Automatic Update of Feature Model After Direct Modeling Operation

Xiaolian Qin[1], Zhihong Tang[2] and Shuming Gao[3]

[1]State Key Laboratory of CAD&CG, Zhejiang University, qqz003@163.com
[2]State Key Laboratory of CAD&CG, Zhejiang University, tang_zhi_hong@126.com
[3]State Key Laboratory of CAD&CG, Zhejiang University, smgao@cad.zju.edu.cn

Corresponding author: Shuming Gao, smgao@cad.zju.edu.cn

Abstract. Feature-based modeling and direct modeling are the two mainstream CAD modeling methods complementary to each other. Although major engineering CAD vendors has begun to push out direct modeling module in their products, integration between the two modeling methods is still an open issue. The key technical issue to be resolved is: how to determine the updated feature model after direct modeling operations are conducted? This paper proposes an approach to resolving this issue, consisting of determination of feature volume variations, generation of candidate update operations and determination of optimal update operation. The experimental results show the effectiveness of the proposed approach.

Keywords: automatic update, direct modeling, parametric and feature-based modeling, design intent maintenance, CAD
DOI: https://doi.org/10.14733/cadaps.2021.170-185

1 INTRODUCTION

Feature-based modeling and direct modeling are the two mainstream CAD modeling methods. Feature-based modeling has been widely applied in commercial CAD software in the past decades. It constructs a history-based feature model of the product, based on which the B-rep model is further generated by Boolean operations [7]. Feature-based modeling is powerful for design intent maintaining and downstream application, but has high requirements on users' modeling ability [6]. Direct modeling is an emerging CAD modeling technology that uses direct manipulation of the geometry to effectively change B-rep models [1]. It is intuitive and flexible, having no burden of history-based dependencies and thus providing users with convenience to make local modifications to existing B-rep model. It can also be easily applied to the touch screen and the VR equipment [2]. The flexibility of direct modeling also makes it easier to destroy the existing design intent.

In general, the two modeling methods are complementary to each other. Feature-based modeling has a good performance in embodiment design stage, while direct modeling behaves well in optimal design stage. So major engineering CAD vendors have added the direct modeling module in their products such as NX [8], Creo [3], and CATIA [4]. However, the integration between feature-based modeling and direct modeling is still an open issue. For instance, in Creo, direct modeling is integrated with feature-based modeling by simply regarding each direct modeling operation as a new feature and adding it at the end of the modeling history. Obviously, such integration does not make sense because the direct modeling operation is not a feature as it does not contain any specific engineering semantics. In addition, such integration makes the feature model difficult to be used in downstream applications.

The main problem of the integration is how to update the feature model after a direct modeling operation, keeping the consistency between the updated feature model and the B-rep model modified by the direct modeling operation. Fu et al [5] firstly put forward this issue and presented a method based on cellular model. In their method, model validity is guaranteed by applying cellular model which is rarely used in mainstream CAD systems. And only extrusion features are taken into consideration, lacking the ability to handle more complex models. Zou et al [10] presented an approach to variational B-rep model analysis for direct modeling using geometric perturbation. The approach focuses on 3D geometric constraint analysis of the model after direct modeling operations, not considering feature model updating after direct modeling operations.

In order to achieve effective integration of feature-based modeling and direct modeling, we propose an approach to the consistent updating of the feature-based model after direct model operations are conducted. The approach consists of three critical steps: determination of feature volume variations, generation of candidate update operations, and determination of optimal update operation.

The rest of the paper is organized as follows. Section 2 gives an overview of the proposed method. Section 3 gives detailed description of the critical steps in the method. The experimental results and discussions of the method is presented in Section 4. Conclusions are given in Section 5.

2 OVERVIEW

The problem to be resolved in this paper is stated as follows.

Problem Given a feature model M_{f} and its associated B-rep model M_{b}, after direct modeling operations O are executed with a new edited B-rep model M_{b}' obtained, determine the optimal updated feature model M_{f}' consistent with M_{b}'.

Figure 1 illustrates the problem. As shown in Figure 1(a), the original feature model consists of two features, and the corresponding B-rep model is shown in Figure 2(b). A direct modeling operation moving a face to a new position is given in Figure 1(c). Four candidate updated feature models corresponding to the direct modeling operation are shown in Figure 1(d-g). From design perspective, an ideal update operation not only changes feature model as less as possible, but also keeps the original design intent as much as possible. Therefore, the updated feature model shown in Figure 1(d) is considered optimal, as it only modifies the feature parameters without adding new features.

Considering that feature is the basic element of feature model, to effectively update the feature model after direct modeling operations are conducted, the features which are influenced by the direct modeling operations need to be identified first, based on which the update operations of the feature model including feature modification, feature addition and feature reordering can be generated. As update operations of the feature model are usually not unique, all candidate update operations need to be found out, from which the optimal update operations are chosen and used to obtain the updated feature model.

Based on the analysis above, an algorithm for automatic updating of the feature model after direct modeling operations is developed as given in Algorithm 1, which consists of three critical steps. First, variations of feature volumes are determined. In this step, the features influenced by the direct modeling operations are identified and the variation of the B-rep model is converted to the volume variations of those features. In the second step, candidate update operations are generated. According to the volume variations of the features, the ideal update way is converting the feature volume variations into feature modifications. Feature addition and feature reordering are applied in this step if necessary. In the third step, the optimal update operations are determined from the candidate update operations by evaluating each candidate update operation based on a scoring system.

In this work, the direct modeling operation considered is limited to face push-pulling which is the most commonly used direct modeling operation.

