

# Feature Model Visualization

Willem F. Bronsvort, Rafael Bidarra and Alex Noort

Faculty of Information Technology and Systems, Delft University of Technology, Delft, The Netherlands

---

## Abstract

*Feature modelling is now the predominant way of modelling products. Feature visualization is an important aspect here that can still be considerably improved. In this paper, an integrated way of visualizing feature models is presented, using new techniques for both the geometry and the structure of models. For the geometry of feature models, techniques are presented to visualize a selected subset of form features in a way that clearly distinguishes them from the rest of the model, as well as functional information such as closure faces of subtractive form features. For the structure of features models, techniques are presented to visualize several types of graphs. The different visualization techniques are used in an integrated way. Implementation of some of the techniques requires a non-manifold representation of the geometry of the feature model. This representation, and some other implementation aspects, are briefly described. Throughout the paper, numerous examples of images of feature models are given which show that the new visualization techniques can indeed improve the effectiveness of feature modelling.*

**ACM CSS:** I.3.7 Three-Dimensional Graphics and Realism—*visible line/surface algorithms*, J.6 Computer-Aided Engineering—*feature modelling*

---

## 1. Introduction

Feature modelling systems are more and more used for modelling products. These systems have the advantage that, besides geometry, also functional information can be stored in a product model [1]. Information that is relevant for a user depends on the way he interprets the product, i.e. his view on the product. In a part detail design view, it can be the function of some area of the part for the end-user of the product; in a part manufacturing planning view, how some area of the part is to be manufactured; and in an assembly design view, how the constituting parts of the product are related. In the first two views, usually form features are used to represent the functional information, in the last view connection features can be used for this purpose.

Although feature modelling systems have been considerably improved during the last years in many respects, e.g. the classes of features that are available and the interactive model specification facilities, the improvement in model visualization has been less profound. The most common way of visualizing a feature model is still to display the complete shape of the model with a single visualization technique, as

in traditional geometric modelling systems. Such an image, however, does not provide much insight into the shape, position and orientation of individual form features in the part, nor does it reveal any additional functional information that is present in the feature model. A similar image of a product consisting of several parts does not provide much insight into the separate parts and how these are related. Although some feature modelling systems can display a feature graph to show the feature model structure, the possibilities in this respect are usually rather restricted.

The goal of this paper is to present visualization techniques that give better insight into the specific feature information in a model. For the feature model geometry, techniques are presented to visualize (i) a subset of selected form features in a way that clearly distinguishes them from the rest of the model, (ii) closure faces of subtractive form features, and (iii) form feature intersections. To show the underlying structure of form feature and assembly feature models, several types of graph visualizations and operations on them are presented. The goal of these techniques is not to generate product images that look as realistic as possible, but images

that enhance the perception of the shape and structure of products.

The work presented here has been done in the context of a multiple-view feature modelling system, in which several views on a product are possible, each view with its own specific feature model of the product [2,3]. All views represent the same product and are therefore kept consistent. It will be shown how the feature models of the part detail design views, the part manufacturing planning views and the assembly design view are visualized in an integrated manner. This means, among other things, that a feature has the same colour in different images. The techniques for feature model geometry and structure visualization presented here can, however, also be used in feature modelling systems that do not support multiple views.

One might expect that all feature models are suitable for the visualization techniques presented here, because information on the individual features would be readily available in the models, but this is certainly not the case. Most feature modelling systems maintain a boundary representation of the resulting model geometry only, in which no information is available on the individual features in the model. Implementation of some of the techniques, notably visualization of closure faces of subtractive form features and intersections of form features, requires a more advanced representation of the feature model geometry, e.g. a non-manifold representation with attributes for its topologic entities. Such a representation, and other implementation aspects, will be described in some detail.

In our approach, there is much flexibility for the user to choose the specific features he wants to visualize, and the way these are visualized. Because the preferences of users for certain visualization techniques may vary, we believe it is best to give them as much control as possible.

Some of the material presented in this paper, in particular on feature geometry visualization, was introduced in preliminary form in [4]. Here we present an update of the original material, several extensions, and a simpler implementation. In addition, there is new material on structure visualization and, in particular, on the integration of geometry and structure visualization. This is the first approach integrating an extensive set of techniques for feature model visualization.

In Section 2, the techniques to visualize feature model geometry will be introduced, and in Section 3, the techniques to visualize feature model structure. In Section 4, integrated visualization of all views in our multiple-view feature modelling system will be discussed. In Section 5, the implementation of the visualization techniques will be described. In Section 6, some conclusions about the work will be given.

## 2. Feature Model Geometry Visualization

For feature model geometry visualization, form features are important. A form feature is defined as the representation of shape aspects of a product that are mappable to a generic volume and are functionally significant. Form features occur, for example, in part detail design and part manufacturing planning views. Typical examples are holes, slots and protrusions [1].

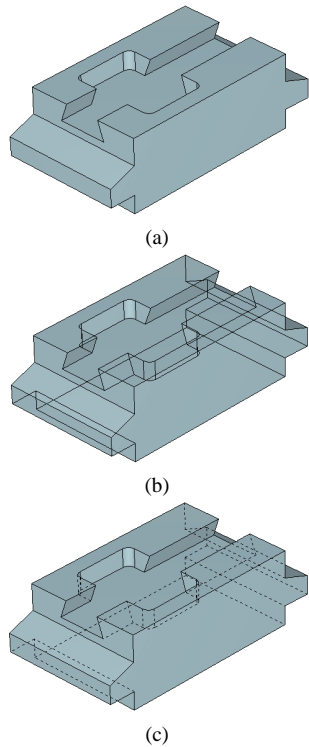
In current feature modelling systems, usually the resulting solid model geometry is visualized, with either a line image or a shaded image. A line image can be a wire-frame, a visible-line or a dashed-hidden-line image. Wire-frame and dashed-hidden-line images give a good geometric insight, in the sense that at least all topologic entities appear in the image. Visible-line and dashed-hidden-line images give a better spatial insight, because they provide more depth information. However, even for models of moderate complexity, line images contain so many crossing lines that it is often hard to get good insight into a model at all. Shaded-face images are more realistic than line images. A disadvantage is that edges are not explicitly visualized, but only implicitly by colour intensity transitions, which may not always be clearly visible. The insight given into a model is comparable to that given by visible-line images, since only visible entities appear in a shaded-face image.

