Proceedings of the 32nd International MATADOR Conference,
Editor A.A. Kochar, pp495-500,
UMIST, Manchester 10-11 July, 1997

STRUCTURED MULTI-LEVEL FEATURE INTERACTION IDENTIFICATION

M.S. HOUNSELL & K. CASE
Department of Manufacturing Engineering, Loughborough University
Loughborough, Leicestershire, LE11 3TU, UK

SUMMARY

Features are an established means of adding non-geometric information and extra geometric semantics to conventional CAD systems. It has been already realised that although feature-based modelling is necessary for the next generation of integrated design and manufacturing systems, inherent *feature interactions* pose a difficulty in representing and manipulating geometric design. This paper presents a structured multi-level geometric feature interaction classification scheme implemented within a Design-by Feature (DbF) system for representation validation analysis. Various feature interaction definitions and classification methods are first surveyed. The elements and the tests used for the identification process are presented. The classification encompasses existing feature interference cases found in the literature, uses a clear structure for the classification and, is applied at three different levels.

Keywords: Feature-based Modelling, Design by Features Systems, Feature Interaction, Feature-based Reasoning, Computer-Aided Design, CAD/CAM, Geometric Modelling, Concurrent Engineering.

INTRODUCTION

Design-by-features (**DbF**) is one approach for implementing Feature-based CAD systems and offers the designer a set of high level entities that can be used to describe and represent the desired component. Features are a means of incorporating knowledge of the form, behaviour, function and related manufacturing processes into a single representation.

The DbF paradigm differs from that of traditional CAD systems where low level entities such as lines, arcs, and circles are used, or more recent CAD systems based on a Geometric Solid Modelling (**GSM**) core where solid primitives of various shapes are provided and combined in a Boolean fashion (with unions, intersections, and differences).

DbF systems are also distinct from the Feature Recognition (**FeR**) approach where features are 'discovered' after a session using traditional or GSM CAD systems. DbF is claimed to be a more promising and efficient way of working.

In currently available DbF systems a lack of attention to formalising the concept of *feature interaction* can be seen even though this is a well-known, important and active issue of research. Feature interactions occur when features cannot be considered in isolation within the model because some influence is exerted among them that must be dealt with due to its meaning and importance (especially for engineering purposes). Interactions between features are at the heart of any Feature-based Modelling (**FBM**) environment because "they are directly and inevitably produced while manipulating the model" [2]. Besides, "intended interferences" are common practice in engineering and can be found for example in tolerances, assembly relationships or when assigning distribution patterns of features ([9], see Figure 1).

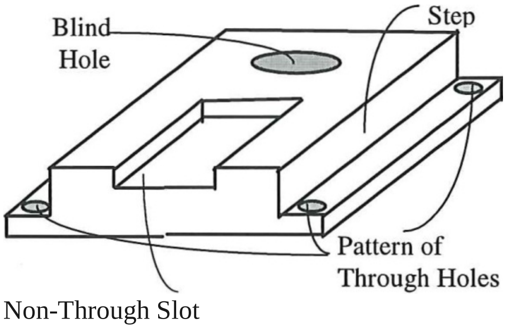


Figure 1: A pattern of four through fastening hole features.

Furthermore, feature interactions are the cause of some of the most serious problems in the development of generative computer-aided process planning (**CAPP**) systems, and "are important for determining process sequences and, sometimes the manufacturing processes themselves" [5].

This paper presents a thorough analysis of feature interaction for prismatic parts in the DbF domain that goes beyond already well explored *parent-child* relationships and geometric 'interference cases'. Existing interaction classification schemes are surveyed, the use of a common-sense definition is claimed and, a classification scheme which is also a geometric analysis structure based on Boolean operators that is applied at various levels on a consistent basis is presented. Finally, a discussion of the classification scheme's advantages and use is given.

The interaction classification herein presented establishes a high level feature interaction terminology and meaning. Related work will be referred to throughout this paper to emphasise that the classification encompass the classification scheme of others, but with a broader spectrum.

DEFINITIONS

Feature interaction means that a *mutual action or influence exists between features.* This definition emphasises that an interaction occurs when features cannot be considered isolated within the model and could happen for volumetric overlapping features as well as between non-overlapping and even non-contacting features.

