Proceedings of DETC2004
2004 ASME Design Engineering Technical Conferences
Salt Lake City, Utah, USA, 28 September-2 October, 2004

DETC2004-57695

CONSTRUCTION OF FREEFORM FEATURE MODELS WITH ATTACHMENTS

Eelco van den Berg

Faculty of Electrical Engineering,
Mathematics and Computer Science
Delft University of Technology
Mekelweg 4, NL-2628 CD Delft
The Netherlands
E.vdBerg@EWI.TUDelft.NL

Rafael Bidarra

Faculty of Electrical Engineering,
Mathematics and Computer Science
Delft University of Technology
Mekelweg 4, NL-2628 CD Delft
The Netherlands
R.Bidarra@EWI.TUDelft.NL

Willem F. Bronsvoort

Faculty of Electrical Engineering,
Mathematics and Computer Science
Delft University of Technology
Mekelweg 4, NL-2628 CD Delft
The Netherlands
W.F.Bronsvoort@EWI.TUDelft.NL

ABSTRACT

In freeform feature modeling, the shape domain of current feature modeling systems is extended with freeform shapes. A challenging task in this context is to develop a method for constructing a feature model in an intuitive, yet unambiguous manner, comparable to the construction methods for regular-shaped feature models. This paper describes a framework in which it is possible to attach a face of a new freeform, volumetric feature instance to a face of a feature instance already in the model. In this framework, features are represented by configurations of freeform definition points, from which the shape is generated. By using geometric constraints on these definition points and other geometric entities within the model, freeform attachments can be realized. Apparent problems that emerge in this context are positioning of the new feature instance on the freeform attach face, fitting the geometry of the new feature instance to the existing model geometry, and maintenance of valid feature model geometry. The framework accommodates for these issues. Two types of attachment, for a freeform extrusion feature respectively a freeform wrap feature, are elaborated. With the attachments presented here, freeform feature models can be constructed in a fully parametrized, constraint-based way, just like regular-shaped feature models.

Keywords: Feature modeling, Freeform features, Feature attachment, Constraint solving, Cellular model

1 Introduction

An important new development in product modeling is freeform feature modeling, which is concerned with either surface features or volumetric features. Freeform surface features can be used for modeling thin artefacts like panels and car bodies, whereas freeform volumetric features can be used for modeling volumetric parts with freeform faces. This paper is about the latter. Freeform volumetric features correspond, just like regular-shaped features, to generic, parametrized volumes. The difference is that there is more modeling freedom, i.e. the faces of the features can be modeled with, for example, NURBS. A detailed survey of freeform feature modeling is given in [7].

The basic idea of freeform volumetric feature modeling is to extend the shape domain of common feature modeling systems from regular volumetric shapes, such as prismatic and cylindrical shapes, to freeform volumetric shapes, such as sweeps with freeform profile curves. Freeform volumetric features are, just as regular volumetric features, parametrized and constraint-based. The advantage of this approach is that it provides accurate shape control through high-level parameters and constraints that capture the semantics of features. This means, for example, that a designer is able to control specific aspects of a feature shape, while other aspects remain fixed. Using high-level parameters instead of directly manipulating low-level NURBS entities also relieves a designer from the requirement to have extensive knowledge of the underlying geometric representation.

In regular-shaped feature modeling, a new type of feature

can be defined in a feature class. An instance of a feature class can be added to a feature model by specifying the parameters for the feature, and the way it should be attached to the model. This can considerably simplify specification of models, compared to geometric modeling. In addition, the properties, or semantics, of a feature can be defined in a feature class by constraints, and these constraints can be maintained during the whole modeling process [1]. A simple example of such a property is that the height of a feature is twice its width. Validity maintenance can guarantee that only valid feature models are created, i.e. models that contain only features that satisfy all properties specified for them. Although most current feature modeling systems have a rudimentary form of validity maintenance only, the ability to perform more advanced validity maintenance will be an important requirement for future feature modeling systems.

Of particular interest in this context are situations where a feature interacts with others. By using so-called interaction constraints, restrictions that are maintained throughout the modeling process can be set on such feature interactions. An example of such a restriction is that the top of a pocket should remain open, i.e. that it may not be covered by any other feature. It has been shown that such constraints can indeed be maintained, if the feature model is represented by a cellular model [2].

The concept of attachment of a feature instance to a model is essential to feature modeling, and has several attractive properties; for example, the depth parameter of a through feature attached to two faces in the model is automatically adjusted when one of the faces is repositioned.

However, whereas attachment of regular-shaped features is relatively simple, this is certainly not the case for attachment of freeform features. Several issues related to the specification and the semantics of such attachments have to be resolved. In this paper, a framework for attaching freeform features is presented in which this has been realized. The feature model used within the framework is again a cellular model, and thus advanced validity maintenance of freeform features has also become feasible.

The framework uses the approach to specify freeform features, both classes and instances, presented in [8]. In this approach, a freeform feature class is specified by using a prototype for the generic shape definition and the constraint-based parametrization. The prototype is exploited by a prototype-driven constraint solving method to efficiently and unambiguously determine the shape of a freeform feature instance.