An important visualization technique to support modelling, used in most systems, is highlighting. Entities, such as form features or model faces, that are selected by the user for some reason are highlighted with a special colour.

However, in the above-mentioned types of images, it is often hard to distinguish certain interesting geometry aspects, due to the large number of model entities visualized in the same way. Even worse, only the resulting solid model geometry is visualized, without any additional feature information; hence much information useful in a feature modelling context is not visualized at all. To avoid these drawbacks, several new techniques for feature model geometry visualization have been developed.

Since form features correspond to 'areas of interest' in a model, the techniques specifically visualize these areas of a model in a special way. A subset of form features can be selected, and this subset can be visualized differently from the rest of the model. In addition, functional information such as closure faces of subtractive form features, and intersection areas of form features, can be shown in an image. These techniques are here introduced with a short description and examples; their implementation is discussed in Section 5.

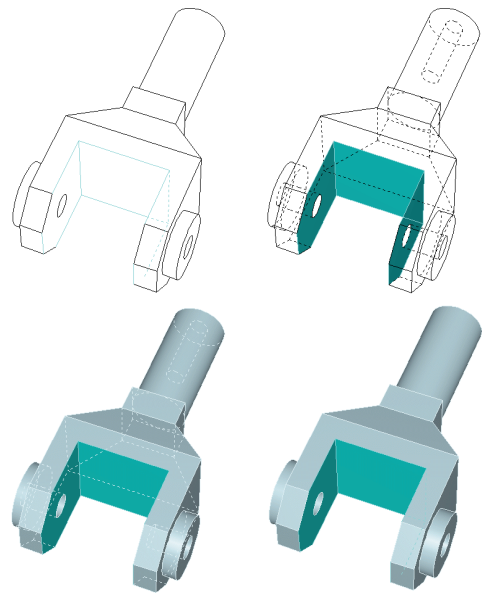
Visualization of the selected form features and the rest of the model can be done independently with any line visualization technique (wire-frame, visible-line or dashed-hidden-line technique), the shaded-face visualization



**Figure 1:** Hybrid line and shaded-face visualization with visible-line option (a), wire-frame option (b) and dashed-hidden-line option (c).

technique, or any hybrid shaded-face and line visualization technique (wire-frame, visible-line or dashed-hidden-line option). Since one of the problems of shaded images is that model edges are not explicitly visualized, the hybrid visualization techniques display a visible face segment by shading it and then superimposing its edges, a technique first suggested for CAD models by Forrest [5], and also used by others [6,7]. The invisible edges are either completely drawn (for the wire-frame option), not drawn (for the visible-line option), or drawn as dashed lines (for the dashed-hidden-line option). If the visible-line option is used, all visible edges are explicitly visualized; see Figure 1(a), in which the whole feature model is visualized with this option. In this image, the edges are indeed clearly visible. If the wire-frame option, see Figure 1(b), or the dashed-hidden-line option, see Figure 1(c), is used, extra geometric insight is added to the image. The hybrid visualization technique thus combines advantages of line and shaded-face visualization techniques.

Often a user wants to have a closer look at a particular form feature in a model only, e.g. because he wants to fine-tune its parameters. In Figure 2, several examples are given in which one form feature is visualized with one of

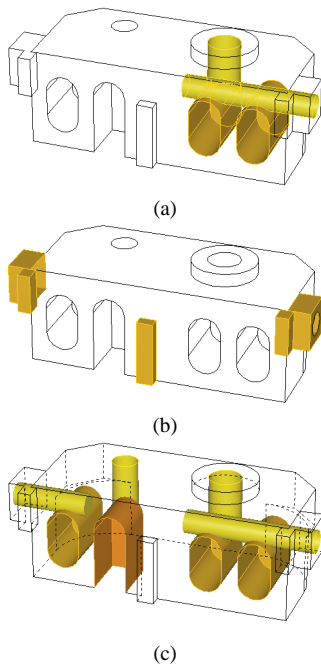


**Figure 2:** One form feature and rest of model visualized with different techniques.

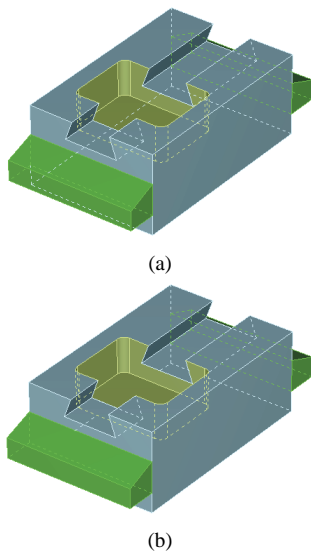
the above-mentioned techniques, and the rest of the model with another technique. This way of visualization offers extra insight into the selected feature, such as its shape and location in the model.

A user may also want to have a closer look at a selected subset of form features that have some property or relation. This subset and the rest of the model can again be independently visualized with one of the available techniques. Three examples are given here. If the user wants to investigate a certain spatial area of a feature model, e.g. to check its construction, all form features in that area can be visualized with shaded faces, and the rest of the model with visible lines (see Figure 3(a)). If he is interested in the distribution of all form features from a certain class, e.g. for manufacturability analysis, all instances of this class may be distinguished in this way (see Figure 3(b)). Finally, he can also have visualized form features that are not clearly visible from outside the model. This is a good alternative for cut-away images: the form features concerned are completely visualized, whereas the rest of the model is visualized in a less pronounced way (see Figure 3(c)).

Visibility of edges is normally defined with respect to the complete model. However, because the selected features are important, hidden model edges are here classified further according to whether they are hidden by a selected feature or by the rest of the model. The former are called occluded edges, the latter hidden edges. To get an even better insight into the selected features, at the cost of losing some geometric detail, the occluded edges may be omitted in



**Figure 3:** Selected subsets of form features and rest of model visualized with different techniques.



**Figure 4:** Visualization of model with (a) and without (b) occluded edges.

the image. In Figure 4(a) the occluded edges have been drawn, whereas in Figure 4(b) they have been omitted. The latter image gives indeed a better impression of the selected feature.

