboring edges, and is then locally corrected.

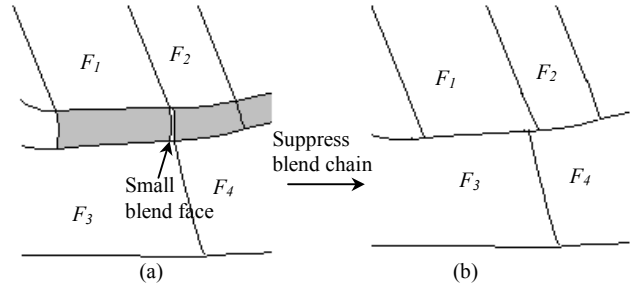

Figure 26 Suppression of a leaf chain without a vertex blend using the delete face operation.

In certain situations, naïve blend suppression may give rise to invalid bodies due to global interactions with other faces. For example, in the shelled model in Figure 27, the outer and inner fillets are not related directly, but suppression of the inner blends first would make the model invalid.

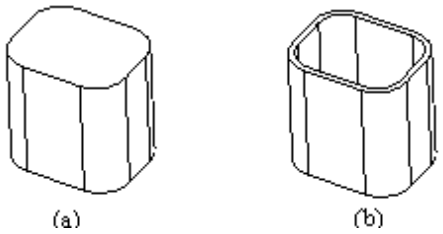

Figure 27 Blend sequencing may need global analysis; (a) Blended model; (b) Shelling performed after blending.

To handle such situations, a test is performed to determine if a suppression of a fillet causes the adjacent faces to intrude other non-local faces of the model. The inner fillets of the shell would fail the test since the extension of the neighboring fillets would penetrate the outer fillet. Hence, they would not be suppressed till the outer fillets are suppressed.

6. IMPLEMENTATION

The blend suppression algorithm has been implemented as a module of a Feature Recognition library at Geometric Software Solutions. Ltd [14]. This module is implemented in C++. The module is available currently on the Parasolid, and can also be ported easily on other geometry kernels. This section touches on some of the implementation aspects of the blend suppression algorithm.

The implementation uses local operations in the form of *Euler Operators* [11] to modify topology locally. Euler Operations are low-level functions that modify a small region of topology. Using these operators, topological elements such as faces, loops, edges and vertices may be added, removed, or modified in a model. Together with functions to attach and detach geometry, these functions enable applications to implement their own modeling operations, such as local operations and feature creation. Euler Operation functions always return a body with valid topological data-structures. However, the functions do not alter geometry - new topology has no geometry attached, and any topology that is deleted has its geometry deleted first. This means that the resulting body is normally invalid. Geometry is later associated with the model to make the model geometrically valid.

The Euler operators can be divided two groups; the *make group* and the *kill group*. The make group consists of operators for adding some elements into the existing model, while the kill group does exactly the inverse of the make group. For example, MEV is an operator that makes an edge and a vertex, while KEV is an operator that kills (or deletes) an edge and vertex. Similarly, MEF is an Euler operator that makes an edge and face and KEF is the corresponding inverse operator.

During blend suppression, cross edges and isolated sharp edges are merged using the KEV operator that deletes an edge and merges the end vertices. The blend face is then collapsed into an edge using the KEF operator to collapse a 2-edged face into an edge. This is shown pictorially in Figure 28.

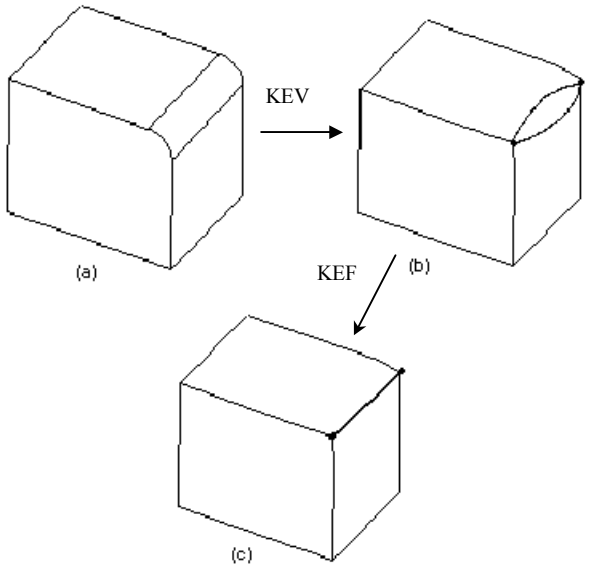

Figure 28 Usage of Euler operators for removing simple blends.