Attaching a freeform feature instance to a feature model consists of several steps: positioning the instance, fitting it to the existing feature model in an appropriate way, and adding the resulting shape to the feature model. For different types of feature classes, the individual steps are different. The steps are in this paper elaborated for freeform extrusion features and freeform wrap features.

Section 2 reviews the concepts of freeform feature class specification and instantiation, and the prototype-driven constraint solving approach, presented in detail in [8]. Section 3 reviews literature on attachments, and gives some requirements for freeform feature attachments. Section 4 discusses the basic ideas of the framework for attaching freeform features. Section 5 gives details for freeform extrusion features, and Section 6 for freeform wrap features. Finally, Section 7 gives some results and conclusions.

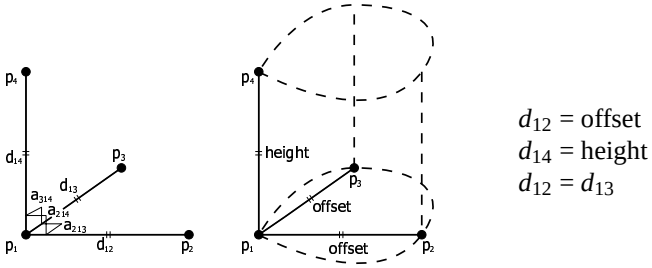


Figure 1. Constraining and parametrizing a configuration of FFDPs.

2 Freeform feature class specification and instantiation

The framework for attaching a freeform feature instance to a feature model presented in this paper, uses the approach for specification and instantiation of freeform feature classes presented in [8]. This approach is therefore briefly reviewed here.

Freeform feature class specification starts by specifying some basic properties of the feature class, e.g. whether it is an additive or a subtractive feature.

The generic shape of the feature class is specified by a *shape prototype*, which is modeled using a set of Freeform Feature Definition Points (FFDPs). These are points in 3D space that are used to define modeling elements such as curves. From these elements, the geometry of the shape prototype, and eventually the shape of feature instances, is created using construction techniques such as sweeping and skinning. A set of FFDPs is referred to as a *configuration*. Within a feature class specification, a set of FFDPs is actually a *configuration prototype*. In Figure 1, an example is given of a configuration consisting of 4 FFDPs.

Subsequently, to be able to generically instantiate a feature class, a system of geometric constraints is defined on the FFDPs. Two types of constraints are available for positioning FFDPs.

A distance constraint between FFDPs $p_a, p_b \in \mathbb{R}^3$ is defined as:

$$\| p_a - p_b \| = d_{ab} \qquad d_{ab} \in \mathbb{R}^+$$

where d_{ab} can be a variable or a fixed value.

An angle constraint between FFDPs $p_a, p_b, p_c \in \mathbb{R}^3$ is defined as:

$$\frac{p_a - p_b}{\|p_a - p_b\|} \cdot \frac{p_c - p_b}{\|p_c - p_b\|} = \cos(\alpha_{abc}) \qquad \alpha_{abc} \in [0, \pi]$$

where α_{abc} can be a variable or a fixed value.

These two geometric constraint types are sufficient to fix an FFDP anywhere in 3D space, relative to three already fixed FFDPs in a configuration. For every subset of three FFDPs within a set of four FFDPs, either the three distances between the FFDPs must be known, or two distances and one angle, or one distance and two angles. In Figure 1, three distance and three angle constraints define a set of four FFDPs.

After the system of geometric constraints has been specified, the parameters for the feature class are defined. With a combination of distance and angle constraints on FFDPs and of algebraic constraints, it is possible to define parameters related to characteristics of the feature shape that are intuitive for a designer who instantiates the feature class. In Figure 1, three equality constraints are defined, resulting in two parameters.

Finally, to allow validity maintenance of a feature model, validity conditions for the feature class can be specified with a variety of constraints [1].

After the freeform feature class specification has been completed, the class is validated. It is attempted to generate an instance of the freeform feature class using the default values for the feature parameters that have been included in the specification. Instantiation can fail for two reasons; the set of constraints specified for the class can be misconfigured, or the resulting geometry can be invalid. If instantiation fails, and the problem is not structural, the default parameter values can be replaced by new values. If at least one instance can be successfully generated, the freeform feature class is considered to be well-specified. Otherwise, a correction to the feature class specification must be made.

Freeform feature instances are, of course, also created during modeling sessions, to add them to a feature model. In this case, the parameter values are determined by the user, some explicitly by specifying numerical values, others implicitly by the way the instance is attached to the feature model.

For both feature class validation and instance creation, the shape of the feature instance is determined by solving the set of constraints defined in the feature class specification. A prototype-driven constraint solving method is used for this. The set of constraints is represented by a constraint graph. The solving method is based on the generic solver presented in [5], and consists of three phases: graph analysis, subproblem solving, and solution construction. For a system of geometric constraints on points such as defined above, there is generally more than one solution [4]. Feature class specifications should, however, be unambiguous. Our prototype-driven constraint solving method automatically selects a specific solution. It guarantees that the selected solution is the solution intended when the freeform feature class was specified, by using the configuration prototype of the freeform feature class specification corresponding to the latest default parameter values as a representation of the intention of the class.