Another useful technique is to visualize not only the feature entities, i.e. faces and/or edges, that are on the model boundary, but also those that are not on the model boundary. The ‘on boundary’ and ‘not on boundary’ entities represent different functional aspects of the form features. For example, in a part manufacturing view, the ‘on boundary’ faces correspond to faces that actually have to be machined, whereas the ‘not on boundary’ faces delimit the material volumes to be removed. Such entities therefore may be visualized with a different technique. The colour of the ‘not on boundary’ entities can be, for example, a darker variant of the ‘on boundary’ entities, or a unique colour for all ‘not on boundary’ entities.

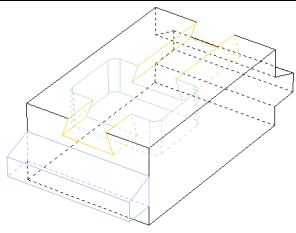
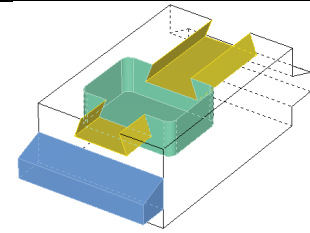
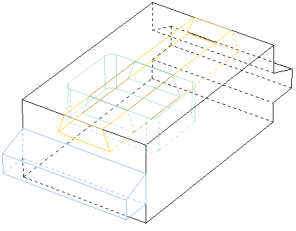
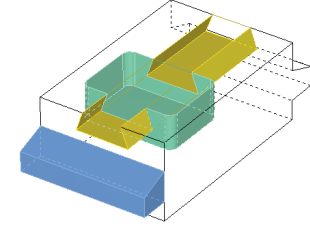
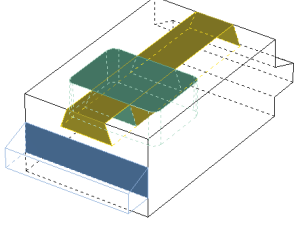
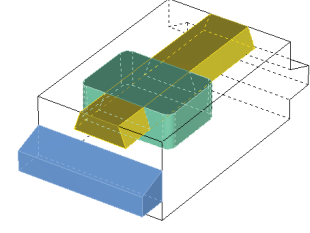
In Figure 5, six possible combinations of different visualization techniques for ‘on boundary’ and ‘not on boundary’ entities are shown. In all of them, three features have been selected for visualization: the pocket, the dove-tail slot, and the front rib. In the combinations shown, the ‘on boundary’ entities of the selected features are visualized by the dashed-hidden-line technique (left column) or the hybrid shaded-face and dashed-hidden-line technique (right column). Similarly, the ‘not on boundary’ entities of the selected features are either not visualized (first row), visualized by the dashed-hidden-line technique (second row), or visualized by the hybrid shaded-face and dashed-hidden-line technique (third row). By visualizing both the ‘on boundary’ and the ‘not on boundary’ entities, also the relation between two regions of one feature can be shown, e.g. that two apparent dove-tail slots (see first row in Figure 5) actually belong to a single dove-tail slot feature (see second and third row in Figure 5).

Feature intersections may involve another type of information useful to visualize. For example, in a part manufacturing planning view, feature intersections may indicate a less efficient process plan, because certain regions are processed more than once. In Figure 6, the model also used in Figures 4 and 5 is visualized with different colours for the two crossing features and their intersection region.

### 3. Feature Model Structure Visualization

In Section 2, several techniques have been presented to visualize the geometry of feature models. In this section, techniques will be presented to show the underlying structure of feature models, first for part feature models, and then for assembly feature models. In particular, graphs are used for both types of models. These techniques are here introduced with a short description and examples, and their implementation is again discussed in Section 5.

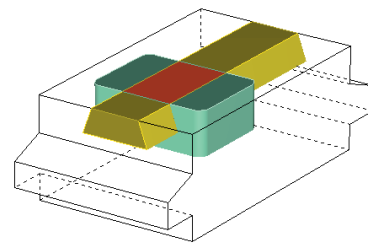
Part feature models, as they occur in part detail design and part manufacturing planning views, contain form features and constraints. In the corresponding graph, a form feature is visualized as a node with a name label or an icon. The icon is a symbolic representation of the feature class to which the

		ON BOUNDARY ENTITIES	
		Line	Hybrid
NOT ON BOUNDARY ENTITIES	None		
	Line		
	Hybrid		

**Figure 5:** Six combinations of different visualization techniques for 'on boundary' and 'not on boundary' entities of features.

instance in the node belongs. A form feature is attached and initially positioned relative to other form features with attach and positioning constraints. These constraints are visualized as directed arcs, from the latter form features to the first form feature. Feature model constraints are available to further constrain a model, e.g. to align the top faces of two form features. Such a constraint is depicted as a node with a name label or an icon symbolizing the constraint, and directed arcs between the form features and the constraint node. See Figure 7(a) for an example of a part, and Figure 7(b) for its feature model graph with icons for several form features and one constraint. A user can 'open' a form feature node or a constraint node, to get more information about the form feature or constraint (see Figure 7(c)).

Assembly feature models, as they occur in an assembly design view, contain components and connection features. A component is either a single component, which consists of a part, or a compound component, which consists of several subcomponents connected by connection features. Connection features contain information such as the type of the connection, the internal freedom of motion of the connection, and the form features on the components needed for the connection. An example is a pin-hole connection feature.



**Figure 6:** Visualization of intersection region.

In visualizing the geometry of an assembly feature model, each component can be given a separate colour (see Figure 8(a)). Another option is to visualize the geometry of two components with a line visualization technique, and the form features they need for their connection with the shaded visualization technique (see Figure 8(b)). Both options are useful in showing the structure of an assembly.

