Proceedings of
The 1996 ASME Design Engineering Technical Conference and
Computers in Engineering Conference
August 18-22, 1996, Irvine, California

96-DETC:DFM-1285

WHAT ARE FEATURE INTERACTIONS?

William C. Regli and Michael J. Pratt[1]
National Institute of Standards and Technology
Manufacturing Systems Integration Division
Building 220, Room A-127
Gaithersburg, Maryland 20899

ABSTRACT

Manufacturing features and their interactions have become areas of research scrutiny in recent years. It is widely accepted that intelligent reasoning about interactions among features is a critical element in the development of systems for automated manufacturing. Conversely, however, there has not emerged any general consensus as to what features are, how they are defined, and what it means when one says that features interact.

This paper attempts to focus and refine what is meant by the term *feature interaction* and outline precisely how interactions can affect automated feature recognition and manufacturing planning. It is our belief that, by establishing a conceptual common ground with regard to these concepts, the research community will be better able to assess how to effectively address the problems that they present.

Keywords: Process Planning, Feature-based Manufacturing, Manufacturing Integration, Feature Recognition.

INTRODUCTION

Research in computer-aided process planning (CAPP) and feature recognition has shown that interactions among features pose many complex problems to the developers of integrated CAD/CAM systems. While it is widely accepted that intelligent reasoning about interactions among features is critical for the development of automated manufacturing systems, there has not emerged any general consensus as to what features are, how they are defined, and what it means when one says that features interact.

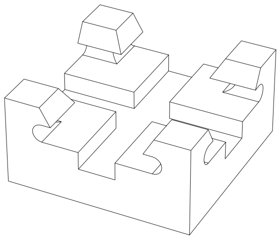
When two or more distinct features interact, individual feature instances can become distorted and information vital for reasoning about them (or recognizing them) can be eliminated. Figure 1 (a) shows an example of a solid model of a mechanical part containing a number of feature interactions. In particular, Figures 1 (b) and 1(c) show feature instances that one might like to find in order to perform effective manufacturing planning. In this example, however, the CAD model of the part does not seem to provide sufficient information from which to generate this volume.

Devising a general means for handling feature interactions has proven very difficult for a variety of reasons—chief among them is that there are differing notions of what it means for features to interact. This in turn has made it difficult to evaluate individual approaches. In most cases, interactions are handled with "rules of thumb" or with a proliferation of heuristics that attempt to capture each special configuration that might arise in a given application.

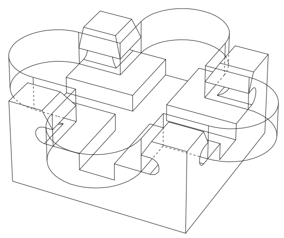
An added difficulty is the fact that features research is moving towards the use of complex multi-purpose models, using multiple feature sets, one for each different application area. Such models will give rise to new kinds of feature interactions, including interactions between features relating to different applications. How feature interactions are defined and analyzed has implications in many critical research areas, including standards and databases.

In this paper we argue for the development of a definition for feature interactions that is independent of the underlying feature representation. This paper attempts to focus and refine what is meant by the term *feature interaction* and to outline precisely how interactions can affect

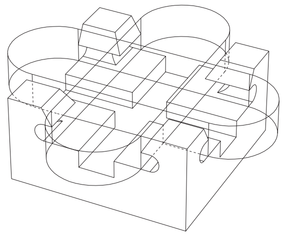
[1] Also affiliated with Center for Advanced Technology, Rensselaer Polytechnic Institute, Troy, NY



(a): part



(b): profile of a target feature



(c): alternative as crossing slots

Figure 1. A difficult interaction among features for machining. In this case the best manufacturing plan might include milling the x-shaped pocket or considering it as two distinct slotting operations. However, in either case, there are no faces left in the part from which to infer how to create the walls of these features.

automated feature recognition and manufacturing planning. We present several examples from both academic research and applications of commercial systems to illustrate different categories of feature interactions. It is our belief that the establishment of a common conceptual framework for such interactions will enable the research community to more effectively address the problems they present.

BACKGROUND

The ability to recognize interacting features has been a stated goal of a number research efforts, among them [4, 12]. However, while it is agreed upon as a problem of critical importance [28], the concept of **feature interactions** lies largely undefined in the general literature. In most cases, the definition for the term is implied, vague, or specific to the particular approach.

Some Definitions

There have been many attempts at a precise definition of the form feature concept. None of them has been totally successful. The implication seems to be that a somewhat elastic definition, whose interpretation may be stretched to accommodate a wide variety of different viewpoints, may be best for practical purposes. Accordingly, in this paper we will use the following definition:

Form feature: *A local geometric configuration on a manufactured part that has some engineering significance during the lifetime of the part.*

Many examples have been given elsewhere. Some writers have distinguished between form features and other classes of features having no 'shape' aspect; the *material features* and *precision features* of Shah et al [22, 23] provide examples. In the present authors' view information of this kind can be handled satisfactorily by the use of attributes without the necessity to invoke a more general concept of features. In what follows the word *feature* will therefore always denote a *form* feature.

It should be noted that the above definition covers not only features on the finished part but also ephemeral features that may be created and disappear during the manufacturing process. An example is a fixturing hole in a lug, created to facilitate an intermediate machining operation, where the lug itself is machined away at some later stage.