See [8] for more details on the feature class specification and the prototype-driven constraint solving method.

3 Feature attachments

The possibility to attach a new feature instance to an existing feature model is an important facility in regular-shaped feature modeling systems. A face of the new instance can be "coupled" to a face in the existing feature model. If both faces are planar, this means that the two faces become co-planar and that the face of the instance is positioned on the face of the feature model, possibly with some variational parameters. Both the co-planarity and the positioning are imposed with geometric constraints, which implies that the attachment remains intact after a subsequent modeling operation. For some types of features, e.g. a blind hole, one face of the feature is attached to the model; for other types of features, e.g. a through hole, two faces of the feature are attached to the model. In the latter case, the depth parameter of the through hole feature is automatically recomputed after a model adjustment. Attachments considerably simplify model specification, and fit very well in the parametrized, constraint-based way of geometric modeling that underlies feature modeling.

Attachments have been used in regular-shaped feature modeling systems for quite some time already, and usually to the satisfaction of users of such systems. There are, however, several issues with regard to the specification and implementation, and in fact the semantics, of feature attachments that are not straightforward. One of these issues is the final extent of a feature if at least one of the two faces involved in an attachment is non-planar. Chen and Hoffmann [3] were the first to observe such issues, and to suggest solutions for several of them. In particular, they give rules for determining the extent of features generated from extrusions and revolutions for several interesting attachment cases. They first create a protofeature, which is a shape of sufficient extent to include the final feature extent, and subsequently compute which parts of the protofeature actually belong to the final feature extent. The latter depends on the type of attachment (for example, one- or two-face attachment), and on the type of feature operation (protrusion, cut or restriction, corresponding to set union, difference and intersection). Although their work is not directed towards freeform features, some of their basic ideas could be exploited in this work.

Almost all work on freeform feature modeling until now has been on surface features [7]. For freeform surface features, the

concept of attachment is not known. Such features are usually positioned either in the absolute coordinate system of the 3D modeling space, or in some other coordinate system defined relative to the absolute coordinate system. The positioning of a feature, in addition to the shape of the feature, can be parametrized. The parameters eventually influence the positions of the control points of the surface features. Additional intra-feature and inter-feature constraints on the control points, complemented with blends, can guarantee the desired continuity of the features. The system presented by Vosniakos [9] is a typical example of a system that offers such functionality. Recently, also more advanced ways to manipulate freeform surface features by way of deformations have been introduced [6].

In this paper, a method for, and implementation of, attachment of freeform volumetric features is presented. The basic idea is to be able to attach such freeform features in a similar way as regular-shaped features, i.e. to keep the attractive properties of attachments of regular-shaped features. However, specification and implementation of attachment of freeform features is much more difficult than attachment of regular-shaped features, particularly if in the latter case only planar attach faces are involved. The main issues that have to be resolved for the attachment of a freeform feature instance to a model are shortly mentioned here.

First, the user should be able to specify the position and orientation of the instance in the model in an intuitive way. This can be done by selecting an attach face in the model, determining one or more parametrized positions on the attach face, and fixing the orientation of the instance relative to this face. On the basis of this positioning scheme and its other parameters, an initial shape of the instance can be determined and placed in the specified position.

Second, if still required, this initial shape has to be fitted to the model. For cases as the one shown in Figure 2, the shape will have to be adapted to the model, retaining the properties imposed by the feature class specification.

Third, the resulting final shape will have to be represented in a feature model, in such a way that maintenance of its validity becomes possible.

All these issues are handled within the framework that is introduced in the next section. The framework is subsequently developed for the realization of attachment of freeform extrusion features, and attachment of freeform wrap features, in the following two sections.

4 A framework for attaching freeform features

The basic procedure for realizing an attachment is similar for all freeform features. Several steps can be identified that together form a framework that resolves the issues for freeform attachments that were enumerated in Section 3.

The first step is selecting one or more feature faces in the current feature model, to which the new feature instance will be attached. These faces are called the *model attach faces*. By selecting feature faces, instead of faces in a boundary representation of the model, the persistent naming problem occurring for boundary representations can be avoided [1]. As every face of every feature already in the model has been given a unique name, a face can always be referenced by this name.

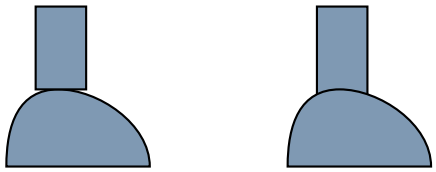
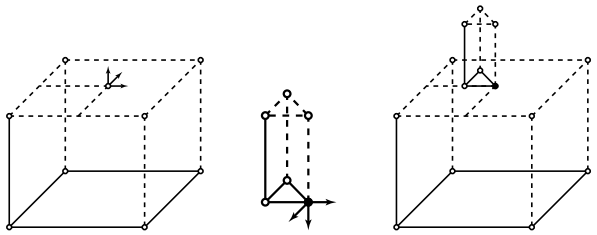


Figure 2. The feature shape in (a) has been fitted to the attach face in (b).