Hierarchical graphs of assemblies show the hierarchy of compound components and their subcomponents (see Figure 9(a)). They contain nodes with a name label or an icon to represent components, and directed edges between

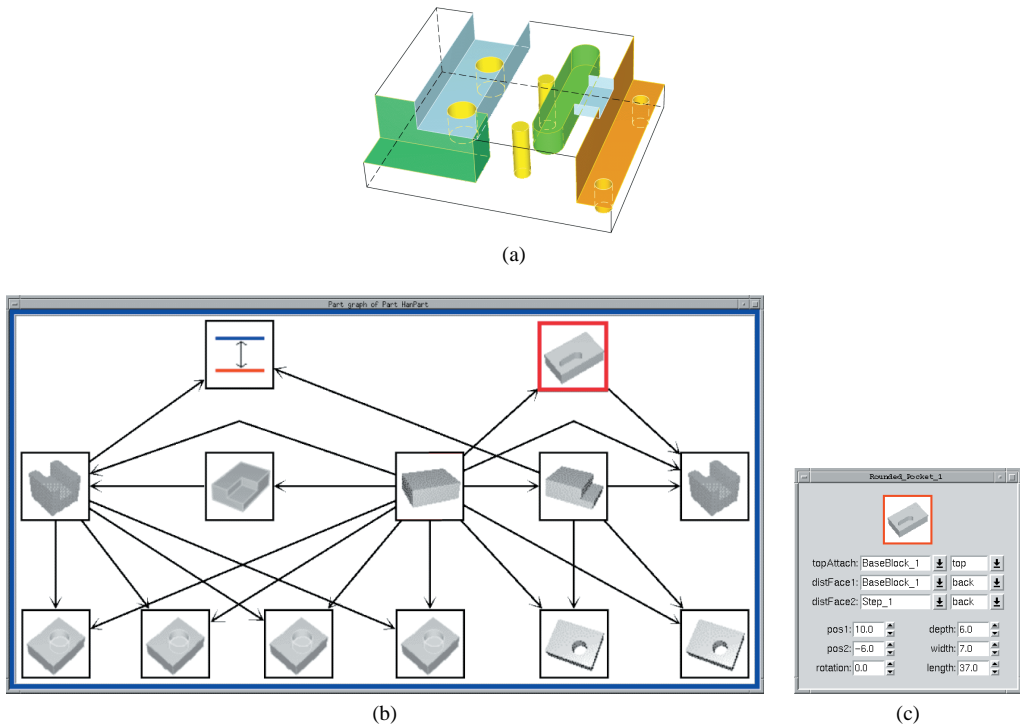


Figure 7: Part (a), its feature model graph (b) and information about the form feature in the opened node (c).

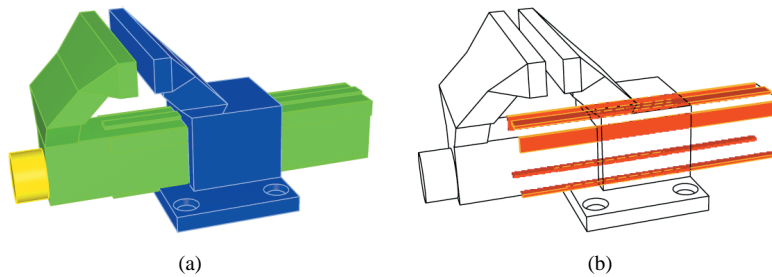


Figure 8: Different geometry visualizations of components in an assembly.

the nodes to represent the hierarchy between the compound components and their subcomponents. Nodes representing compound components can be expanded and collapsed to get a better insight into a model. Expanding a node for a compound component shows its subcomponents (see Figure 9(b)). Collapsing the nodes for an expanded compound component recursively hides all subcomponents.

Relational graphs of assemblies show the connections between components (see Figure 10(a)). They contain nodes with a name label or an icon for both components and connection features, and edges between the nodes to show how these are related, i.e. which components are connected by which connection features. Nodes representing

compound components can again be expanded and collapsed to get better insight into a model. When expanding a node for a compound component, here the node is replaced by nodes for its subcomponents and connection features (see Figure 10(b)). When collapsing the nodes for an expanded compound component, these nodes are replaced by a single node for the compound component, hiding all subcomponents and connection features of the compound component.

A part feature model graph can be used to select a form feature or a feature model constraint; a hierarchical graph of an assembly to select a component; and a relational graph of an assembly to select a component or a connection feature.



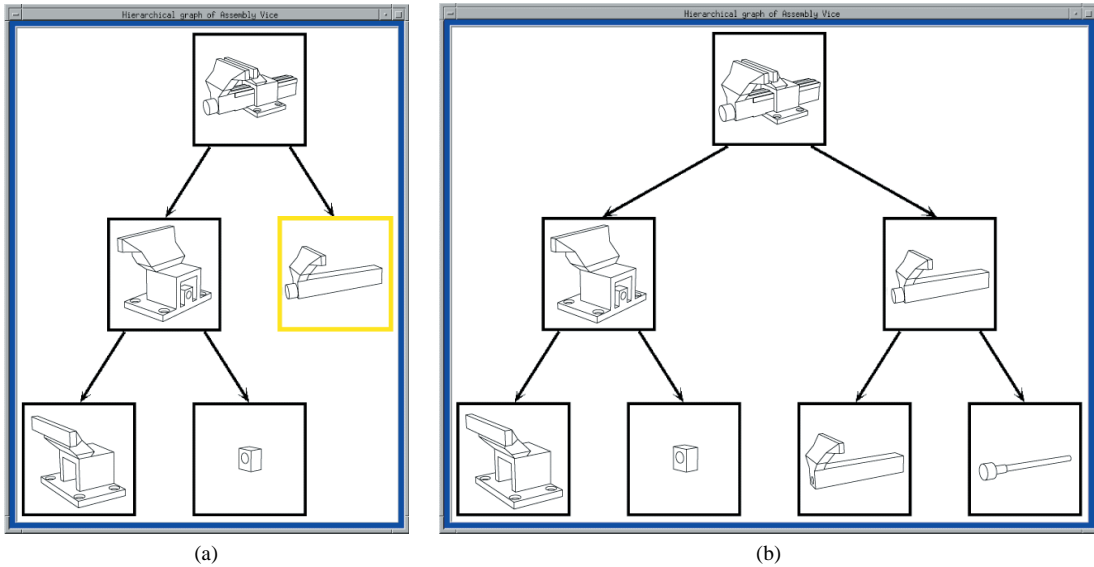


Figure 9: Original (a) and expanded (b) hierarchical graph.