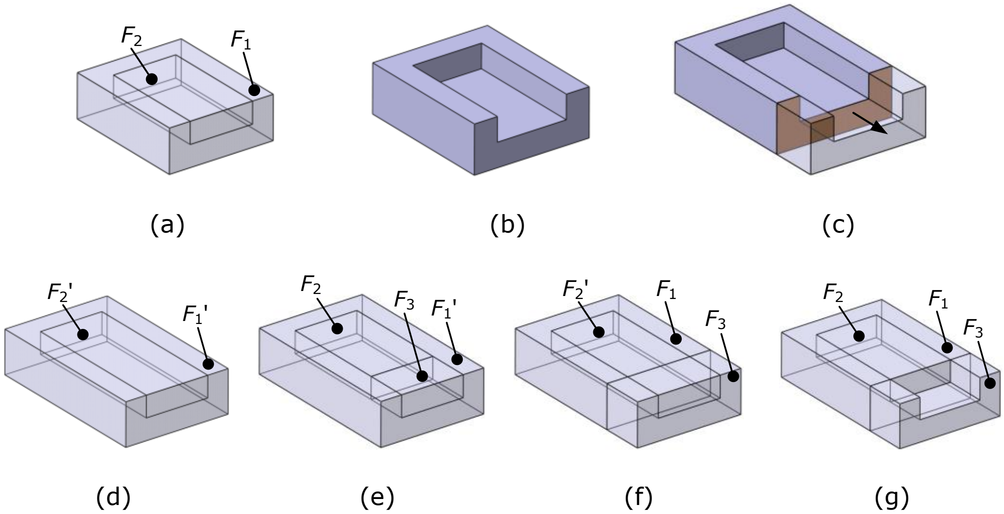


Figure 1: Illustrating of the problem: (a) Original feature model, (b) Original B-rep model, (c) Direct modeling operation, and (d-g) Four candidate updated feature models among which (d) is the optimal one.

Algorithm 1: Automatic update of feature model after direct modeling operation.

Input: O, M_{b}, M_{f} – Direct modeling operations, original B-rep model, original feature model

Output: M_{f}' – Updated feature model

1. $V \leftarrow$ DeterminationOfFeatureVolumeVariation (O , M_{b} , M_{f})
2. $C \leftarrow$ GenerateCandicateOperations (V)
3. $A \leftarrow \varphi$ //Array of operation scores
4. **for** each candidate operation $c \in C$ **do**
5. $s \leftarrow$ Evaluate (c)
6. add s to A
7. **end for**
8. $o_o \leftarrow$ GetOptimalOperation (A , C)
9. $M_{\mathrm{f}}' \leftarrow$ UpdateFeatureModel (o_o)
10. **Return** M_{f}'

3 ALGORITHM

3.1 Determination of Feature Volume Variations

In order to effectively update the feature model after direct modeling operations are conducted, we first identify the features influenced by the input direct modeling operations, and then determine the variation volumes of these features.

3.1.1 Identification of relevant features

In feature-based modeling, B-rep model is generated from feature model by Boolean operations of feature volumes. A boundary face in the resulting B-rep model is originally created as a face in a feature volume, and is then edited by faces of subsequent feature volumes in the following two ways: a) The face is merged or cut out by another face with the same underlying surface. b) The face is trimmed by another face with a different underlying surface. Therefore, if a boundary face in the B-rep model is push-pulled to a new position in a direct modeling operation, the corresponding new feature model should update the generating process of this face, i.e. update the features which are relevant to the process.

For the convenience of discussion, a feature that needs to be updated in the feature model after a direct modeling operation is called a relevant feature of the direct modeling operation. The involving features editing the face in a) and in b) are respectively called direct relevant feature and indirect relevant feature. As shown in Figure 2, given a feature model, the corresponding B-rep model and a direct modeling operation push-pulling a face in Figure 2 (a), we illustrate the generating process of the push-pulled face Figure 2 (b). Because feature F_1 creates the original face and feature F_2 merges the face in process, these two features are identified as direct relevant faces. Because feature F_3 and feature F_4 trims the face in process, these two features are identified as indirect relevant faces. The identification result is given in Figure 2 (c).

We define the rules of recognizing relevant features as follows.

Rule (Recognition of direct relevant feature) Given a set of push-pulled faces F and a feature S, if there exist a face $f' \in F$ and a face $f \in S$ which satisfies: 1) f and f' have the same underlying surface equation. 2) f and f' have non-empty intersection area. 3) Given $\mathbf{n}$ and $\mathbf{n}'$ which are respectively normal vectors of f and f' at intersection, $\mathbf{n}$ and $\mathbf{n}'$ are at the same direction if S is an additive feature, and are opposite otherwise. Then S is a direct relevant feature of the direct modeling operation, and f' is a direct feature face.

Rule (Recognition of indirect relevant feature) Given a set of direct relevant feature faces F, if there exists a feature S which satisfies: 1) Each $f \in S$ and $f' \in F$ has different underlying surfaces. 2) S and F have non-empty intersection area. Then S is an indirect relevant feature of the direct modeling operation.

Given the direct modeling operations, we obtain the corresponding push-pulled faces first. For each push-pulled face, we identify the direct relevant features and the indirect relevant features according to the rules above. With all the relevant features identified, the direct modeling operations can next be decomposed to variations of relevant feature volumes.

3.1.2 Generation of feature-variation volumes

Direct modeling operations bring a variation to the original B-rep model, which can be described as a 3D bounded region. Based on the relevant features identified, the 3D variation region can be converted to variations of these relevant feature volumes, called feature-variation volumes.

Feature-variation volume of direct relevant feature can be easily generated by method proposed in [11]. However, such direct modeling method cannot be applied directly on indirect relevant features, as the push-pulled face is actually nonexistent in the feature volume. On the basis of Zou's method, we give a method to generate feature-variation volume of indirect relevant