Then, it has to be established where the new feature instance will be positioned on the selected model attach face(s). A position on a freeform feature face can be determined by making use of the underlying mathematical representation of the face. A predefined position on the face, called its origin, is used as a reference for computing other positions. Using the parametric space of the surface, every position on the face is uniquely defined by two geometric distances along the surface, one for each parametric direction. The set of positioning parameters for a freeform feature depends on its feature class specification. A freeform protrusion may require just a single attach position on a model attach face, whereas a freeform rib may require several attach positions on multiple model attach faces. In each attach position, an *Attach Variable* is positioned (see Figure 3(a)). An Attach Variable is essentially a constraint variable that defines a coordinate system, thereby establishing both a position and an orientation.

The Attach Variables are fixed to the model attach faces, and as a result fixed to the global coordinate system of the feature model. They will therefore only be reevaluated when the feature instance to which the model attach face belongs is modified.

Corresponding Attach Variables are created on the *feature attach faces* of the new feature instance, on the basis of its feature class specification (see Figure 3(b)). During the constraint solving for the attachment, the configuration of FFDPs for the new feature instance is not only determined, it is also moved to the attach position by aligning the involved Attach Variables (see Figure 3(c)). When attaching an additive feature to the model, the new feature volume is directed outwards of the existing feature model. When attaching a subtractive feature, the new feature volume is directed inwards to the existing feature model. For some feature types, such as the rib mentioned above, the ini-



(a) (b) (c)

Figure 3. Performing an attachment on a feature model; (a) shows the outline of a feature model, with an Attach Variable positioned on its top face; (b) shows the outline for the feature that was referred to earlier in Figure 1, with an Attach Variable positioned on its bottom face; (c) shows the situation after the attach operation has been performed.

tial configuration of FFDPs is deformed whilst attaching it to the designated model attach faces. The constraint-based approach ensures that, throughout the modeling process, the Attach Variables stay aligned, thereby retaining the attachment. Alignment of Attach Variables is done by imposing freeform attach constraints between them. Freeform attach constraints enforce equal positions for the involved Attach Variables, but also fix their orientations relative to each other, considering the nature of features as described above.

Next, the initial geometry of the new feature instance is created. The procedure to do this is again fully specified in the feature class. An example is creating curves through the FFDPs and using them as input for a construction method such as sweeping (see Figure 1). The construction method then generates an *initial shape*.

After creating the initial shape, the attachment procedure may not yet be finished. Situations such as the one shown in Figure 2 can occur, where the attach faces do not connect seamlessly. Therefore, if required, the initial shape of the new feature instance has to be fitted to the face it is being attached to. This can, for example, be achieved by extending or blending the initial shape to the attach face. However, for cases such as the one shown in Figure 4, it is unclear how the attachment should be completed. Such cases are therefore considered *ill-defined* and cannot be completed successfully. Every type of attachment has its own scheme for detecting ill-defined freeform attachments.

After the attachment has been established, the validity of the resulting feature model, including the newly added feature, must be checked. If no constraints have been violated, the modeling operation is concluded successfully.

The framework described above can handle attachments for a variety of different freeform feature types. For two such feature types, the freeform extrusion feature and the freeform wrap

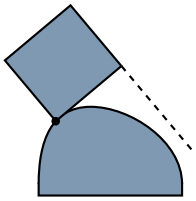


Figure 4. Ill-defined freeform feature attachment.

feature, this will be elaborated in the following sections.

5 Attachment of freeform extrusion features

In this section, the details of the framework for freeform feature attachments will be elaborated for the case of freeform extrusion features, which are constructed by sweeping a freeform profile curve along a linear path curve.

The first step during the attachment of such a feature is selecting one or, for through features, two model attach faces. One Attach Variable is created and positioned on each model attach face according to the values of the positioning parameters.

Subsequently, corresponding Attach Variables are created and positioned on the new extrusion feature. These Attach Variables are positioned relative to the configuration of FFDPs in the feature class definition. In addition, freeform attach constraints are imposed between the corresponding Attach Variables in the new feature instance and on the model attach faces. After the constraint solving for the attachment, the configuration of FFDPs of the new feature is positioned and oriented according to the model Attach Variables, also taking into account its nature.

Following the specification in its feature class, the geometry for the extrusion feature can now be generated. The result is a well-positioned *initial shape* for the new feature instance (see Figure 5). The feature will hold this position throughout the modeling process, as this is enforced with the constraint solver by the freeform attach constraints.

From Figure 5, it is clear that, after an extrusion feature has been positioned on the existing feature model, there is not necessarily a seamless connection between them. In some cases, such as shown in Figure 5(a), it seems natural that extra material is added to the new feature instance, whereas in other cases, such as shown in Figure 5(d), it seems natural to remove extra material from the feature model to which the new instance is being attached. For this reason, the initial shape is extended. This is shown for both a protrusion and a depression in Figure 6. As it must be ensured that the initial shape gets large enough, it is extended just beyond the bounding volume of the feature model. For through features, the shape needs to be extended in two directions, but the procedure is similar.