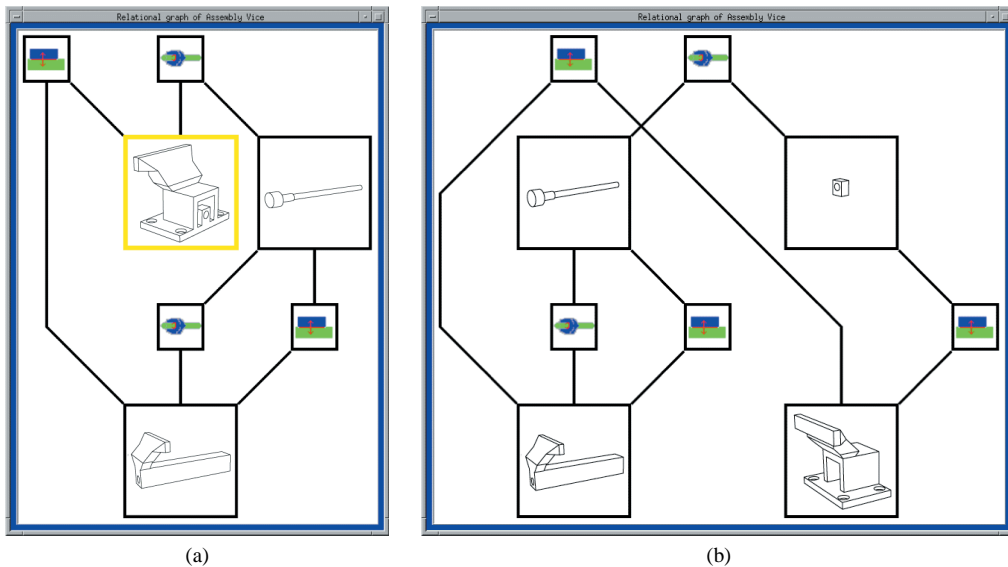


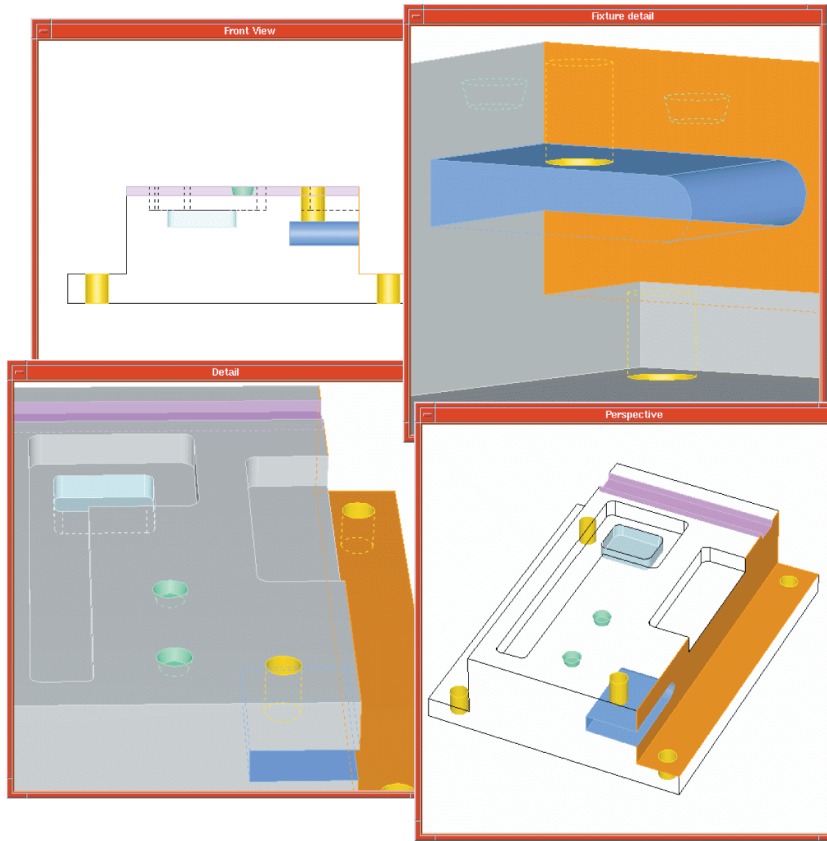
Figure 10: Original (a) and expanded (b) relational graph.

#### 4. Integrated Visualization of Feature Models

The visualization techniques described in the previous sections have been developed for the multiple-view feature modelling system SPIFF [2,3]. In this system, each product life cycle phase, or application, can have its own view on a product. Each view has its own feature model of the product, with features from an application-specific feature library. Currently, views are supported for part detail design, part manufacturing planning, and assembly design. All views on

a product are kept consistent by the system.

Integrated visualization of models is supported, both within one view and between different views. The feature models of the views can be visualized with two types of so-called cameras: geometry cameras show the model geometry, and graph cameras show the model structure, as described in Sections 2 and 3. Each feature and each camera has several visualization settings that can be specified by the user.



**Figure 11:** Geometry camera windows for part detail design view.

The settings for a feature include:

- whether it is selected for visualization;
- its colour.

A default colour is assigned to each feature class, but the user can change that colour, and all instances are then adapted accordingly. A user can, however, also assign a specific colour to each feature instance.

The settings for a geometry camera include:

- the viewing parameters (view reference point, center of projection, zoom factor, etc.);
- the projection method (parallel or perspective);
- the visualization technique (wire-frame, visible-line, dashed-hidden-line, shaded-face or hybrid with one of the three options) for the 'on boundary' entities of the features selected for visualization;
- the visualization technique for the 'not on boundary' entities of the selected features;