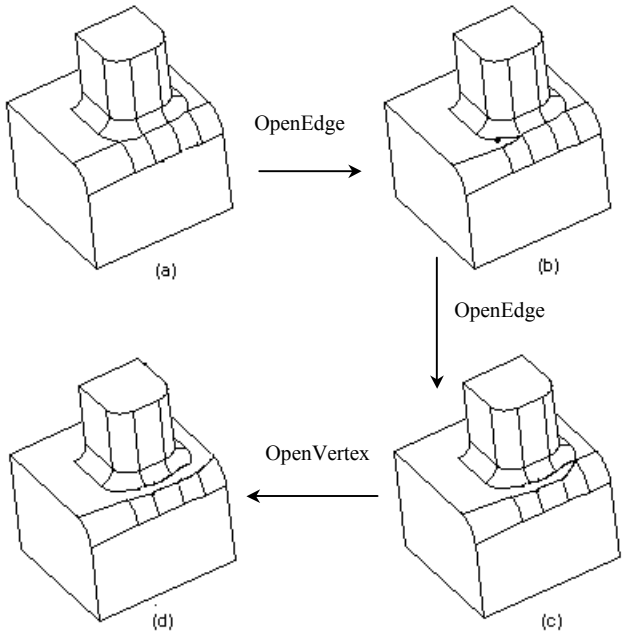

Figure 29 Usage of specialized Euler operators for separating blend on blends.

More specialized forms of Euler operators are used for suppressing blend on blends and cliff blends. Figure 29 shows the sequence of steps used to separate blend on blends in step 1 of the blend suppression algorithm. The *OpenEdge* operator is used to separate an edge into two edges at a vertex. Finally, the OpenVertex operator is used to split a vertex into two.

The implementation of the blend suppression algorithm proceeds in three stages. The first stage performs an analysis of the whole network and creates a sequence of operations that need to be performed to suppress the blend chain. After the analysis, a validation is done to ensure that each edge of the chain is accounted by some operation. If all edges are not accounted, then the function returns failure denoting that suppression cannot be performed. The input model is not modified in this stage.

The second stage modifies the topology according to the sequence of suppression operations found in the first stage. Each suppression operation makes calls to corresponding Euler operators and stores the created topological entities in specific classes. For example, the operation that collapses a cross edge stores the created vertex in the corresponding class.

In the third stage, geometry is associated with the new/modified topological entities. Typically, most of the time taken by the algorithm is spent in the last stage that involves geometry computation. The face deletion algorithm is implemented as a separate module and is called in the last step of the blend suppression algorithm. If a failure is encountered in any step, the operations are undone, and the model is restored back to the original state.