The *extended shape* has now been created. For referencing purposes during future modeling operations, the faces of the ex-

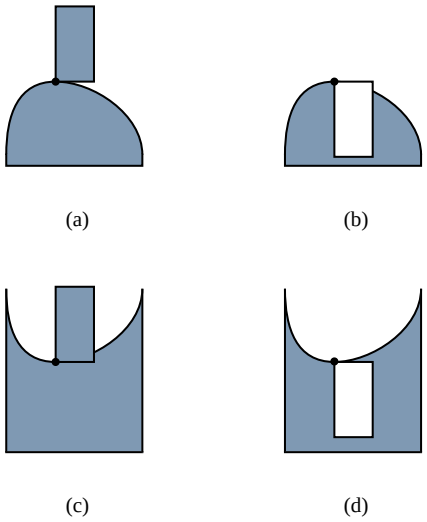


Figure 5. Four extrusion features that have been attached to a model; in (a) and (c) an additive (gray) feature is attached, in (b) and (d) a subtractive (white) feature.

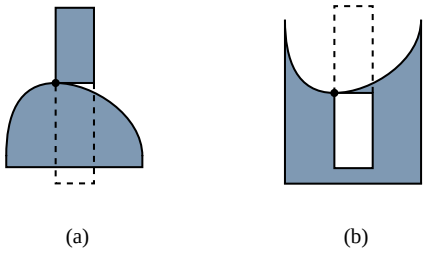


Figure 6. Generating the extended shape from the initial shape.

tended extrusion feature are then identified and given a name. It is important to note that the extended shape has two cap faces, which close the extruded surface. One cap face lies in the direction of the extension, whereas the other bounds the initial extrusion shape.

Obviously, the extended extrusion feature still does not seamlessly connect to the model attach face. For a well-defined attachment, the intersection of the extended shape with the model attach face contains a subface of the model attach face through the Attach Variable that divides the extended shape into two subvolumes, each containing one (complete) cap face. In order to determine the final shape, the extended shape must be trimmed along this subface. The cellular model that was introduced in Section 1 can be used to achieve this.

The procedure that is followed for a protrusion and a depression extrusion feature is shown in Figures 7 and 8, respectively.

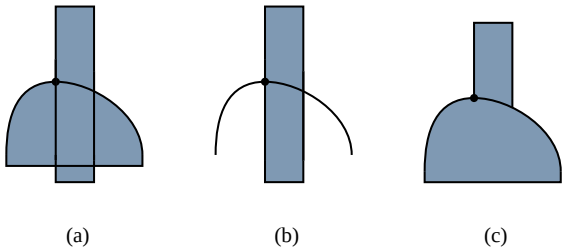


Figure 7. Extending and selecting the final shape for a protrusion; (a) shows the original model and the extended shape, (b) shows the extended shape and the model attach face, (c) shows the original model and the final shape, i.e. the feature.

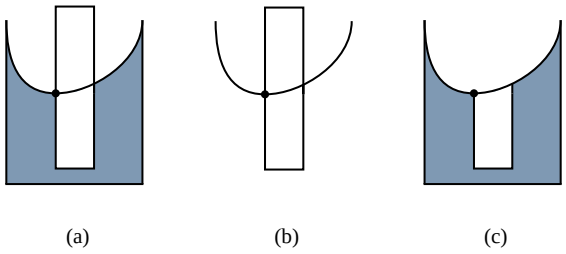


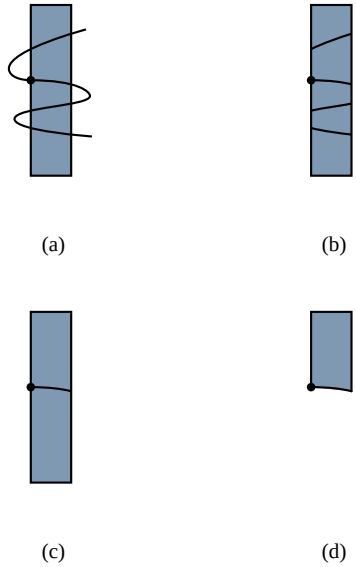
Figure 8. Extending and selecting the final shape for a depression; (a) shows the original model and the extended shape, (b) shows the extended shape and the model attach face, (c) shows the original model and the final shape, i.e. the feature.

After the initial shape has been extended (see Figures 7(a) and 8(a)), a non-regular union operation is performed between the model attach face and the extended shape. This results in two cells (see Figures 7(b) and 8(b)). The cell that contains the initial extrusion cap face is selected as the final shape (see Figures 7(c) and 8(c)). To rule out ill-defined attachments, it can be checked whether the extension cap face is also in the final shape; if that is the case, the attachment is ill-defined (see Figure 9).

In more general cases than in Figures 7 and 8, the described non-regular union operation may divide the extended shape into more than two cells (see Figure 10(a) and 10(b)). All cell boundaries that are shared by two cells are checked for containment of the Attach Variable corresponding to this attachment. Shared cell boundaries that do not contain the Attach Variable are removed from the extended shape, i.e. the corresponding cells are merged (see Figure 10(c)). Then, the cell that contains the initial extrusion cap face is selected as the final shape from the remaining cells in the extended shape (see Figure 10(d)). To establish whether the attachment is well-defined, it must still be checked that the selected cell does not contain the extension cap face. If

(a) (b)

Figure 9. Ill-defined attachments; in both (a) and (b), the extension cap face is in the final shape.