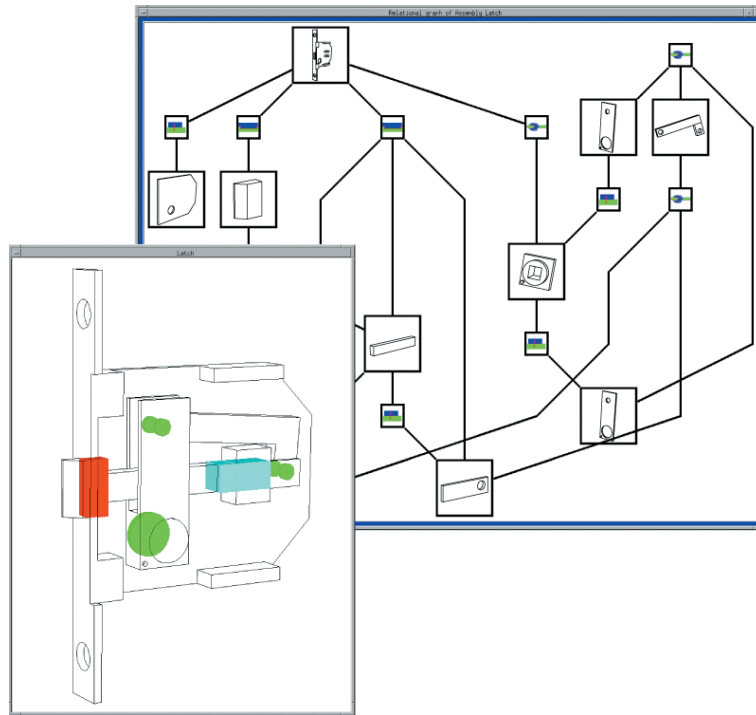
- the colour and visualization technique for the non-selected features of the model;
- the colour for the intersection regions of features;
- the background colour.

The settings for a graph camera include:

- the camera position;
- the zoom factor;
- whether a name label or an icon is depicted in the nodes of the graph.

Within one view, multiple cameras can be opened, each with its own settings. However, each camera uses the same set of selected features and the same colour settings for the features, which results in better model insight. In Figure 11, four geometry camera windows are shown for a part detail design view. In Figure 12, a geometry camera window and a relational graph camera window are shown for an assembly design view. Notice the similarity in colour settings in the four cameras in Figure 11, and the two cameras in Figure 12.





**Figure 12:** Geometry and relational graph camera windows for assembly design view.

Between several views, integrated visualization is also possible. An engineer can simultaneously work on multiple views on a product. For example, when he is working on the detail design of a part, he can check the manufacturability of some aspect of the part by opening the manufacturing planning view of the part. Camera windows for both views can be simultaneously displayed (see Figure 13). Notice that the windows for the part detail design and part manufacturing planning views show the same product geometry, but in terms of different features, which illustrates the basic idea of multiple-view feature modelling. The two feature models are kept consistent: when the user modifies the feature model of the part detail design view, the modification is propagated to the part manufacturing view, and the other way round. In either case, the modifications in both views are reflected in the corresponding camera windows.

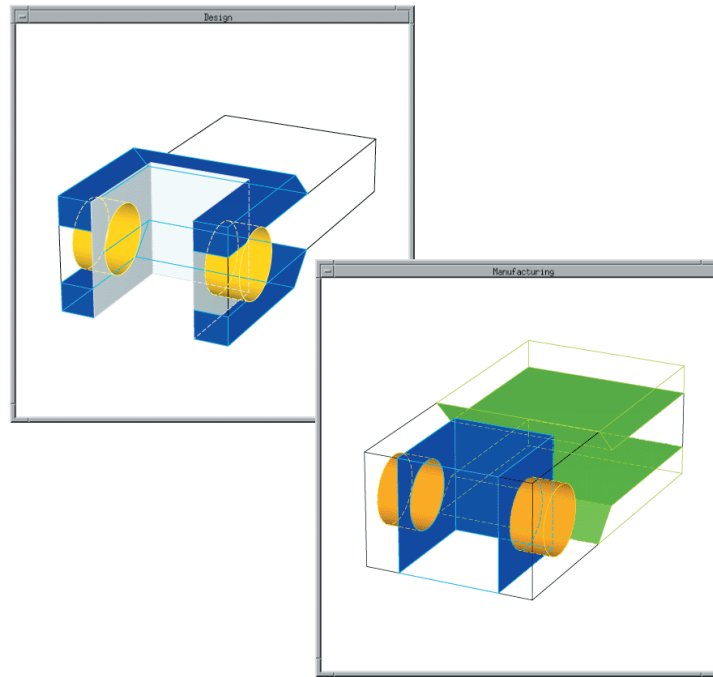
## 5. Implementation

To implement the visualization techniques discussed in the previous sections, feature information has to be present in the product model, in addition to the model boundary of the parts. We have implemented the techniques by using a Feature Dependency Graph and a Cellular Model for parts, and an Assembly Graph for assemblies.

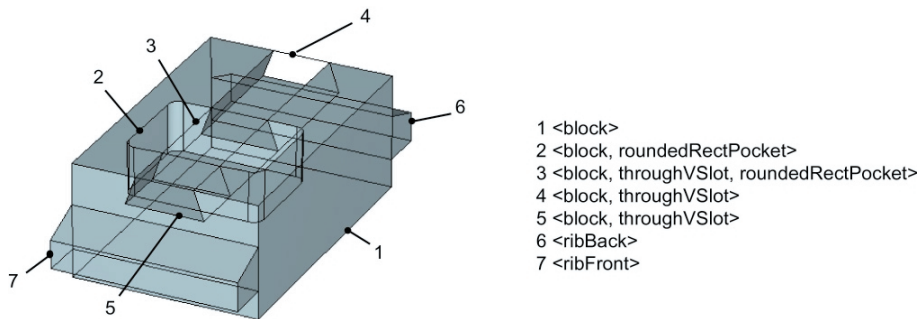
The Feature Dependency Graph for a part provides the high-level structure of the feature model. It contains exactly the information that is visualized in the feature model graph, as described at the beginning of Section 3 (see Figure 7). The Feature Dependency Graph is, among other things, used to generate and maintain the Cellular Model.

The Cellular Model is a non-manifold geometric representation of the feature model of the part, integrating the contributions from all form features in the Feature Dependency Graph. It represents a part as a connected set of volumetric quasi-disjoint cells of arbitrary shape, and represents each form feature as a connected subset of these cells. The cellular subdivision is determined by the property that two cells may not volumetrically overlap. Whenever two form features overlap, cells are created in such a way that one or more cells are shared by the two form features, and the rest of the cells lie in either of the two form features. As a consequence of the cell subdivision, each form feature face is represented by a connected set of cell faces. See Figure 14 for an example of a Cellular Model.