The definition generally covers only single (or *simple*) features. It needs to be supplemented with the following notions:

compound features: these are combinations of simple features, often of different types, and usually having some functional relationship. An example is a counterbored hole, regarded as being composed of a hole feature and a larger-diameter counterbore feature. However, it should be noted that, according to some viewpoints this is a simple rather than a compound feature. It is an important requirement of features technology that the user should be allowed to use his or her own preferred definitions of feature classes and feature types in any particular application context.

pattern features: these are regular spatial arrangements of features of the same type, in which the pattern has its own semantic significance. A circular pattern of cylindrical bolt-holes in a flange provides an example.

complex features: (this is *not* standard terminology, but such things must somehow be distinguished from the other cases). A complex feature as defined here is a combination of features *belonging to different parts*, which, taken together, play some engineering role. Two examples will be given:

1. A pair of rivet-holes, one in each of two metal sheets, together with the rivet fastening the sheets together, is a complex feature in the domain of fastening applications.
2. A round pin on one part, and a round hole on another, may form a complex location feature in the assembly domain.

How are Features Represented?

Most product modeling systems in widespread use today represent an artifact in terms of its faces, edges and vertices. Systems of the 2D drafting and 3D wireframe types do not contain face information, and may handle vertices (meeting-points of edges) either explicitly or implicitly. The various types of systems permit several ways of representing features, including the following:

2D drafting system: as characteristic patterns of edges in each of the three views of the drawing, which must satisfy requirements of spatial correlation when interpreted in 3D [14].

3D wireframe systems: as sets of edges in 3D; such sets will usually be connected, and the edges will define boundaries of the faces constituting the feature.

boundary representation solid modelers: (1) as sets of faces of the product model (again, these will usually but not invariably be connected sets). Such face-sets define *surface features*, corresponding to subregions of the overall boundary of the product model.

boundary representation solid modelers: (2) as self-contained volumes in their own right, each feature volume having some part of its boundary lying on the boundary of the product model. In general, the feature volumes will possess one or more faces that do *not* lie on the part boundary. These have been given various names including *closure faces, virtual faces, entrance faces* and so on.

These kinds of representations are said to be *explicit* [27]. They may be contrasted with *implicit* representations which are essentially procedural, usually specifying how the volumetric shape of a feature may actually be constructed in terms of a 2D profile and a linear or rotational sweep operation. Such representations may be *parametric*, key dimensions being specified as variables, and in this case they may be used as a basis for feature libraries in CAD systems. A feature instance on a product model will then be created by assigning specific numerical values to the dimensional parameters and positioning and orienting the instantiated feature appropriately on the model. It is also possible to define explicit parameterized feature descriptions in terms of geometric and topological elements of boundary representation models and constraints and other relationships between them. This flexible type of feature class description is usually termed a *feature definition language* [3, 13, 20, 24].

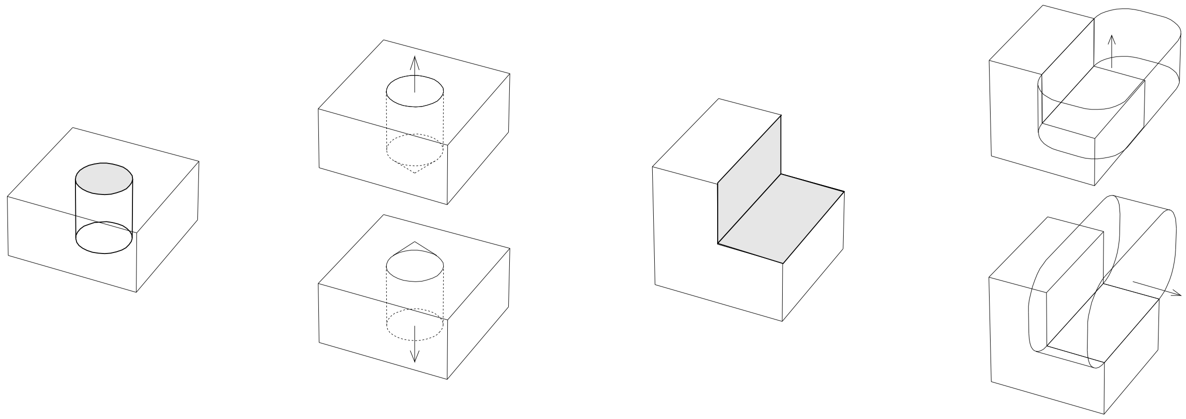
A final distinction which is often useful is between *canonical* and *installed* feature shapes. The first is the 'ideal' shape of a feature which results, for example, from the instantiation of an implicit feature from a feature library. The second is the shape of the feature as it actually occurs on the model, which may differ from the ideal because of interactions with the part surface or with other features. By way of example, consider a hole drilled normal to a cylindrical surface. The canonical form of a drilled hole feature is a cylinder with one planar and one conical end. However, because in this case the hole is drilled in a non-planar surface the installed version of the hole (which might be thought of as the volume of material to be removed) will have the planar end replaced by a face lying on the original cylindrical surface. The distinction between canonical and installed feature volumes is useful in analyzing feature interactions.

Neutral and Process-specific Features

It is sometimes desirable to think of localized shape configurations on a part model in pure geometric terms. In this case we can talk of a *neutral feature*. On the other hand, there is a wide spectrum of engineering activities, each of which has its own view of *process-specific* features. For the designer a feature provides functionality, for the machinist it indicates volumes of material to be removed (or, in some cases, retained), for an assembly planner it represents a region where the part bearing the feature will mate or otherwise connect with a corresponding feature on another part, and for an inspection planner a feature may represent a pattern of measurement points.

In general, each application domain will have its own distinctive decomposition of the CAD data into features that are significant in particular context. Thus, as is now well known, a design feature will not in general also be a machining feature (though sometimes it may), and so on. Some examples of this distinction are given in Figure 2.