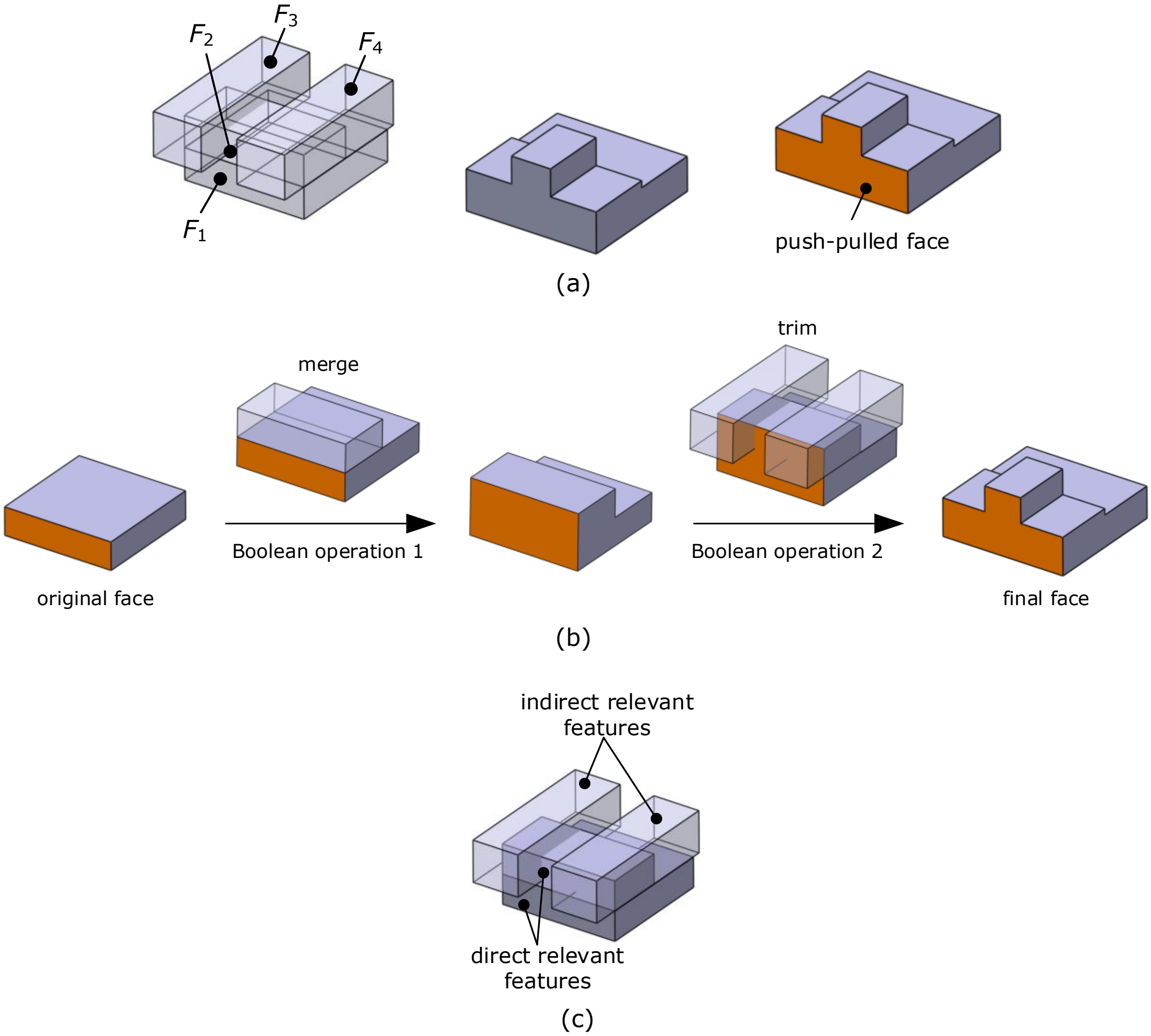


Figure 2: Determination of relevant features. (a) The instance feature model (left), the corresponding B-rep model (right) and the direct modeling operation (right). (b) Generation steps of the push-pulled face. (c) Corresponding direct and indirect relevant features.

feature. Given a direct modeling operation push-pulling a face f and the indirect relevant feature F, the feature-variation volume of F can be generated as follows:

1) Determine the virtual push-pulled face of F. Obtain the cross-section face of F at the position of f as the virtual push-pulled face.

2) Generate the auxiliary volume V_a. Trim the adjacent extending faces by the underlying surfaces of the virtual push-pulled face at its original position and its new position. The faces obtained can form a closed volume, which is called auxiliary volume.

3) $V = V_a - F$. If $V \neq \varnothing$, V is the feature-variation volume. Else, feature-variation volume is null.

We note that in some cases, part of the changing region cannot be converted to feature-variation volume, which is called independent-variation volume, shown in Figure 3. This is due to the differences among extending directions of adjacent faces of push-pulled faces. Such volumes will be handled later.

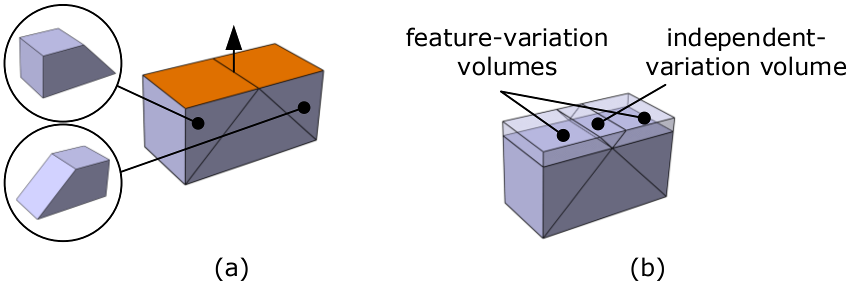


Figure 3: A case with independent-variation volume. (a) Direct modeling operation to a model with overlapped features, (b) Appearance of independent-variation volume.

3.2 Generation of Candidate Update Operations

Given a direct modeling operation, the corresponding available update operations are usually not unique. An approach to generating candidate update operations is given, consisting of three main steps. In the first step, we convert the feature-variation volumes to feature modifications. As some feature-variation volumes cannot be converted directly, we use a method called feature mending to form up a valid feature volume for feature modification, which also generates extra volumes. In the second step, the remaining volumes, including independent-variation volumes and the extra volumes generated in the first step, are to be added as new features. In the final step, if inconsistency still exists between the updated feature-based model and B-rep model, feature reordering is used to modify the modeling history.