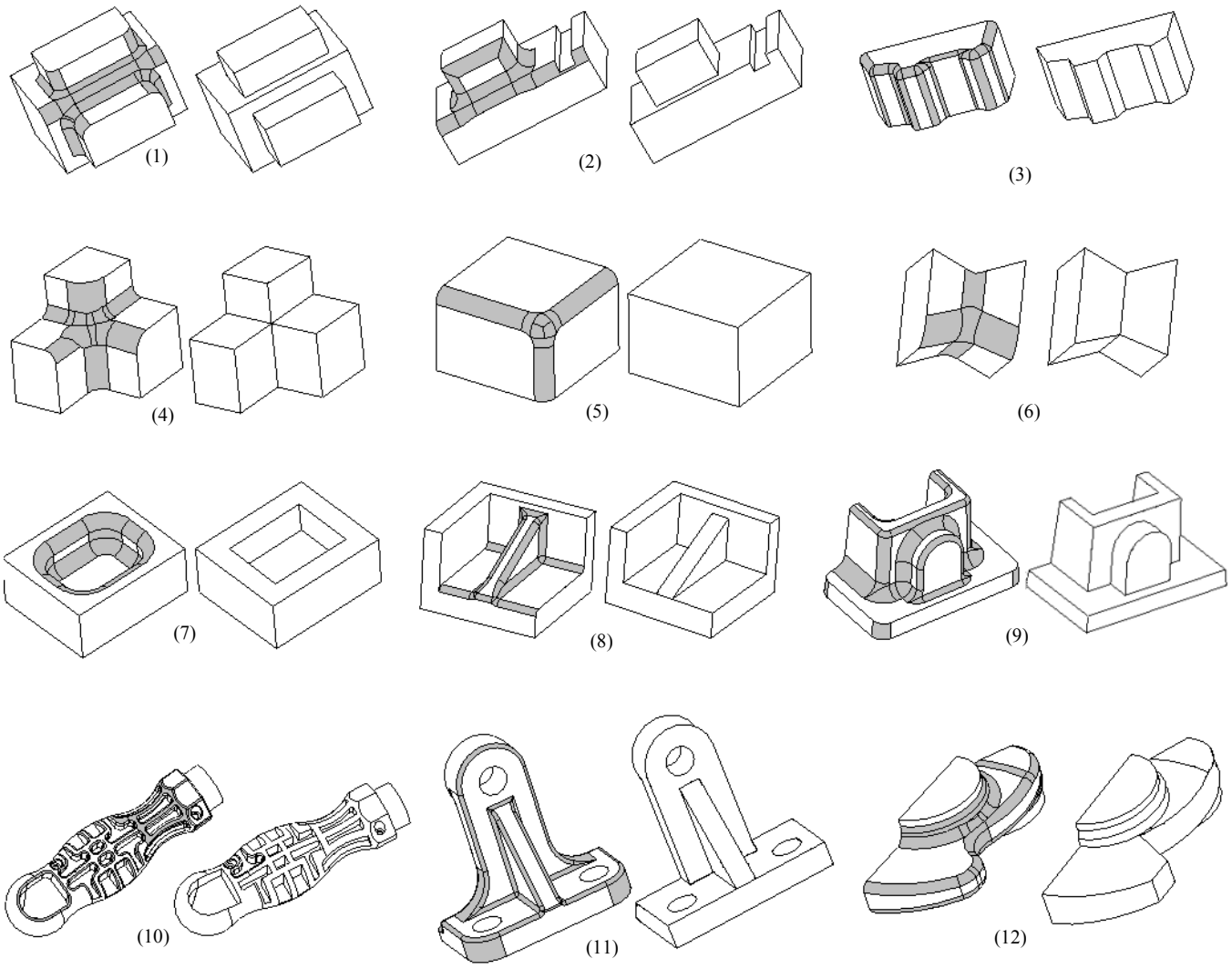

Figure 30 Examples showing the working of the blend suppression operation in 12 parts.

7. EXAMPLES

Figure 30 shows several examples that demonstrate the working of the delete face algorithm in several situations. Each example shows the input model and the final model after the delete face algorithm. The blend faces passed for deletion are shown in grey.

Example 1 shows a sequence of three blend chains. In this example, all the chains are grouped in a single blend cluster, and are suppressed sequentially. Example 2 shows a blend cluster interacting with a slot feature. Our algorithm predicts the topology around the blend on blend, and uses the face deletion algorithm near the slot feature. Example 3 shows a long blend chain rolling over several faces that is suppressed fully in a predictable manner. It may be noted that examples 1, 2, and some others are not handled by any known geometry kernel.

Example 4 shows blends at a complex junction where six edges meet. The blend suppressor removes all the blends and remakes the 6-edged vertex. Example 5 shows a non-conventional vertex blend that is deleted successfully. Example 6 shows blends near a degenerate vertex. The blend suppressor recreates the degeneracy using the face-deletion algorithm.

Examples 7, 8 and 9 shows complex blend networks around pocket and rib features. The blend suppressor uses the results of the blend recognizer to suppress the blend chains in the right sequence. All these blends are suppressed in a fully predictable manner without using the face-deletion algorithm.

Examples 10, 11 and 12 are real industrial cases with many blend chains. These examples have many freeform surfaces and occurrences of blend on blends. The blend suppressor has been used to removal all the blends in these parts

8. CONCLUSION

This paper presents a specialized face-deletion algorithm for removing blend chains from a solid model. Blend suppression is used in many applications in the pre-processing stage to simplify parts such as in feature recognition and finite element analysis.