(a) (b)

(c) (d)

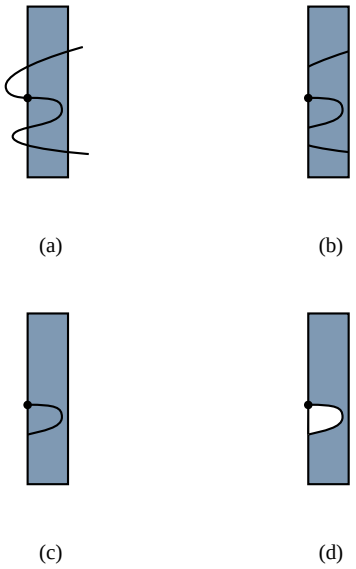
Figure 10. Selection of the final shape; (a) shows the extended shape and the model attach face, (b) shows the cells in the extended shape after the non-regular union operation, (c) shows the remaining cells in the extended shape after the merge operations, (d) shows the final shape containing the initial extrusion cap face.

this test is successful, the final shape is returned. Otherwise, the attachment is ill-defined (see Figure 11 for an example).

For through features, the procedure is similar. Both attachments are processed sequentially, once for both model attach faces. The resulting shape of the first attachment procedure is the basis for the second attachment procedure.

In Figure 12, the algorithm for selecting the final shape is given.

In order to finalize the attach operation, identification must take place for the newly created faces in the final shape. When the final shape was determined, some of the faces of the extended shape may have been clipped and some, namely the attach face(s), removed entirely. The clipped faces still have their identity, but the attach face(s) will need to be identified again. Fortunately, due to the freeform attach constraints that have been imposed during the attachment operation, the Attach Variable for the attachment lies on the attach face. For every Attach Variable it is known to which face it belonged in the initial shape, and thereby it is possible to assign a name to every unidentified feature face.



(a) (b)

(c) (d)

Figure 11. Determining an ill-defined attachment; (a) shows the extended shape and the model attach face, (b) shows the cells in the extended shape after the non-regular union operation, (c) shows the remaining cells in the extended shape after the merge operations, (d) shows the final shape containing both the initial extrusion cap face and the extension cap face.

```
PROCEDURE determine final shape

    FOR ALL model attach faces DO
        perform non-regular union operation with extended shape
        FOR ALL cells that share a cell boundary DO
            IF NOT attach variable on shared cell boundary DO
                merge cells

        IF two cap faces in same cell DO
            RETURN ill-defined attachment
        ELSE
            select cell with initial extrusion cap face

    RETURN final shape

END
```

Figure 12. Algorithm for determining the final shape from the extended shape.

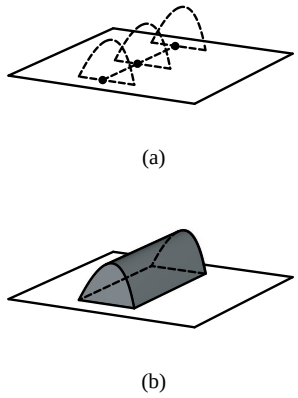
6 Attachment of freeform wrap features

In this section, the case of freeform wrap features will be demonstrated. For a wrap feature, a path is defined on a model attach face. Profiles are placed along this path, adapting themselves to the model attach face. With a skinning operation, geometry is created from these profiles, guaranteeing a seamless connection between the wrap feature and the model attach face.

The first step during the attachment of such a feature is again selecting an attach face. In contrast to a freeform extrusion feature, several Attach Variable are now created and positioned on the model attach face according to the values of positioning parameters. These variables specify the path of the wrap feature.

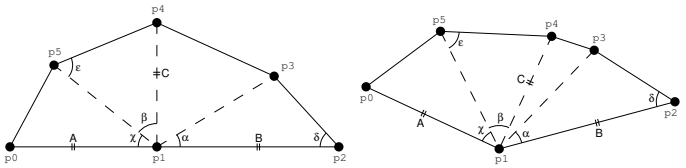
Second, specific FFDPs of the new feature instance are fixed to the Attach Variables. For a wrap feature, the FFDPs in the configuration prototype are not rigorously fixed relative to each other, i.e. several relative degrees of freedom remain. This allows the feature instance to be wrapped along the model attach face. Consider Figure 13(a), where three Attach Variables have been positioned on a freeform model attach face. Using freeform attach constraints, parametrized profiles that have been specified in the feature class are positioned in all three Attach Variables. In Figure 14, an example of such a parametrized profile is shown. For a freeform wrap feature with subtractive nature, the profiles will be directed into the existing feature model. For one with additive nature, the profiles will be directed outwards of the existing feature model. Subsequently, designated FFDPs in the profiles are positioned on the model attach face. For example, for the profile in Figure 14(a), Attach Variable p1 will be positioned on the path, and p0 and p2 will also be positioned on the model attach face. p1 is determined using the parametrized positioning scheme discussed earlier in Section 4. p0 en p2 are determined by creating a plane in p1, orthogonal to the path, and computing a specified distance from p1 along the intersection of this plane with the model attach face in both directions. The other FFDPs of the profile are related to the ones forced onto the attach face by constraints. As a consequence, during the constraint solving to realize the attachment, the FFDPs are repositioned according to these constraints, thereby deforming the profiles according to the semantics defined in the feature class (see Figure 14(b)). During the positioning of the profiles in the Attach Variables in the model attach face, situations can occur where Attach Variables lie outside the attach face. Such cases are considered to be ill-defined attachments.