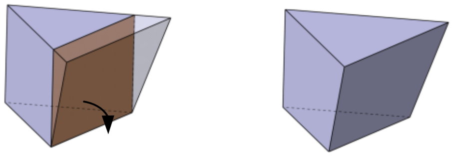
3.2.1 Feature modification

Apparently, the most reasonable update of a relevant feature is converting the feature-variation volume to its parameter modification. However, not all the feature-variation volumes can be converted directly as some may destroy the inherent feature semantics. In order to convert feature-variation volume to feature modification as much as possible, basic modification and feature mending are applied successively.

Basic modification is used in the situation that feature-variation volume can be directly converted to parameter modification of the feature. For each relevant feature, we enumerate all the possibilities of merging its feature-variation volumes into its original feature volume. When a merged volume is enumerated, a simple feature recognition is given to it based on the original feature. If the resulting feature type is the same as the original feature, update the parameter values of the feature.

We take extrusion feature as an example. If the merged volume has two non-adjacent faces which satisfy that all the edges that don't belongs to these two faces are straight edges with the same length, perpendicular to and connect these two faces, we can regard the merged volume as an extrusion feature volume. Then, the parameters of the original feature are mapped into the updated feature with the parameter values updated according to the merged volume.

However, feature-variation volumes cannot be directly converted in the following two situations: a) The merging operation destroys the original feature semantic, shown in Figure 4. b) The merging operation makes the merged volumes too complex to be recognized. To make conversion available in such situations, a method named feature mending is used. For a merged volume, the method adds some material to the original merged volume or delete some material from it to make it a recognizable feature volume again. After recognition and parameterization to the modified merged volume, the material added/deleted is considered extra volumes, which are left to be handled in the next steps.



(a) (b)

Figure 4: A merging destroying feature semantic. (a) Variation of an extrusion feature volume, (b) The merged volume is no longer an extrusion feature volume.

We also take extrusion feature as an example. The steps of extrusion feature mending are as follows:

1) For each vertex in the volume, calculate the signed value of the distance from the vertex to the feature sketch.

2) Get the vertex with the minimum distance, create a new sketch parallel to the original one.

3) Construct the new sketch by projecting all edges in the volume into the new sketch.

4) Create a new extrusion feature as the mended feature. The extruding direction is the same as the original feature. The extruding distance can be the maximal distance difference between vertices, or can be the minimum non-zero distance difference between the vertex at the minimum and another vertex. Multiple mended features may be generated in this step.

5) Calculate the difference between the new feature volume and the original one. The result is the extra volumes.

For a single relevant feature, the update operation is usually not unique. As is shown in Table 1, for a feature volume variation, three candidate update operations are listed. The first candidate operation only uses parameter modification, but the end faces after modification are no longer parallel. The other two operations use feature mending, which mostly keep the original feature semantics but generate extra volumes.

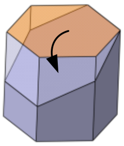
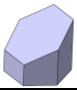
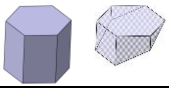
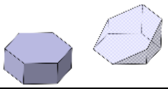
Model variation	*Operations*	*Description*
		Modify extruding plane.
		An extrusion feature volume with an extra reductive volume.
		An extrusion feature volume with an extra additive volume.

Table 1: Candidate update operations of an extrusion feature volume variation.

3.2.2 *Feature addition*

To ensure the consistency between the new B-rep model and the updated feature model, the remaining volumes, including the independent-variation volumes and the extra volumes generated from feature mending, are supposed to be instantiated as new features and be added to the feature model. From design perspective, adding features will have great impact on the feature model and the design intent, but such operations are inevitable in some situations where the variation volumes cannot be completely converted. What can be done is to minimize the influence on the feature model. To this end, we give two principles of feature addition as follows: a) The number of instantiated features should be as less as possible. b) The instantiated features are supposed to be simple. Sometimes these two principles cannot be satisfied both, so it may produce multiple results to be evaluated.

We give a method to convert the remaining volumes to new features. First, the remaining volumes are divided into a set of volume groups. Volumes in each group are adjacent. Then, for each group of volumes, we enumerate all of the merging possibilities. For each possibility, a feature recognition is given to the merged volume. If the volume is recognized successfully, instantiate it as a new parameterized feature. If it fails, we regard the volume as complex user-defined features. The new instantiated features are added to the end of the modeling history. It is worth noting that feature mending is not suitable to be used for the merged volumes, because such operations will once again bring extra volumes.

3.2.3 *Feature reordering*

In feature-based modeling, if a feature A is added before a feature B, then B will cover A in their overlapped region by Boolean operation. Therefore, two feature models with the same feature set and different feature orders will usually result in different B-rep models, because the difference in the feature order causes different sequences of Boolean operations. Inconsistency between the B-rep model and the feature model is generated after direct modeling operation as the update of features may change the covering region of the feature volumes which further changes the covering relationship between feature volumes. As is shown in Figure 5, the direct modeling operation changes the feature volume of F_2, generating a new overlapped region between F_2 and F_4. According to the new B-rep model, F_4 should be instantiated before F_2 to be covered in their overlapped region, but that conflicts with the original feature order, which further leads to the inconsistency. Therefore, the essence of the inconsistency is the unreasonable feature order, so feature reordering is necessary to be applied in this situation.

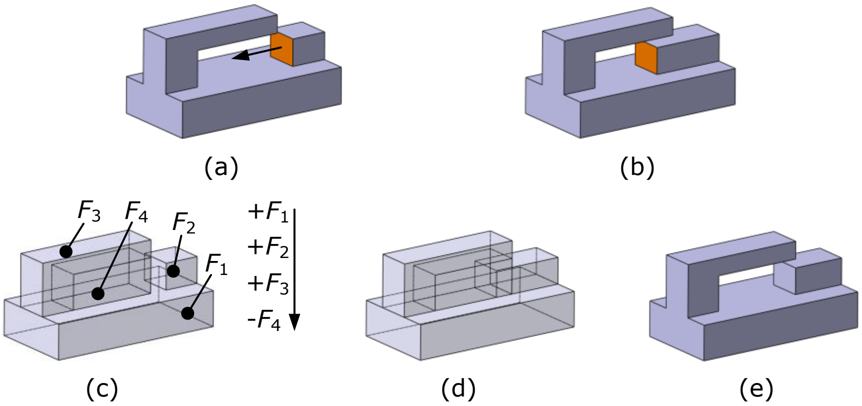


Figure 5: Inconsistency results from feature order. (a) Original B-rep model and a direct modeling operation, (b) Resulting B-rep model, (c) Original feature model, (d) Updated feature model without feature reordering, (e) Inconsistent B-rep model generated from (d).