The manufacturing features on a part depend on the intended mode of manufacture. Manufacturing processes that have been modeled with features so far include machining, sheet metal processing, casting, die-casting, injection molding and assembly [1, 7, 10, 16, 21]. One problem that has not so far been addressed in the literature is that of feature-based design-for-manufacture when the intended manufacturing process is not known to the designer. Once the process is subsequently chosen, it may be necessary to



(a): neutral form feature: hole (b): process-specific form features: drilling (c): neutral form feature: shoulder (d): process-specific form feature: milling

Figure 2. Distinction between neutral and process-specific form features (from [7]).

make adjustments to the part design to adapt it better for that particular mode of manufacture and reduce the costs of production. This requires the solution of a basic problem of *concurrent engineering*—the simultaneous design of products and processes.

Typically, concurrent engineering requires the study of feature interactions across manufacturing domains. For example in machining, the part setup on the machine tool is partly determined by the tool access orientations of machining features. Setups require the subsequent determination of fixturing and clamping features. But a clamping feature must be chosen so that the clamp will not obstruct a machining feature to be processed in that setup.

TYPES OF FEATURE INTERACTIONS

Feature interactions pose a major challenge in systems for feature recognition and feature-based manufacturing. The question arises as to what *interact* means. For example, in the machining domain, given that two features f_1 and f_2 intersect volumetrically, a process planner has to make a choice about how to machine the shared volume. This kind of feature interaction can only be handled satisfactorily when manufacturing attributes (such as tolerances, surface finish, and the like) are considered.

We here explore a preliminary classification into three types, based on *spatial* relationships, initially without reference to any application domain:

Interference interactions: These occur when the canonical volumes of two features have a non-null intersection, or to put it another way there is some volume shared by the two features.

Adjacency interactions: These are characterized by the adjacency of two features, often belonging to different parts in an assembly context. In this case the canonical or installed feature volumes effectively coincide over part of their boundaries.

Remote interactions: Here the two features are totally disjoint, but there is nevertheless some relation between them that affects the way they can be handled in their application context. This relation may be explicitly specified by the designer, or may arise in some downstream application context as an unintended result of a design decision.

The following subsections discuss the three types of interaction in a little more detail.

Interference interactions

With interference interactions there are two possibilities. First, two or more features might have their surfaces overlap, as for example when two gripping features for robotic assembly planning share some common area. Second, two or more features might share a common volume, such as illustrated previously in Figure 1 (b). When two or more volumes interpenetrate they generate one or more 'intersection loops' of edges resulting from intersections of faces of the first volume with faces of the second (a fact well known to practitioners of solid modeling). If a feature is represented as a surface feature rather than a volume feature then some of the faces of the canonical representation of each feature will be modified by the intrusion of parts of the intersection loop. Such interactions may also be characterized when the representation used is a 2D drawing—an example will be given later to illustrate this fact.

Another effect of interference interactions is that they may modify or destroy the *hints* or *traces* that are used by certain feature recognition algorithms [19, 8, 9, 25]. Intuitively, a **trace** represents the partial information remaining in the solid model of the part produced by an instance of a feature. A trace can comprise geometry and topology, design features, tolerances, and other design attributes associated with the CAD model. Most existing work in this area has thus far only considered geometric and topological traces and some cases of their interactions.

When features overlap or intersect spatially their traces can become distorted: portions of the same surface might be traces for multiple features of the the same or different types; a local geometric configuration might be realizable via several different manufacturing operations, possibly involving different manufacturing processes all with overlapping feature traces. These interactions affect rules and algorithms based on the traces and influence which feature instances get produced. Given any specific trace-based method, the information required for the generation of a specific feature instance might be removed or distorted in such a way as to make the feature instance unidentifiable within a given system.

For example, if one or more volumetric machining features $f_1 \ldots f_n$ interfere with the traces of another feature, f_0, they may distort or destroy information critical to the algorithmic reconstruction of f_0. If so, it may be more difficult or even impossible to deduce the existence of f_0 from the information remaining in the design (geometry, topology, design attributes etc.). An example of such an interaction would be a milling feature which leaves as its only trace a single planar side face in the delta volume [2]. In this situation there is no direct evidence from this side face that can determine a unique orientation for the feature. In practice for these degenerate cases, heuristics are used to infer plausible orientations based on other information available.

Adjacency interactions

Adjacency between two features on the same part may be taken to mean that they are in canonical form, having no volumetric interaction, and share one or more boundary edges or faces. Adjacency is of greatest practical concern when the two features are on different parts. They may be, for example, mating parts in an assembly, and the adjacency interaction is then concerned with the way they fit together, how they are held together, and any tolerance information relevant to their interaction.

Remote interactions

In these case the features are not in any sense adjacent or overlapping, but they interact either through some functional relationship or because they are both simultaneously significant to some process downstream of design. The functional relationship may be somewhat implicit. For example, several pockets may share a common planar surface for their floor faces; if this surface is moved up or down, the depths of all the pockets change at the same time. The common floor surface probably reflects some related functionality of the pockets, whether or not this was explicitly stated by the designer.

Life-cycle interactions

The foregoing is a *spatially-based* classification of feature interactions. A more or less orthogonal classification scheme may be based on interactions in multi-purpose feature-based models between features in different application areas of through the life-cycle of a product [6]). This includes relationships between features from different manufacturing domains (such as machining and assembly), and also from post-manufacturing life-cycle activities such as maintenance and disposal.