Our algorithm has several merits when compared with other face deletion algorithms. Earlier proposed algorithms for face-deletion are mainly geometric in nature. These algorithms use face extensions and geometry heuristics to arrive at the final topology. Such approaches usually work for simple and specific kinds of face-sets, and fail in complex situations. In contrast, the algorithm presented in this paper is mainly topological in nature in which the final topology is directly predicted using the underlying blend structure. The blend structure is found using the blend recognition module as presented in [7], and hence blend suppression is integrated with blend recognition. In our approach, heuristics are eliminated for most kinds of blend networks, and the result is more predictable. In complex situations wherein the blend features interact with other features, a geometric face-deletion algorithm is used to determine the topology around the interacting region. Even in such cases, the regions needing the face deletion algorithm are smaller and less complex than the entire blend network. The remaining region is resolved using the normal topological rules of the algorithm.

During suppression of large blend networks, the blend suppression algorithm groups blends into blend clusters. Geometry computations are postponed till all the blend chains in the cluster are removed. This technique reduces the computations performed, thereby increasing the speed of suppression. This also avoids tangential intersections that are prone to instability.

Another unique feature of our algorithm is the capability of recreating new faces in certain situations. The special case of recreating extrude faces has been demonstrated in this paper. This can also be used for other surface types such as surface of revolution.

The blend suppressor has worked successfully on a wide variety of test cases. Some of the test cases include industrial parts with freeform surfaces with large blend networks. Several cases containing vertex blends and degeneracy are also handled robustly by our algorithm.

9. ACKNOWLEDGMENTS

We would like to thank Geometric Software Solutions Ltd. (GSSL) for extending co-operation in promoting this research. We would like to specially thank Dr. Vinay Kulkarni for his valuable suggestions and encouragement in this project.

10. REFERENCES

[1] Joshi. S. B., and Chang, T. C., Graph-based heuristics for recognition of machined features from a 3D model. *Computer Aided Design*, 1988. Vol. 20(2), 58-66.

[2] Little G., Clark D. E. R., Corney J. R., and Clark D.E.R., Tuttle J.R., Delta-volume decomposition for multi-sided components. *Computer Aided Design*, 1998. Vol. 36(9), 695-705.

[3] Gao S., and Shah J. J., Automatic recognition of interacting machining features based on minimal condition subgraph. *Computer Aided Design*, 1998. Vol. 30(9), 727-739.

[4] Shah, J.J., Manyla, M. amd Nau, D., eds. *Advances in Feature Based Manufacturing*, Elsevier/North-Holland, Amsterdam. 1994.

[5] Regli W.C., Geometric algorithms for recognition of features from solid models, *PhD thesis*, University of Maryland, 1995.

[6] Braid I. C., Non Local Blending of Boundary Models. *Computer-Aided Design*, Vol 29, No 2, February 1997.

[7] Sashikumar V., Sohoni M., Blend Recognition Algorithm and Applications. *ACM Symposium on Solid Modeling and Applications*, June 2001, Ann Arbor, Michigan.

[8] Dong, X. and M. Wozny, "A method for generating volumetric features from surface features," *ACM Symposium on Solid Modeling and Applications*, 1991.

[9] Sakurai, H. and D. Gossard, "Recognizing Shape Features in Solid Models," *IEEE Computer Graphics & Applications*, September, 1990, pp. 22-32.

[10] Sandiford D., Hinduja S., Construction of feature volumes using intersection of adjacent surfaces. *Computer Aided Design,*2001. Vol 33, pp 455-473.

[11] Mäntylä, M. (1988). *An Introduction to Solid Modeling. Principles of Computer Science*. Computer Science Press, Maryland, U.S.A.

[12] ACIS Geometric Modeller, Format Manual, Version 6.0, Spatial Technologies, (www.spatial.com) January 2000.

[13] Parasolid, Functional Description Manual, Version 12, Unigraphics Solutions, (www.parasolid.com) May 2000.

[14] Feature Recognition Library, Version 11, Geometric Software Solutions Limited, (www.geometricsoftware.com) October 2001.