Simply moving a single conflict feature may break the relative positions between features, leading to some new inconsistencies. Therefore, feature reordering should also take relative positions between features into consideration. To resolve this problem, we give the following definition describing the position relationship between features:

Definition (Preposition feature) Suppose feature A and feature B are two features in feature model M. If B should be instantiated before A in M's modeling history, then B is called a preposition feature of A. The set which contains all of the preposition features of A is called the preposition feature set of A.

Definition (Dislocation feature) Suppose feature A and feature B are two features in feature model M, and B is a preposition feature of A. If B locates after A in M's modeling history, then B is called a dislocation feature of A. The set which contains all of the dislocation features of A is called the dislocation feature set of A.

Apparently, a model without inconsistency satisfies that the dislocation feature sets of any features are empty. To this end, the aim of feature reordering is to reorder the feature order to empty all of the features in dislocation feature sets. When the preposition feature set of a feature is obtained, the dislocation feature set of this feature is determined from the current modeling history. Hence, the critical point is to determine the preposition feature set of the feature. We give several basic heuristic rules as follows:

1) Indirect relevant features should locate after direct relevant features.

2) For direct relevant features, reductive features should locate after additive features.

3) Additional features should locate after their corresponding father features.

4) If feature A's feature-variation volume overlaps with feature B's original volume, and the two volumes have the opposite attributes, then A should locate after B.

After determination of the preposition feature set of each feature, the dislocation feature set can also be determined. The next step is to reorder the feature sequence to make each dislocation feature set empty. An intuitive way is to move the feature towards the end of the sequence step by step until all its dislocation features locate before it. However, the movement is not always feasible as conflicts will occur in the following two situations:

1) If the moving feature is the preposition feature of the next feature, the movement will bring another dislocation.

2) If two features' original feature volumes have the opposite attributes and have overlapping regions, the movement will bring inconsistency.

To resolve the conflicts above, the conflict feature is also moved together to maintain the relative position between the two features.

Based on the analysis above, the algorithm of the feature reordering is given in Algorithm 2. For each feature which has non-empty dislocation set, try to swap the feature with the next one until the dislocation set is empty. If conflict occurs in a swapping step, move the next feature until the conflict no longer exists. When all the dislocation sets are empty, the consistency is achieved again between the feature model and the B-rep model. An example is given in Figure 6 to illustrate the algorithm resolving the inconsistency in the model in Figure 5. In the feature model, the inconsistency occurs between feature F_2 and feature F_4, and F_4 is the only dislocation feature of F_2. To eliminate the inconsistency, F_2 needs to move until F_4 locates before F_2. Firstly, swap F_2 with the next feature F_3 without conflict. As the dislocation set of F_2 is still non-empty, we then swap F_2 with the next feature F_4. After that, the dislocation set of F_2 is empty, and the consistency is achieved.

It is worth noting that in some typical situations, inconsistencies cannot be totally eliminated using this method, as there exists coupling between features. An example is given in Figure 7, where F_1 and F_2 are both the preposition features of F_3. After the direct modeling operation in Figure 7 (c-d), one update operation is converting the variation volume into parameter modification of F_2.

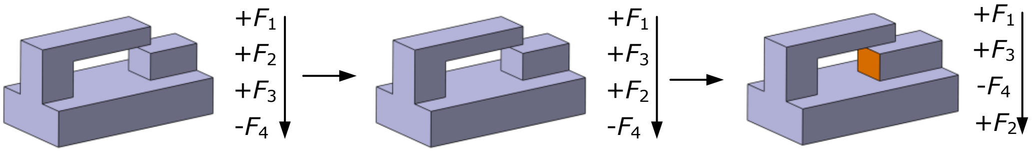


Figure 6: Feature reordering resolving the inconsistency of the model in Figure 5.

After that, F_2 covers F_3 in another overlapped region, and F_3 becomes a preposition feature of F_2, which brings a coupling that F_2 and F_3 are proposition feature to each other. Such couplings result from unreasonable conversion of variation volumes. If the inconsistency cannot be eliminated, the candidate update operation should be abandoned.

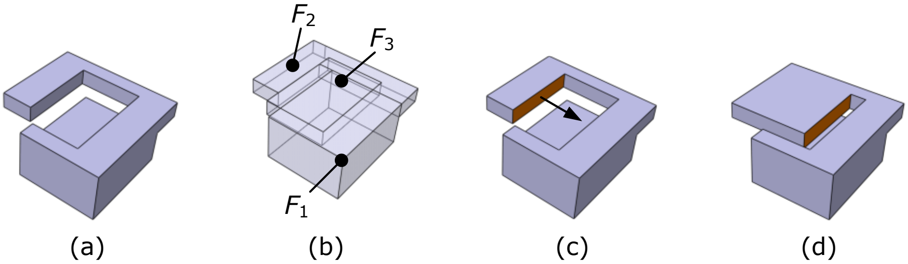


Figure 7: Inconsistency results from feature coupling. (a) Original B-rep model, (b) Original feature model, (c) A direct modeling operation, (d) The new B-rep model after direct modeling operation.

3.3 Determination of Optimal Update Operation

The optimal update operation is to be determined from a set of candidate operations. Basically, an ideal update operation is supposed to maintain feature semantics and user's design intent as much as possible. In this paper, we design a scoring system for determining the optimal update operation, in which each candidate operation is independently evaluated and gets a penalty score for comparison. The higher score the update operation gets, the worse its quality is. According to all of the scores, the candidate operation with the lowest score turns out to be the optimal one.