Early work on feature interactions concentrated on interactions within the same domain. For example, the very nature of the machining process is such that different feature-generating operations frequently interact with each other, making it difficult to treat features in isolation. An additional complication is due to the fact that there usually exists more than one way of manufacturing the same part, i.e. multiple machining features may be associated with a single neutral or geometric feature. However, if we consider interactions between features in *different* application domains, at least three categories of life-cycle interactions may be identified:

Manufacturing plan-level interactions: These are interactions that occur among features employed in the generation and evaluation of manufacturing plans. For the most part, these interactions occur at the level of individual components. For example, interactions might induce precedence constraints on operations such that one feature has to be manufactured prior to another. Accessibility criteria based on the configurations of the manufacturing domain, tools, and facility (i.e. Can the robot arm re-position the workpiece? Must the pocket be machined to get at the hole?). The designer might assign tolerances to an artifact which, when translated

[2] In the machining domain the convention is that a **machined part**, P, is a solid object to be produced from an **initial workpiece** or **stock** S by a finite set of machining operations, which create the geometry of the part. These operations are modeled as features. The **delta volume** (Δ) is the total volume of material removed, i.e., the regularized set difference between the stock and the machined part: $\Delta = S -^{*} P$.

into manufacturing tolerances may render certain features invalid or unrealizable. Due to the lack of available fixtures, some features might have to be machined in different set-ups. These represent but a few examples of plan-level remote interactions.

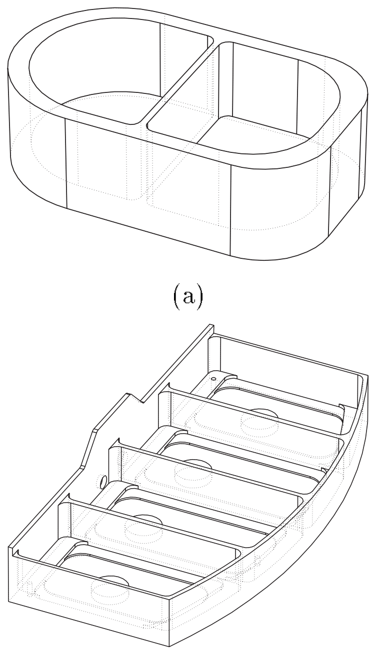
Manufacturing production-level interactions:
These are interactions that occur due to interactions among features during the production process. This includes interactions among features on multiple components (i.e. an assembly) or across multiple manufacturing processes on the factory floor. For example, consider a part that is going to be cast and then machined. The parameters on the first process and its features affect the features on the second process. In these situations, considering interactions over a number of processes requires an evaluation of trade-offs among interacting features from different manufacturing domains. Another type of production-level feature interaction can occur when features affect the integration of a process plan for an artifact into a factory or shop floor schedule. Features may interact with scheduling constraints (i.e. to make this part with these features might pose an unacceptable burden on some particular manufacturing resource, creating problems in manufacturing other features on other parts needed in the shop). Such production-level interactions can certainly also include constraints imposed by other inventory and resources—such as might be tracked in product data management tools. For example, cutting tools might be unavailable or worn, stock material might not be available, delivery times and release schedules might conflict, etc.

Downstream interactions:
There can be interactions among features throughout the product life-cycle. Features related to manufacturing criteria might have an influence on the features of the part that impact its maintainability and disposal. For example the selection of fasteners during design and assembly planning greatly influences maintenance and disposal cost by making a device difficult (or easy) to dismantle and repair.

Existing research efforts have begun to address manufacturing plan-level interactions; however, the problem posed by production and life-cycle interactions have not been significantly addressed by the research community.

EXAMPLES

Existing work has addressed the interactions problem in a number of ways, usually touching on one or more of the above levels. In most cases, however, it is usually not made clear which type of feature interaction is intended. This section provides several examples of feature interactions. In each case, the type of interaction may have aspects of one or more of the interaction categories outlined above.



(a)

(b)

Figure 3. Two example parts having thin walls.

Thin Walls and Machining

This well-known example (see Figure 3 (a) and (b)) illustrates feature interaction at a distance, albeit only a small distance. The designer's features for this part are a 2D base shape, a peripheral flange, a stiffener across the middle, and some corner rounds. The defining parameters will include the base thickness, the height and thickness of the flange and stiffener, and the corner radii. From the manufacturing viewpoint the features of the part are a peripheral profile and two pocket features. However, there is an interaction between the two pockets because the stiffener separating them is a thin wall which may deform under cutting forces when the second pocket is being machined, possibly leading to unacceptable inaccuracy in the stiffener thickness. It is interesting to note that while the pocket features exist in the machining domain it is the stiffener feature from the *design* domain which dictates a choice of cutting strategy to avoid this problem.

Locational Tolerances

In the machining domain, the determination of a machining strategy for an isolated feature is often a simple matter. However, when a feature interacts with one or more other features then the relationships influence the possible manufacturing solutions. An example of an explicit relationship, deliberately imposed by the designer, may be a tolerance of parallelism between one wall of a rectangular pocket and a planar peripheral face of a part. Essentially, the planar face is being used as a datum feature, and the designer has *specified* an interaction between it and the pocket feature. This is not dissimilar to the previous example except that the inter-feature relationship there was not explicitly spelled out by the designer. Interactions which are unintentional by-products of the designer's actions in this way have been referred to by Mill et al. [15] as *implicit* interactions.

To further illustrate this type of interaction we present two examples from Gupta [5]). Figure 4 shows a part with two drilling features, holes h_1 and h_2, requiring a tight concentricity tolerance. In this case the hole is too long to be drilled in with a single drilling operation (the length/diameter ratio is too large). Further, the interaction among the tolerances of the holes creates a manufacturability problem: drilling h_1 and h_2 with two operations at different setups will make the tolerances unachievable.