To identify form features and their faces in the Cellular Model, each cell and cell face has as attribute an owner list indicating to which form features, respectively form feature faces, it belongs (see again Figure 14). In addition, the nature of a cell expresses whether its volume represents 'material' of the part or not. Similarly, the nature of a cell



**Figure 13:** Camera windows for detail design and manufacturing planning views of a part.



**Figure 14:** Cellular Model of a part with owner lists of cells.

face expresses whether it lies on the boundary of the part or not. So the Cellular Model contains much more information than only the model boundary of the part. In particular, it contains information on the 'not on boundary' faces of subtractive features and on intersecting features, which is needed for some of the visualization techniques described in Section 2. The Cellular Model, including its attribute mechanism to maintain the owner lists of cells and cell faces, was implemented using the Cellular Topology Husk of the ACIS geometric modelling kernel [8,9].

The underlying data structure for assembly feature models is the Assembly Graph. It contains information on the way in which subcomponents are combined into

compound components, and on the way components are connected by connection features. Both components and connection features are represented by nodes, and the relations between them are represented by edges between the nodes (see Figure 15). Edges between nodes that represent components indicate a compound component built from multiple subcomponents and are directed. Edges between a node that represents a component and a node that represents a connection feature indicate a connection feature on the component and are undirected.

In addition to the above-mentioned feature information, the visualization settings for each feature and the camera settings, as discussed in Section 4, are available.

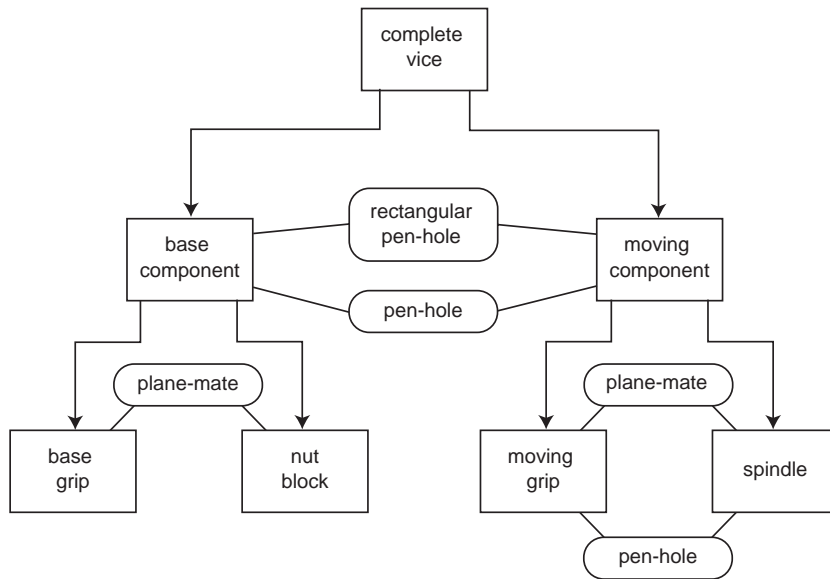


Figure 15: Assembly graph.

Feature model geometry visualization uses entities directly from the Cellular Model, i.e. no dedicated geometric model has to be maintained for visualization purposes. Instead, every time the feature model and/or the set of selected features is modified, all entities in the Cellular Model are analyzed and classified into the sets *Mfaces*, *FfacesOB*, *FfacesNOB*, *Medges*, *FedgesOB* and *FedgesNOB*. *Ffaces* and *Fedges* are cell faces and edges from selected features, and *Mfaces* and *Medges* are cell faces and edges from the rest of the model. *OB* entities are on the model boundary, *NOB* entities are not on the model boundary. *Mfaces* and *Medges* are, by definition, *OB* entities. For the classification, the owner lists of the Cellular Model entities are queried. Basically, a cell face is classified as *NOB* if it separates two cells of the same nature, and as *OB* otherwise; a cell edge is classified as *OB* if it bounds any cell face that is classified as *OB*, and as *NOB* otherwise. Table 1 summarizes the classification rules.

Depending on the geometry camera visualization settings, different combinations of these sets are input to the visualization process. This process involves the following steps.

*Rendering of faces* — The sets of faces for which the shaded or the hybrid visualization technique is selected are visualized using the capabilities of the ACIS Rendering Husk. The colour used to render a face is determined from its owner list: if it contains only one selected feature, its colour is used; if it contains several selected features, the colour specified for intersection regions is used.

*Hidden-line computation* — This step is required whenever, for any of the sets *MedgesOB*, *FedgesOB* and *Fedges-*

*NOB*, either a line visualization technique is selected other than wire-frame, or the hybrid visualization technique is selected with an option other than wire-frame. It computes the visible and hidden segments of the edges, using as occluding faces those in *Mfaces* and *FfacesOB*, and those in *FfacesNOB* in case these are also visualized. This computation makes use of the Precise Hidden Line Husk of ACIS.

*Occluded-line computation* — This step, again executed by the Precise Hidden Line Husk of ACIS, is required whenever occluded edges are distinguished (see Section 2). It computes the hidden segments of edges in *Medges*, using as occluding faces those in *FfacesOB*, and those in *FfacesNOB* in case these are also visualized. Drawing of these hidden segments is omitted in the next step.

*Drawing of edges* — Finally, the sets of edges for which a line or the hybrid visualization technique is selected are drawn. If in the first step already shaded faces have been visualized, the edges are superimposed on the shaded image. For the wire-frame technique and the hybrid technique with wire-frame option, all edges are drawn. For all other possibilities, for each edge the visible segments are displayed, and the hidden segments either omitted or drawn in dashed line, depending on the technique and option applied.

The above described the visualization of the feature model geometry of a part. Geometry visualization of an assembly feature model is done in a similar way.