The final score of a candidate update operation is determined by evaluations of three sub-operations discussed in this paper: modifications of feature parameters, variations of feature order and additions of new features. Given the original feature model M_{fo} and a candidate update operation O, the penalty score $\mathrm{S}(O, M_{\mathrm{fo}})$ can be represented as follows:

$$\mathrm{S}(O, M_{\mathrm{fo}}) = \omega_{\mathrm{p}} \mathrm{S}_{\mathrm{p}}(O, M_{\mathrm{fo}}) + \omega_{\mathrm{a}} \mathrm{S}_{\mathrm{a}}(O) + \omega_{\mathrm{r}} \mathrm{S}_{\mathrm{r}}(O, M_{\mathrm{fo}})$$

In this formula, S_{p}, S_{a} and S_{r} are the penalty functions of the sub-operations, among which S_{p} punishes the feature parameter variations, S_{a} punishes the feature additions and S_{r} punishes the feature order variations. ω_{p}, ω_{a} and ω_{r} are respectively the weights of S_{p}, S_{a} and S_{r}, which can be modified by users according to their preferences. The scoring criteria of each sub-operation is described in detail as follows.

Algorithm 2: Feature reordering.

Input: O_{i} – Original feature order

Output: O_{o} – Reordered feature order

$O_{\mathrm{o}} \leftarrow O_{\mathrm{i}}.\mathrm{duplicate}()$
for each $\mathit{feature} \in O_{\mathrm{i}}$ **do**
$pos_{\mathrm{cur}} \leftarrow \mathit{feature}.\mathrm{Position}()$
$pos_{\mathrm{next}} \leftarrow pos_{\mathrm{cur}} + 1$
while $pos_{\mathrm{next}} < O_{\mathrm{i}}.\mathrm{Count}()$ **and** $\mathit{feature}.\mathrm{DislocationSet}().\mathrm{Count}() \neq 0$
$\mathit{flag} \leftarrow \mathit{true}$
for $i \leftarrow pos_{\mathrm{cur}}$ **to** $pos_{\mathrm{next}} - 1$ **do**
$\mathit{feature}_{\mathrm{tmp}} \leftarrow O_{\mathrm{i}}.\mathrm{GetFeature}(i)$
if $\mathit{Conflict}(\mathit{feature}, \mathit{feature}_{\mathrm{tmp}})$ **or** $\mathit{feature}.\mathrm{PrepositionSet}().\mathrm{Contains}(\mathit{feature}_{\mathrm{tmp}})$ **then**
$\mathit{flag} \leftarrow \mathit{false}$
break
end if
if flag **then**
for $i \leftarrow pos_{\mathrm{next}} - 1$ **down to** pos_{cur} **do**
if $\mathit{feature}_{\mathrm{tmp}}.\mathrm{DislocationSet}().\mathrm{Contains}(\mathit{feature})$ **then**
$\mathit{feature}_{\mathrm{tmp}}.\mathrm{DislocationSet}().\mathrm{Remove}(\mathit{feature})$
$O_{\mathrm{o}}.\mathrm{SwapPosition}(\mathit{feature}, \mathit{feature}_{\mathrm{tmp}})$
end if
$pos_{\mathrm{cur}} \leftarrow pos_{\mathrm{cur}} + 1$
end if
$pos_{\mathrm{next}} \leftarrow pos_{\mathrm{next}} + 1$
end while
end for
Return O_{o}

3.3.1 Scoring criteria of feature parameter variations

We give the penalty function $\mathrm{S_p}(O, M_{\mathrm{fo}})$ of feature parameter variations as follows:

$$\mathrm{S_p}(O, M_{\mathrm{fo}}) = \sum_{f \in M_{\mathrm{fo}}} w_{\mathrm{pf}}\, \mathrm{S_{pf}}(f, \mathrm{Update}(f, O))$$

In this formula, f is a feature in the original feature model M_{fo}. $\mathrm{Update}(f,O)$ outputs the corresponding updated feature of f. S_{pf} punishes the parameter variation of a single feature, of which w_{pf} is the weight.

According to parameter types, variations of feature parameters is further divided into sketch variations and other parameter variations. The penalty function is represented as:

$$\mathrm{S}_{\mathrm{pf}}(f,f') = \omega_{\mathrm{s}}\mathrm{S}_{\mathrm{s}}(f,f') + \omega_{\mathrm{o}}\mathrm{S}_{\mathrm{o}}(f,f')$$

In this formula, f is the original feature and f' is the corresponding updated one. S_{s} is the penalty function punishing the sketch variation and S_{o} punishes the variation of other feature parameters. ω_{s} and ω_{o} are weight of respectively S_{s} and S_{o}.

Sketch is a 2D shape formed by a set of line or curve segments connected in sequence. Variation of sketch is generally discussed from two aspects: geometry and topology. For geometry, geometric descriptors are used to describe to describe a 2D geometry, but geometric descriptors are more suitable for mesh models than B-rep model. In addition, using of geometric descriptors will complicate the evaluation process of sketch and reduce its efficiency. To this end, we use a simpler and more intuitive approach to describing geometric variation of sketch which compares the perimeter and the area of sketch before and after the update. The geometric penalty function S_{g} is first given:

$$\mathrm{S}_{\mathrm{g}}(s,s') = w_{\mathrm{p}}\triangle_{\mathrm{p}}(s,s') + w_{\mathrm{s}}\triangle_{\mathrm{s}}(s,s')$$

In this formula, s and s' are the sketches of the original feature and the updated feature. $\triangle_{\mathrm{p}}$ is the difference in perimeter and $\triangle_{\mathrm{s}}$ is the difference in area between s and s'. w_{p} and w_{s} are the weights of respectively $\triangle_{\mathrm{p}}$ and $\triangle_{\mathrm{s}}$.

For topology, variations of edges and loops in sketch are taken into consideration. The penalty function S_{t} is defined as:

$$\mathrm{S}_{\mathrm{t}}(s,s') = w_{\mathrm{e}}\triangle_{\mathrm{e}}(s,s') + w_{\mathrm{l}}\triangle_{\mathrm{l}}(s,s')$$

In this formula, $\triangle_{e}$ is the difference in edges and $\triangle_{l}$ is the difference in loops. w_{e} and w_{l} are the weights. For edges, variation of their type and amount is considered. We classify the edge into three types: line, quadratic curve and free curve. For loops, variation of their amount is considered.