In the second example from Gupta [5], shown in Figure 5, there is a tight perpendicularity tolerance between the side face of slot feature s_1 and the side face of the step feature s_2. In order to create an entrance face to machine the hole h, s_2 should be machined before h; however Gupta points out that in order to satisfy h's approach condition requirements, h should be machined *before* s_1. Note that h has a different set up direction from s_1 and s_2, therefore s_1 and s_2 cannot be machined in the same setup. Hence this part is unachievable on a three-axis vertical machining center.

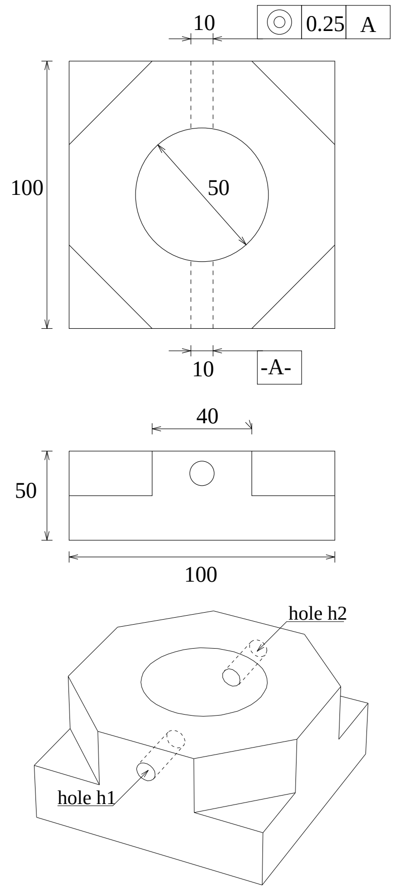


Figure 4. An Example of a design with dimensions and tolerances that cannot be achieved on a 3-axis vertical machining center (from [5]).

Assembly

An example from the assembly domain is shown in Figure 6. Here a cap (Figure 6 (a)) fits over the projecting end of a spindle (Figure 6 (b)) as shown in Figures 6 (c) and (d). From the assembly point of view the mating relations between corresponding features of both parts must be considered. There is a adjacency interaction between the flange features on the spindle and on the cap, and also between the projection feature of the spindle and the inner hole of the cap. Furthermore, the interaction in each case involves the whole of one mating surface but only part of the other. Such a situation is typical of the inter-feature relationships arising in assembly modeling. The interaction would be more complex if a fit criterion were applied to the relationship between the parts. In the case of a force fit, for example, the diameter of the pin would be marginally larger than that of the hole initially, so that actual material deformation is involved in assembling the two together.

This illustrates but one simple interaction of assembly features. Feature interactions in the context of design-for-assembly and assembly planning have been, for the most part, under-addressed by the research community.

Feature Recognition

Interactions when recognizing features in 2D drawings. Consider the problem of feature recognition from 2D drawings. Features manifest themselves as characteristic profiles in each of the three views, and these profiles must correlate spatially when the drawing is interpreted in 3D [14]. Volumetric interference between two features

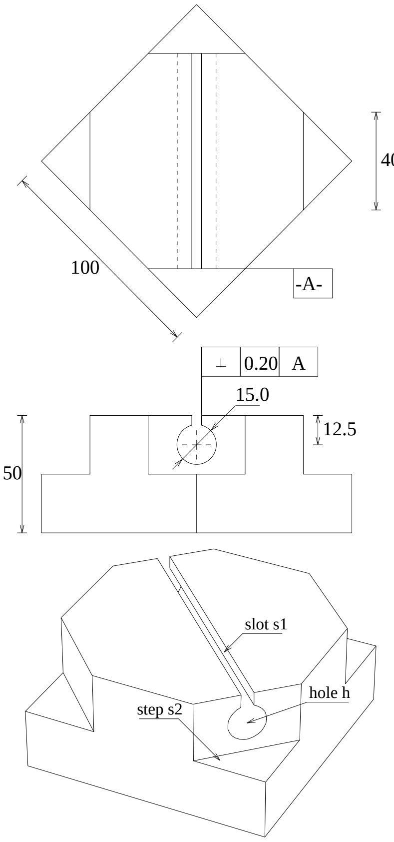


Figure 5. A second example of a design that is unrealizable on a 3-axis vertical machining center due to interactions among dimensions and tolerances on various features. Note that this design can be machined on a horizontal machining center—hence feature interactions can arise from the particular hardware configuration (from [5]).

shows up in the modification of feature profiles in one or more views, and geometric reasoning must be used to reconstruct the full profiles to enable feature recognition in such cases.

Volumetric interference example. A common type of interaction affecting recognition systems is where one or more features volumetrically interfere with one another. This case is illustrated by Figure 7. The part shown has a rectangular pocket feature and four holes, so disposed that there is volumetric overlap between the canonical feature volumes involved.

Interaction among feature traces. As noted above, interaction among feature traces can render features unrecognizable. The clover-leaf-shaped pocket of the part in Figure 8 has several cylindrical surfaces which could be traces

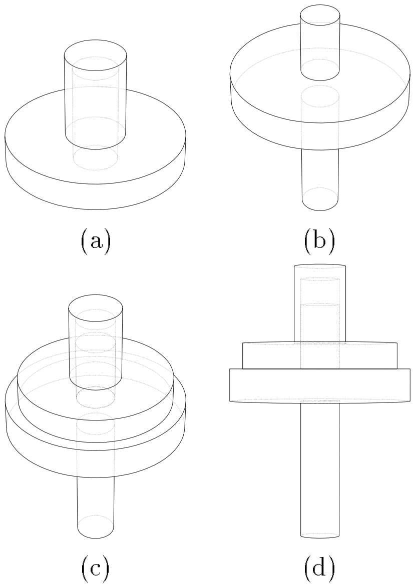


Figure 6. A simple assembly consisting of a cap (a) which fits over the tip of a spindle (b). Figures (c) and (d) show the completed assembly.