Visualization of the feature model structure of a part is straightforward: the visualized graph (Figure 7(b)) is a one-to-one map of the Feature Dependency Graph. Visualization

**Table 1:** Classification of Cellular Model entities for visualization purposes

	Faces		Edges	
Model	<i>Mfaces</i>	<i>On boundary cell faces from non-selected features</i>	<i>Medges</i>	<i>On boundary cell edges from non-selected features</i>
Features selected for visualization	<i>FfacesOB</i>	<i>On boundary cell faces</i>	<i>FedgesOB</i>	<i>On boundary cell edges</i>
	<i>FfacesNOB</i>	<i>Not on boundary cell faces</i>	<i>FedgesNOB</i>	<i>Not on boundary cell edges</i>

of the feature model structure of an assembly can be done with the two graphs introduced in Section 3: the hierarchical and the relational graph. The hierarchical graph for an assembly or a compound component (Figure 9) is derived from the Assembly Graph (Figure 15) by omitting (i) all nodes that are not a child of the node representing the assembly or the compound component, (ii) all subtrees of children of this node representing a compound component that should not be expanded, and (iii) all nodes representing a connection feature. The relational graph for an assembly or a compound component (Figure 10) is derived from the Assembly Graph by (i) omitting all nodes that are not a child of the node representing the assembly or the compound component, (ii) omitting all subtrees of children of this node representing a compound component that should not be expanded, (iii) replacing all nodes representing compound components that should be expanded by its children, and finally (iv) omitting the node representing the assembly or compound component. All graph cameras have been implemented with Graphscript [10], a programming language for graph algorithms used in user interfaces based on Tcl/Tk [11]. Graphscript also determines the layout of a graph.

Integrated visualization of feature models, both within one view and between several views, is a straightforward extension of the above. All cameras for one view work on the one feature model of that view. The visualization settings of all features in the view are the same for all cameras, thus implementing colour similarity between the cameras, but the visualization settings for the cameras themselves can be different. Cameras for different views work, by definition, on different feature models of a product, and these models are kept consistent by the modelling system with underlying feature conversion techniques. All cameras can therefore work in the uniform way described above.

## 6. Conclusions

In the previous sections, numerous examples have been given of feature model images generated with the visualization techniques proposed here. The general

conclusion is that insight into feature models can be considerably improved by the use of these techniques. Some more specific conclusions are the following.

For visualizing the geometry of selected features or the rest of a model, good results are obtained with the hybrid visualization technique that visualizes both the faces and the edges. On the one hand, it gives a realistic image due to the shading of the faces, and, on the other hand, it emphasizes the edges and improves spatial insight by drawing them.

Visualizing a set of selected form features in another way than the rest of the model, emphasizes the shape and the location of these features in the model. This gives good insight into the way the model is built up.

By visualizing functional aspects, such as closure faces of subtractive form features and intersection regions of form features, insight into important engineering aspects is improved.

It is obvious that it is not possible to visualize all interesting aspects of the feature model geometry at the same time, and also that each user will have his own preference in the way these are visualized. Therefore the user can select a subset of form features he is interested in, determine the way in which these are visualized, and choose many other settings. These settings are not always straightforward, and an interesting research issue would be to automatically generate them, dependent on, for example, the complexity of the model, the viewing parameters and the application.

For visualizing feature model structure, the part feature graph and the hierarchical and relational graphs of assemblies are very suitable. Collapsing and/or expanding nodes of a hierarchical or relational graph as appropriate, can set it to an optimal size.

The use of multiple cameras for one feature view offers the possibility of looking to the model of the view in several ways. Multiple views on a product model are a good basis for a concurrent engineering system: several aspects of the model can be simultaneously considered. Integrated visualization of the views is very important to support the

users of such a concurrent engineering system. Currently, we are extending the multiple-view modelling system with web-based collaborative modelling facilities.

Altogether, a suite of techniques to improve visualization of feature models has been presented. Although the techniques have been described and implemented in the context of a multiple-view feature modelling system, most of the techniques can be implemented in any modelling system that stores enough feature information in its model. In this way, an important step towards more effective feature modelling is set.

### Acknowledgements

We thank Maurice Dohmen, Geoffrey Hoek, Winfried van Holland, Ruud de Jong, Klaas Jan de Kraker, Kees Seebregts, Jan Wouter Versluis and Marco van der Zwet for their contributions to our ongoing efforts to improve feature model visualization.

### References

1. J. J. Shah and M. Mäntylä. *Parametric and Feature-based CAD/CAM - Concepts, Techniques and Applications*. Wiley, New York, 1995.
2. W. F. Bronsvooort, R. Bidarra, M. Dohmen and W. van Holland. Multiple-view feature modelling and conversion. In W. Strasser, R. Klein and R. Rein (eds), *Geometric Modeling: Theory and Practice — The State of the Art*, Berlin: Springer, pp. 159–174. 1997.
3. A. Noort and W. F. Bronsvooort. Enhanced multiple-view feature modelling. In P. Brunet, C. Hoffmann and D. Roller (eds), *CAD Tools and Algorithms for Product Design*, Berlin: Springer, pp. 80–94. 2000.
4. J. W. Versluis, W. F. Bronsvooort, K. J. de Kraker and K. Seebregts. Feature visualization. In *CD-ROM Proceedings of the 1997 ASME Design Engineering Conferences*. New York: ASME, 1997. Held in Sacramento, USA, 14–17 September 1997.
5. A. R. Forrest. User interfaces for three-dimensional geometric modelling. In F. Crow and S. M. Pizer (eds), *Proceedings 1986 Workshop on Interactive 3D Graphics*, New York: ACM Press, pp. 237–249. 1986. Held in Chapel Hill, USA, 23–24 October 1986.
6. T. Saito and T. Takahashi. Comprehensible rendering of 3-D shapes. In *Proceedings Siggraph '90*, New York: ACM Press, pp. 197–206. 1990. Held in Dallas, USA, 6–10 August 1990.
7. A. Gooch, B. Gooch, P. Shirley and E. Cohen. A non-photorealistic lighting model for automatic technical illustration. In *Proceedings Siggraph 98*, New York: ACM Press, pp. 447–452. 1998. Held in Orlando, USA, 19–24 July 1998.
8. Spatial. *ACIS 3D Modeling Kernel, Version 6.2*. Spatial Technology Inc., Boulder, 2000.
9. R. Bidarra, K. J. de Kraker and W. F. Bronsvooort. Representation and management of feature information in a cellular model. *Computer-Aided Design*, 30(4):301–313, 1998.
10. M. Himsolt. The Graphscript Language. Faculty of Mathematics and Computer Science, University of Passau, <http://www.fmi.uni-passau.de/Graphlet/>, 1998.
11. J. Ousterhout. *Tcl and the Tk Toolkit*. Addison Wesley, Reading, MA, 1994.