Representation of penalty function S_{o} depends on the feature types. Different features have different parameters, which cannot be evaluated by a uniform standard. Take extrusion feature as an example, except for sketch, its main parameters are: extruding direction and distance. For extruding direction, the angle between the extruding vectors before and after is used as the measure of variation. For extruding distance, relative difference between the distance values before and after is used.

3.3.2 Scoring criteria of feature additions

Compared to modifying feature parameter, adding new features has a greater impact on design intent, so the penalty score should also be higher. One way to achieve that is increasing ω_{a}, but it is not enough. For example, suppose there are two candidate update operations corresponding to a direct modeling operation, one is simply modifying feature parameters, the other is simply adding a new feature. Generally, the former operation is more reasonable and should get a lower score.

However, if the value of S_{a} is pretty low, the value of $\omega_{\mathrm{a}}\mathrm{S}_{\mathrm{a}}$ may be even lower than $\omega_{\mathrm{p}}\mathrm{S}_{\mathrm{p}}$, leading to an unreasonable result which will select the latter operation as the optimal one. To this end, we additionally give a constant value to S_{a} to guarantee that the penalty of adding features is always higher than that of modifying parameters no matter how "good" the new features are. The penalty function S_{a} is represented as:

$$\mathrm{S}_{\mathrm{a}}(O)=\sum_{f\in O} w_{\mathrm{af}}(\mathrm{S}_{\mathrm{af}}(f)+c_{a})$$

In this formula, f is an adding feature of O. S_{af} is the penalty function of a single adding feature. c_{a} is the constant value. w_{af} is the weight.

To make the feature model less complex, the adding features are supposed to be as simple as possible. Hence, we evaluate a single adding feature depending on its complexity. The simpler the feature is, the lower penalty score it gets. The final penalty score of the whole feature addition operation is the accumulation of all the penalty scores of adding features.

The complexity of the adding feature is evaluated from geometry and topology. For geometry, we use the ratio of concave edges in the feature volume as the criterion. The higher ratio of concave edges is, the more complex the feature volume will be. For topology, we use the number of genera in the feature volume as the criterion. We think that adding a genus from nothing will has a great impact on the feature complexity, but when the number of genera is large, adding one will have less impact. Hence, we use the arctan function to evaluate the topological complexity of the feature volume. On the one hand, it is because the arctan function is a monotone increasing function, and the function value has upper bound but the range of independent variable has no bound. On the other hand, as its derivative function is decreasing, the increasing rate of function value will decrease as the number of genera increases. Based on the analysis above, the penalty function S_{af} is represented as follows:

$$\mathrm{S}_{\mathrm{af}}(f)=\frac{\arctan(\mathrm{G}(f))}{\pi/2}+\frac{\mathrm{E}_{\mathrm{c}}(f)}{\mathrm{E}(f)}$$

In this formula, G counts the number of genera in feature f. E_{c} counts the number of concave edges. E is the number of edges.

3.3.3 Scoring criteria of feature order variations

For each feature, we can get the change between their original position to their after-reordered position, i.e. the absolute value of the difference between former and latter. In addition, we should consider the size of feature tree, because for the same distance, the bigger the feature tree is, the less influence it has, so we use the ratio of position variation to the size of feature tree to measure the change of a feature's position. Finally, we use the weighted mean of all features' position change as the feature model's position change, so the penalty function S_{r} is defined as:

$$\mathrm{S}_{\mathrm{r}}(O,M_{\mathrm{fo}})=\frac{\sum_{f\in M_{\mathrm{fo}}} w_{\mathrm{r}}\dfrac{\left|\mathrm{pos}(f)-pos(\mathrm{new}(f,O))\right|}{\mathrm{Count}(M_{\mathrm{fo}})}}{\mathrm{Count'}(O,M_{\mathrm{fo}})}$$

f is a feature in the original feature model. Function $\mathrm{pos}()$ returns the position of a feature. Function $\mathrm{new}()$ returns the position of an updated feature. Function $\mathrm{Count}()$ returns the number of features in the original feature model. Function $\mathrm{Count'}()$ returns the number of features in the updated feature model.

4 RESULTS

A program has been implemented based on ACIS. After direct editing of the b-rep model, the program starts to run and the feature model is automatically updated as output by the optimal update operation. To ensure the simplicity of the output result, the scoring system is running inside the program and only the optimal operation is provided to the user.
Three representative cases are tested to verify the validity of the method proposed in this paper, shown in Figure 8. It is noted that feature models in the following cases are mainly presented by primitive features. Theoretically, the method is also valid for models with functional features (i.e. slots, holes), because the feature volumes of these functional features are also able to be constructed and recognized, as long as the corresponding recognition methods exist.

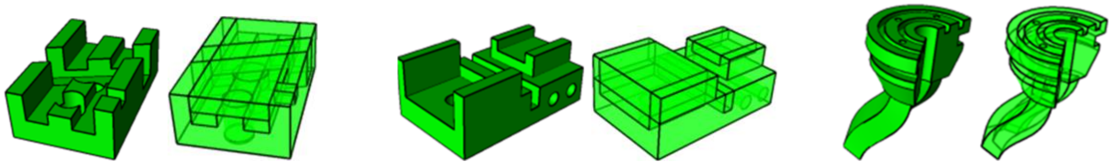


Figure 8: Three cases to be tested.

The model in the first case is composed by 8 features, including an additive extrusion feature and 7 reductive extrusion features. Two direct modeling operations are separately given, shown in Figure 9. The first direct modeling operation rotates the red face and obtains a new B-rep model. The update operation output modifies parameters of 4 features. The second direct modeling operation is given pulling two cylindrical faces corresponding to a hole feature. The output update operation only modifies the parameter of the hole feature.