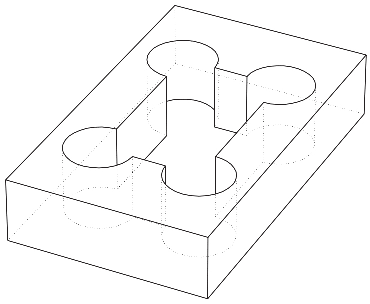
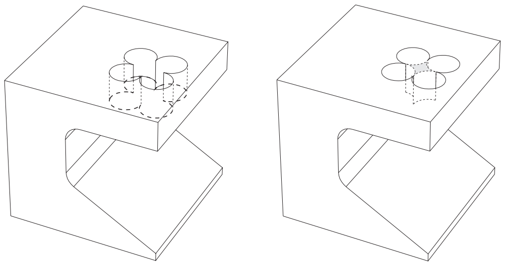


Figure 7. An example of volumetric interferences among machining process features.

for drilled holes. However, the holes alone do not describe all the necessary stock material to be removed, as seen in Figure 8 (b).

An approach to feature recognition that does not include a trace for recognizing pockets based on corner radii[3] would be unable to describe this part. A recognition system which includes feature construction algorithms for pockets based on curved wall faces or corner radii might avoid the

[3] When performing traditional machining, a rotating cutting tool leaves a corner radius when machining concave corners of the part. This implies that with traditional machining processes, one cannot create a part requiring the machining of sharp concave corners.



(a) (b)

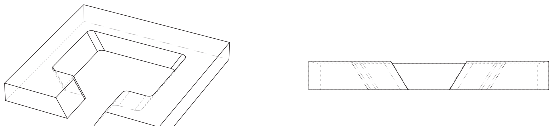
Figure 8. An example of volumetric feature interference resulting in the elimination of feature traces. Not that the clover-shaped pocket cannot be described in terms of the four drilled holes produce by the cylindrical surface hints, as they would leave a residue volume (shown in (b)).

problem presented by this particular configuration, but this approach will not necessarily generalize (one would need a trace for each degenerate configuration!).

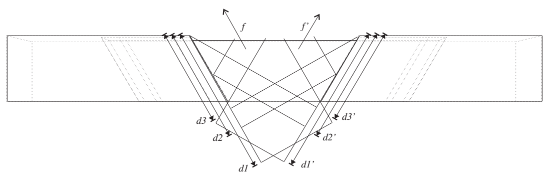
Interaction among feature parameters. As an example, consider the part shown in Figure 9(a). This part has an angled pocket and needs to be milled from a rectangular block of stock. As shown in Figure 9(b), because of the angle on the walls of the pocket, two operations are needed to create it. Therefore, we need to represent this pocket as two milling features f and f'. Any value of d between d_1 and d_3 can be selected as the depth of feature f; depending on d, there are many possible values for d', the depth of the feature f'. Since d' can have values between d'_1 and d'_3, there are infinitely many possible combinations of feature instances f and f'. Which pair of these feature instances are most appropriate depends on the available manufacturing resources (clearly the feature depth will depend upon the diameters of available cutting tools) and the optimization criteria. If this part had other features, they would also influence which of these possible parameterizations produces the most desirable feature instances f and f'.

Interactions at a Distance: Features Affecting Downstream Activities

Interactions can affect downstream activities and lifecycle considerations. We present an example of this type of interaction, where an interaction among machining features results in a problem in the process plan generated. Figure 10 shows a machined part for which a process plan has been created by the Technomatix PART^{tm} process planning system (originally developed at the University of Twente) [24, 2]; the resulting process plan has been tested



(a): upper right view (b): front view



(c): possible depths for features f and f'

Figure 9. A part with interacting pocket features that gives rise to infinitely many unique feature instances f and f' (arrows denote feature orientation). In this case, there are infinitely many choices for feature depths that will result in a pair of valid features capable of machining the part (from [18]).

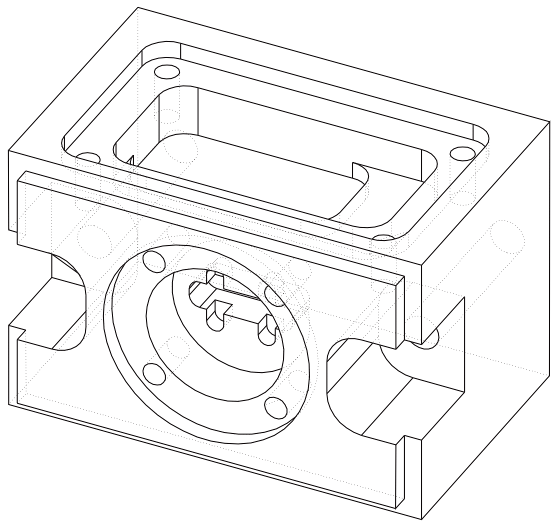


Figure 10. A part with a number of different types of feature interaction.

using Deneb Robotics' *Virtual NC*^{tm} simulation software.