Third, the final geometry for the wrap feature is created. A surface is spanned through all profiles by performing a skinning operation. Rail curves traversing the model attach face ensure a seamless connection between the skinned surface and the model attach face. Using the first and last profile, the model attach face, and the resulting skinned surface, a volume is enclosed. The result is a well-positioned shape for the new feature instance (see Figure 13(b)). The feature will hold this position throughout the modeling process, as this is enforced with the constraint solver



(a) (b)

Figure 13. Positioning profiles on a model attach face; (a) shows three Attach Variables that are positioned on the model attach face, and the profiles that are positioned in the Attach Variables, (b) shows the final shape of the freeform wrap feature.



(a) (b)

Figure 14. Constrained profile for wrap feature; (a) shows an initial profile that is defined in the feature class, (b) shows a deformed profile, where p0, p1 and p2 have been positioned on the model attach face.

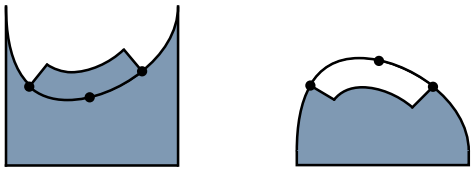
by the freeform attach constraints.

It is clear that now a seamless connection has been established between the new feature instance and the model attach face (see Figure 15). Contrary to the freeform extrusion features described in the previous section, no additional operation is therefore required. However, to ensure other properties, such a smooth connection between the freeform wrap feature and the model attach face, the feature class specification may prescribe other postprocessing operations, such as blending.

The final shape has now been created. For referencing purposes during future modeling operations, the faces of the feature are identified again and given a name.

7 Results and conclusions

A framework for constructing freeform volumetric feature models with attachments has been presented. A freeform feature



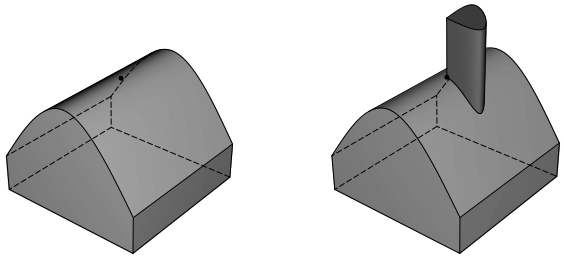
(a) (b)

Figure 15. Attachment of a freeform wrap feature; (a) shows an additive wrap feature attached to a freeform feature model, (b) shows a subtractive wrap feature attached to a freeform feature model.

instance to be added to a model can first be positioned on an attach face in the model in an intuitive way; the resulting position can be changed by adjusting the position parameters. If necessary, the thus defined shape is fitted to the model; the final shape has the properties imposed by the feature class specification. In the last step, the final shape is added to the feature model, which is a cellular model that can very well serve as a basis for validity maintenance [1].

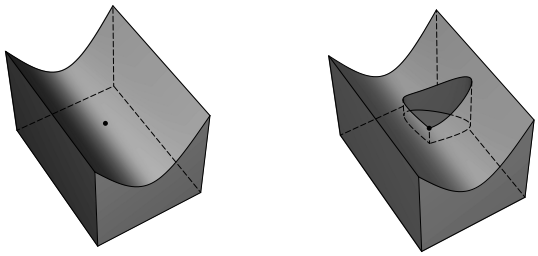
Several examples illustrate how the framework handles a variety of attach situations. The examples have been modeled and displayed with the SPIFF prototype feature modeling system that is being developed by our group [1], with the extensions for freeform feature modeling [8]. For each example, the attach positions selected are shown on the model, together with the resulting model geometry after the feature attachment has been performed.

1. *Freeform protrusion attached on a freeform base feature (Figure 16)*
   This situation has already been sketched in the previous sections (Figures 6(a) and 7). After the attach position has been selected on the attach face, see Figure 16(a), the final shape of the protrusion is computed and added to the model, as shown in Figure 16(b).
2. *Freeform depression attached on a freeform base feature (Figure 17)*
   This situation has also already been sketched (Figures 6(b) and 8). After specifying the attach position on the attach face, see Figure 17(a), the shape of the depression is computed and added to the model, as shown in Figure 17(b).
3. *Freeform through cut attached between two faces of a freeform base feature (Figure 18)*
   The two attach faces are first selected, and an attach position is specified on each of them, see Figure 18(a). The final shape of the through cut, extending between the two attach faces, is then computed and added to the model, as shown in Figure 18(b).
4. *Freeform through slot attached on a freeform base feature (Figure 19)*
   The top attach face is selected first, and two attach positions are specified on the intersection of the top attach face with two additional attach faces, the *from* and *to* attach faces, see Figure 19(a). The final shape is then computed and added to the model, as shown in Figure 19(b).
5. *Freeform rib attached on a freeform base feature (Figure 20)*
   The attach face is selected first, and two attach positions are specified, see Figure 20(a). The profile of the wrap is placed in these positions and deformed according to the attach face. The final shape is then computed and added to the model, as shown in Figure 20(b).