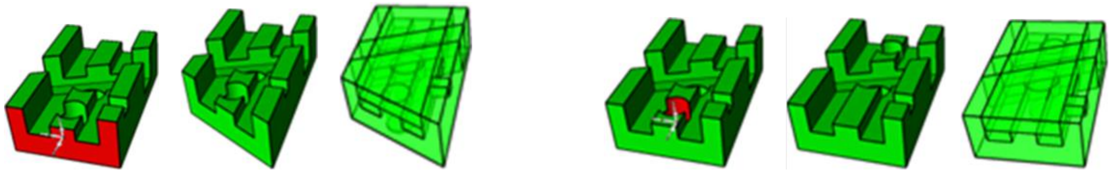
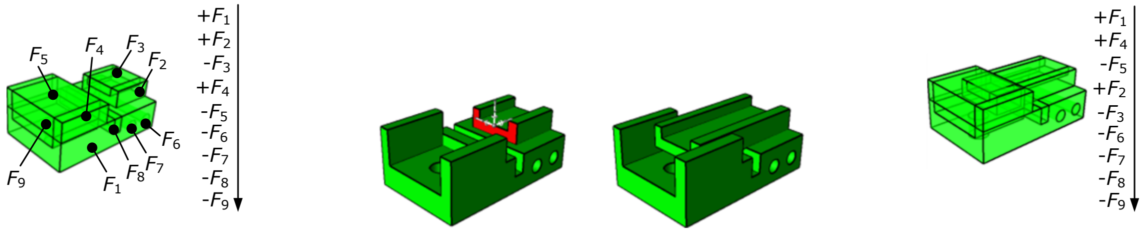


Figure 9: The first case. (a) First direct modeling operation and corresponding updated feature model, (b) Second direct modeling operation and corresponding updated feature model.

The second case is used for testing the ability of feature reordering, shown in Figure 10. The original model consists of 9 features. A direct modeling operation given pulls one face to a new position. If feature reordering is not applied, inconsistency will occur between the updated feature model and the B-rep model. The optimal update operation output modifies the parameter of feature F_2 and F_3, and change the feature order, which maintains the consistency between the new B-rep model and the new feature model.

The third case in Figure 11 gives a complex model consisting one additive revolve feature, one additive sweep feature and three reductive extrusion features. Two direct modeling operations are separately given. The first operation pulls the face on the revolve feature. The output update operation modifies the parameters of revolve feature and three extrusion features. The second direct modeling operation pulls the face on the sweep feature. The update operation output adds a user-defined feature because the method cannot simply modify the parameters of sweep feature to achieve the update.



(a) (b) (c)

Figure 10: The second case. (a) Original feature order, (b) Direct modeling operation and the new B-rep model, (c) The output updated feature model and the new feature order.

(a) (b)

Figure 11: The third case. (a) First direct modeling operation and corresponding updated feature model, (b) Second direct modeling operation and corresponding updated feature model.

5 CONCLUSION

A new method is presented in this paper to update feature model after direct editing, which realize the conversion from direct modeling to feature modeling. The method guarantees that the updated feature model is consistent with the B-rep model edited. Compared with Fu's method, the method proposed in this paper doesn't have to depend on cellular model. In addition, with the scoring system, users are allowed to set their preference to make the updated feature model more satisfactory.

The method can handle a number of situations but not all, such as complex cases with feature coupling. Our future work will focus on the following aspects: 1) Improve the method to make it be able to handle more complex situations; 2) Extend the method to support other direct modeling operations besides push-pulling of faces; 3) Improve the efficiency of the method to make it be able to efficiently handle the situation where large number of the features are influenced by direct modeling operations.

6 ACKNOWLEDGEMENTS

The authors are very grateful to financial support from TC190A4DA/3.

7 ORCID

Xiaolian Qin, https://orcid.org/0000-0001-7074-8936
Zhihong Tang, https://orcid.org/0000-0002-5262-5059
Shuming Gao, http://orcid.org/0000-0002-7061-9912

REFERENCES

[1] Ault, H.; Phillips, A.: Direct modeling: easy changes in CAD? Proceedings of the 70th ASEE engineering design graphics division midyear conference, 2016, 99–106. https://commons.erau. edu/asee-edgd/conference70/papers-2016/14

[2] Bourdot, P.; Convard, T.; Picon, F.; Ammi, M.; Touraine, D.; Vézien, J.-M.: VR–CAD integration: Multimodal immersive interaction and advanced haptic paradigms for implicit edition of CAD models, Computer-Aided Design, 42(5), 2010, 445-461. https://doi.org/10.1016/j.cad.2008.10.014

[3] Direct Modeling with Creo, https://www.ptc.com/en/products/cad/creo

[4] Direct Modeling with Dassault Systems. http://perspectives.3ds.com/interviews/direct-modeling-with-dassault-systemes

[5] Fu, J.; Chen, X.; Gao, S.-M.: Automatic synchronization of a feature model with direct editing based on cellular model, Computer-Aided Design and Application, 14(5), 2017, 680-692. https://doi.org/10.1080/16864360.2016.1273585

[6] Gaughran, W. F. (2002). Cognitive modeling for engineers, Proceedings of the 2002 American Society for Engineering Education Annual Conference and Exposition, 2002. https://peer.asee.org/10547

[7] Shah, J. J.; Mantyla, M.: Parametric and Feature-based CAD/CAM: Concepts, Techniques, and Applications. John Wiley & Sons, 1995.

[8] Synchronous technology with NX, http://www.plm.automation.siemens.com

[9] Tornincasa, S.; Monaco, F. D.: The Future and the Evolution of CAD, Proceedings of the 14th international research/expert conference, 2010, 11-8. https://doi.org/10.1299/jsdd.4.575

[10] Zou, Q.; Feng, H.-Y.: Variational B-rep model analysis for direct modeling using geometric perturbation, Journal of Computational Design and Engineering, 6(4), 2019, 606-616. https://doi.org/10.1016/j.jcde.2019.03.002

[11] Zou, Q.; Feng, H.-Y.: push-pull direct modeling of solid CAD, Advances in Engineering Software, 127, 2019, 59-69. https://doi.org/10.1016/j.advengsoft.2018.10.003