PART^{tm} supports more than 30 feature types, defined in terms of relations between faces of a solid model. The system performs semi-automated (human assisted) feature recognition and generation of process plans for the machining of prismatic parts. PART^{tm} first makes an attempt to automatically identify features in the CAD model using surface-based recognition methods. The automatically gen-

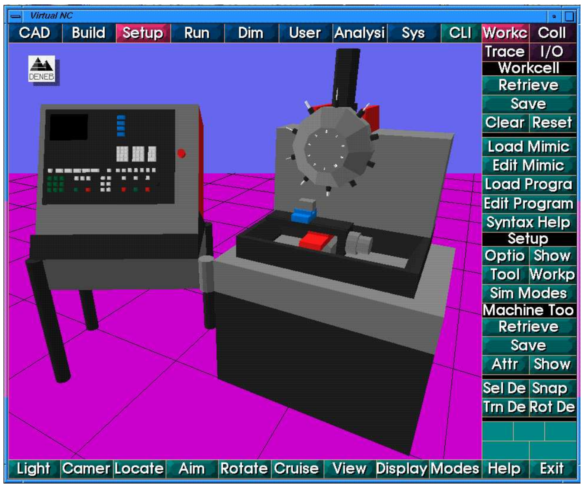


Figure 11. A screen shot of Deneb Robotics' *Virtual NC*^{tm} machining simulation software. Virtual NC is a physics and dynamics-based NC program verifier.

erated feature set serves as a starting point for the user, who then interacts with a feature editor to add, delete or modify features prior to setup planning, analysis of fixtures, and generation of a process plan with corresponding NC code.

In the Manufacturing Engineering Toolkit Project [11] at the National Institute of Standards and Technology, PART^{tm} has been integrated with Deneb Robotics' NC code simulation software *Virtual NC*^{tm}, shown in Figure 11. *Virtual NC*^{tm} executes the NC program produced by PART^{tm} in a 3D virtual environment complete with solid models of the stock material, machine and cutting tools, as well as the tool holder assembly. As the process plan is executed, machining operations are simulated and material is removed from the stock block in correspondence with what would actually occur in practice on the shop floor (Figure 12 (a)). However, as *Virtual NC* runs the process plan it detects a collision between the cutting tool holder assembly and the workpiece, shown in Figure 12 (b). This interaction between the hole feature, current workpiece, and tool holder results in a collision that renders the process plan invalid. This interaction lies outside the scope of the version of PART^{tm} used by NIST, and is only discovered in simulation—when the plan is considered in the context of the facilities and tools on the shop floor.

DISCUSSION AND CONCLUSIONS

This paper has illustrated that there are many different types of feature interaction. How interactions are handled is of vital importance in manufacturing planning. What is evident is that existing commercial and academic systems have only begun to address the broader interaction issue. In particular:

- Most commercial and academic systems have begun to address *interference interactions.* In particular, it has been widely noted that consideration of alternative manufacturing plans is vital in any automated planning system—and interference interactions often give rise to alternative feature-based representations (and thus alternate manufacturing plans) [7].
  Difficulties still remain in this category, however. Most systems still rely on ad hoc rules of thumb and heuristics to decide how to handle interference interactions. Further, interactions that interfere with feature recognition algorithms (such as those affecting hints and traces) have not been addressed. In these situations the common approach has been to assume that if the hint is gone, the feature is also gone; conversely if the hint is still there, the feature may or may not still be there. Similar problems exist for all automated feature recognition approaches (examples such as the one in Figure 1 can be devised for decomposition algorithms such that they cannot generate the primitives needed to find an optimal plan).
- Most researchers are just begining to deal effectively with interference interactions, and have so far given little attention to other types of feature interactions and their effect on planning. Interference interactions are important to feature recognition and the generation of alternative feature-based representations. However, *adjacency interactions* and *remote interactions* are far more complex and are of arguably greater importance in planning throughout the product life-cycle. In the existing research there is little in the way of explicit discussion in this category. A limited number of efforts have touched on this problem but to produce more general solutions will require an advance beyond geometry-only schemes [17].

Among the relevant issues beyond the scope of discussion in this current paper are:

1. *Feature recognition issues:*
   (a) **Graph-based interactions:** Graph-based approaches to feature recognition often rely on matching patterns in a graphical representation of a part to those of feature instances. Interactions can severely disrupt the canonical patterns associated with feature types making them difficult if not computationally impossible to identify with graph-based recognition methodologies.

Figure 12. **During simulation, Virtual NC^{tm} identifies problems in the machining process plan produced by the PART^{tm} system. In particular, there is a collision (shown in (b)) between the cutting tool holder assembly and the workpiece. This interaction between the hole and the workpiece can only be determined when the cutting tool, tool holder assembly, fixtures, and operation plan are considered in action on the machine tool.**

(b) **Shared face interactions:** When part faces belong to more than one feature, one must determine the extent of each feature's contribution to the face (contributions might not be exclusive and there may be non-trivial overlaps). In these cases one must determine if it is just a matter of defining feature boundaries appropriately or if more subtle criteria are needed.

(c) **Split face interactions:** Split faces [26] introduce many difficulties to automated feature recognition techniques. In particular, a face belonging to a single feature may be divided across several spatially disjoint regions of the part. These regions might be computationally expensive or even impossible to identify and correctly associate with the feature in question.

2. *Defining features for other domains:* To reason about products across different manufacturing processes and throughout their life-cycle we need new feature concepts and feature definitions. Research will have to advance beyond a machining-dominant mentality.

The discussion presented in this paper represents only an initial attempt at building a more general notion of "what are feature interactions." The authors hope that this contribution may stimulate a fruitful discussion, eventually leading to a common conceptual framework enabling an effective attack on the problems presented by feature interactions.