(a) (b)

Figure 16. Freeform protrusion attached on a freeform base feature.

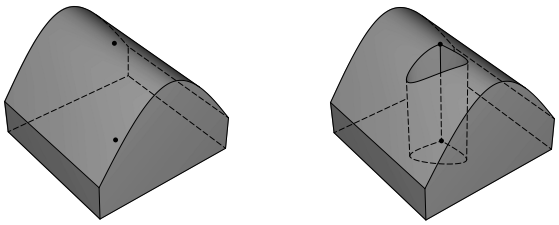


(a) (b)

Figure 17. Freeform depression attached on a freeform base feature.

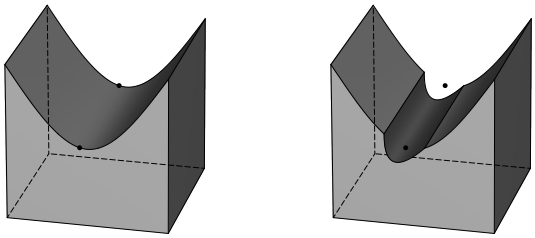
In the framework, the attractive properties of attachment of regular-shaped features have been realized for freeform volumetric features. Freeform feature models can now be constructed in the same way as regular-shaped feature models, i.e. in a fully parametrized, constraint-based way, which was the main goal of this work.

Many important validity conditions for freeform features can now be maintained, in particular conditions such as that the top face of a feature should remain open, and that a feature should not be split by another feature [1]. This has been



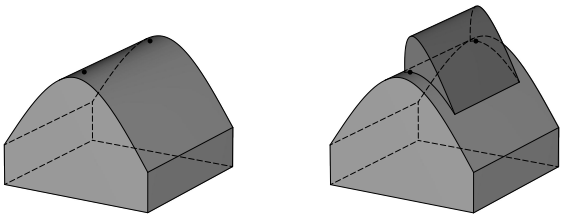
(a) (b)

Figure 18. Freeform through cut attached between two faces of a freeform base feature.



(a) (b)

Figure 19. Freeform through slot attached on a freeform base feature.



(a) (b)

Figure 20. Freeform rib attached on a freeform base feature.

realized by using a cellular model for the representation of a freeform feature model. Although originally developed for regular-shaped features, this cellular model could easily be accommodated for freeform feature modeling. The same validity maintenance mechanism that is used for regular-shaped features [1] can also be used for freeform features. However, in the context of freeform feature modeling, many more validity conditions may be considered, e.g. on geometric properties such as the curvature of faces of features. Definition and maintenance of such conditions is a major topic of future research.

The construction scheme for freeform feature models with attachments presented here, is a major step forward to realize a freeform feature modeling system that offers all functionality found in regular-shaped feature modeling systems and, in addition, validity maintenance specific for freeform features.

Acknowledgement

Eelco van den Berg's work is supported by the Netherlands Organization for Scientific Research (NWO).

We thank Hilderick A. van der Meiden for his work on prototype-driven constraint solving.

REFERENCES

[1] R. Bidarra and W. F. Bronsvoort. Semantic feature modelling. *Computer-Aided Design*, 32(3):201–225, 2000.

[2] R. Bidarra, K. J. de Kraker, and W. F. Bronsvoort. Representation and management of feature information in a cellular model. *Computer-Aided Design*, 30(4):301–313, 1998.

[3] X. Chen and C. M. Hoffmann. Towards feature attachment. *Computer-Aided Design*, 27(9):695–702, 1995.

[4] C. Durand and C. M. Hoffmann. A systematic framework for solving geometric constraints analytically. *Journal of Symbolic Computation*, 30(5):493–519, 2000.

[5] C. M. Hoffmann and P. J. Vermeer. Geometric constraint solving in R2 and R3. In *Computing in Euclidean Geometry, Second Edition*, pages 266–298, World Scientific Publishing, Singapore, 1995.

[6] J.-P. Pernot, B. Falcidieno, F. Giannini, S. Guillet, and J.-C. Leon. Modelling free-form surfaces using a feature-based approach. In *Proceedings of the Eighth ACM Symposium on Solid Modeling and Applications (Seattle, WA, USA, June 16-20, 2003)*, pages 270–273, ACM Press, New York, 2003.

[7] E. van den Berg, W. F. Bronsvoort, and J. S. M. Vergeest. Freeform feature modelling: concepts and prospects. *Computers in Industry*, 49(2):217–233, 2002.

[8] E. van den Berg, H. A. van der Meiden, and W. F. Bronsvoort. Specification of freeform features. In *Proceedings of the Eighth ACM Symposium on Solid Modeling and Applications (Seattle, WA, USA, June 16-20, 2003)*, pages 56–64, ACM Press, New York, 2003.

[9] G. Vosniakos. Investigation of feature-based shape modelling for mechanical parts with free form surfaces. *International Journal of Advanced Manufacturing and Technology*, 15(3):188–199, 1999.