ACKNOWLEDGEMENTS

Any opinions, findings, and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the supporting government and commercial organizations.

Certain commercial software systems are identified in this document. Such identification does not imply recommendation or endorsement by NIST; nor does it imply that the products identified are necessarily the best available for the purpose.

REFERENCES

[1] David A. Bourne and Cheng hua Wang. Design and manufacturing of sheet metal parts: Using features to resolve manufacturability problems. In A. A. Busnaina, editor, *ASME Computers in Engineering Conference*, September 17-20, Boston, MA 1995, pages 745–753. ASME, New York, NY 10017.

[2] Control Data Corporation. *ICEM PART Reference Manual*, July 1994. Version 1.2.

[3] Xin Dong. *Geometric Feature Extraction for Computer-Aided Process Planning*. PhD thesis, Rensselaer Polytechnic Institute, Troy, NY, USA, 1988.

[4] R. Gadh and F. B. Prinz. Recognition of geometric forms using the differential depth filter. *Computer Aided Design*, 24(11):583–598, November 1992.

[5] Satyandra K. Gupta. *Automated Manufacturability Analysis of Machined Parts*. PhD thesis, The University of Maryland, College Park, MD, 1994.

[6] Satyandra K. Gupta, William C. Regli, and Dana S. Nau. Integrating DFM with CAD through design critiquing. *Concurrent Engineering: Research and Applications*, 2(2), 1994.

[7] Satyandra K. Gupta, William C. Regli, and Dana S. Nau. Manufacturing feature instances: Which ones to recognize? In Jaroslaw Rossignac, Joshua Turner, and George Allen, editors, *Third Symposium on Solid Modeling Foundations and CAD/CAM Applications*, May 17-19 1995, Salt Lake City, Utah, pages 141–152. ACM Press, New York, NY, USA. ACM SIGGRAPH and the IEEE Computer Society.

[8] JungHyun Han and Aristides A. G. Requicha. Incremental recognition of machining features. In Kosuke Ishii, editor, *ASME Computers in Engineering Conference*, September 1994, pages 143–150. ASME.

[9] JungHyun Han and Aristides A. G. Requicha. Integration of feature-based design and feature recognition. In A. A. Busnaina, editor, *ASME Computers in Engineering Conference*, September 17-20, Boston, MA 1995, pages 569–578, New York, NY 10017. ASME.

[10] Yong-Jung Huh and Sang-Gook Kim. A knowledge-based CAD system for concurrent product design in injection molding. *International Journal of Computer Integrated Manufacturing*, 4(4):209–218, 1991.

[11] M. Iuliano. Overview of the manufacturing engineering toolkit prototype. Technical Report NISTIR-5730, The National Institute of Standards and Technology, Gaithersburg, MD 20899, USA, October 1995.

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

[13] Timo Laakko and Martti Mäntylä. Feature modelling by incremental feature recognition. *Computer Aided Design*, 25(8):479–492, August 1993.

[14] S. Meeran and M. J. Pratt. Automated feature recognition from 2D drawings. *Computer Aided Design*, 25(1):7–17, January 1993.

[15] F. G. Mill, J. C. Naish, and J. C. Salmon. Design for machining with a simultaneous engineering workstation. *Computer Aided Design*, 26(7):521–527, 1994.

[16] J. Miguel Pinilla, Susan Finger, and Friedrich B. Prinz. Shape feature description using an augmented topology graph grammar. In *Proceedings NSF Engineering Design Research Conference*, pages 285–300. National Science Foundation, June 1989.

[17] M. J. Pratt. Application of feature recognition in the product life-cycle. *International Journal of Computer Integrated Manufacturing*, 6(1 & 2):13–19, 1993.

[18] William C. Regli. *Geometric Algorithms for Recognition of Features from Solid Models*. PhD thesis, The University of Maryland, College Park, MD, 1995.

[19] William C. Regli, Satyandra K. Gupta, and Dana S. Nau. Extracting alternative machining features: An algorithmic approach. *Research in Engineering Design*, 7(3):173–192, 1995.

[20] D. W. Rosen and J. R. Dixon. Languages for feature-based design and manufacturability evaluation. *Int. J. Systems Automation: Research and Applications*, 2(4):353–373, 1992.

[21] Scott A. Safier and Susan Finger. Parsing features in solid geometric models. In *European Conference on Artificial Intelligence*, 1990.

[22] J. J. Shah and D. W. Miller. A structure for integrating geometric tolerances with form features and geometric models. *ASME Computers in Engineering Conference*, American Society of Mechanical Engineers, New York, NY, 1989.

[23] J. J. Shah and M. T. Rogers. Expert form feature modelling shell. *Computer Aided Design*, 20(9):515–524, 1988.

[24] F. J. A. M. van Houten. *PART: A Computer Aided Process Planning System*. PhD thesis, University of Twente, 1991.

[25] J. H. Vandenbrande and A. A. G. Requicha. Spatial reasoning for the automatic recognition of machinable features in solid models. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 15(12):1269–1285, December 1993.

[26] Douglas L. Waco and Yong Se Kim. Geometric reasoning for machining features using convex decomposition. *Computer Aided Design*, 26(6):477–489, June 1994.

[27] P. R. Wilson and M. J. Pratt. A taxonomy of features for solid modeling. In M. J. Wozny, H. W. McLaughlin, and J. Encarnaçao, editors, *Geometric Modeling for CAD Applications*, pages 125 – 136. North-Holland Publishing Co., 1988.

[28] J. R. Woodwark. Some speculations on feature recognition. *Computer Aided Design*, 20(4):189–196, May 1988.