Interference has sometimes been used to refer to interactions as a whole because it represents one (of many) very important analyses with a direct impact on manufacturing decisions. Some authors for example would claim that “interaction implies intersection between the entities of a feature” ([2], [4], [7], [8]). *Interferences* are special cases of interaction where destructive influences occur and possibly lead to a redundancy or loss of initial properties of a feature or its associated intents.

The terms *interrelation* and *relationship* have also been used to mean (special cases of) feature interaction.

RELATED WORK

Feature interaction is an active and important issue but has principally been explored by researchers who are involved with FeR systems, and is considered as a challenge (the number of features may be finite but features resulting from their interaction are infinite). A consequence is that “no general approach to recognise all interactions is yet known” [1]. DbF systems have been misconsidering (or at least misrepresenting) feature interaction so that only simple and straightforward interrelationships can be found in the literature.

This paper concentrates on feature-to-feature geometric interactions within the same “feature representation space” ([7]) and many other authors have concentrated on a sub-set of this domain with emphasis on their impact on manufacturing applications. Some of these interaction classification proposals include:

Zhang ([9]) classified *interferences* into two categories *(collision* and *cover)* that basically represent volumetric interaction between features and the stock material. Zhang also used a complementary set of criteria at the face level for checking the validity of an operation. He suggested that an interference can be valid in one application but invalid in others. This raises the necessity to identify as many feature interaction cases as possible (including *interferences),* while leaving the selection and binding process to a subsequent reasoning process that has information concerning the designer, the product, standards with which to comply, manufacturing processes, etc.

Bidarra ([2]) claimed to have encapsulated in each feature class definition the detection and reaction methods for the *interaction* phenomena which were classified as follows: *Topological,* the designer's intent is preserved and individuals feature parameter’s maintained, despite the overlapping of features (see Figure 2); *Transmutation,* the intended semantic behaviour of a feature is destroyed by feature manipulation (such as when the enlargement of a slot encroaches on an adjacent pocket and gives it the behaviour of a slot); *Geometric,* the feature’s geometry is affected without affecting its semantic behaviour (basically parameter-driven manipulations); *Closure* and *Absorption,* these occur when access to the feature is closed or the feature’s behaviour itself is absorbed by another feature.

The few existing feature interaction classifications reported, although possibly very efficient, do not comply with any comprehensive classification scheme, are oriented towards specific applications and are thus, biased and constrained by that domain.

Furthermore, “neighbouring” or "adjacency" ([6], [7]) of features has been considered to be of crucial importance for applications such as computing tool approach directions. However, these interactions are seldom represented in DbF systems because they are not considered to be *interferences* [1].

THE CLASSIFICATION FRAMEWORK

Entities and Levels

Form-features have a strong volumetric meaning and are concerned with the addition *(positive)* or removal *(negative)* of volumes (closed solids) which have been called Feature Produced Volumes (**FPV**, [6], [7]). FPV's can be the means by which features are analysed against each other to determine volumetric interactions. This aspect, called volumetric interaction (**VI**), must be part of a broader classification scheme that should be applied to, at least, three levels of interaction: volumetric (**VI**), boundary (**BI**) and facial (**FI**). It was found that various reasonings need to know the interaction between features (and their components) at all of these levels. Thus, similarly to FPVs, **FPB** is defined as the Feature's Produced Boundary and, **FPS** is the Feature's Produced Surfaces or faces.

The analyses consider a pair of *elements* at a time, called the *joint* **A** and **B**, from a specific *entity set* ($\sum$) with a *relative dimension* (**n**), denoted by (**A, B**) $\epsilon \sum n$. The classification is made according to the results of operations on the *joint.* Table 1 exemplifies entity sets at various levels with their *relative dimensions.* The *relative dimension* is a term used here only to clarify and to distinguish between entities with some respect to their complexity and dimensional representation but no mathematical meaning or relationship is used or implied.

Table 1: Entity sets and their examples

Entity Set	**Entity Example**	**Relative Dimension**
$\sum^5$	FPV (volume)	n=5
$\sum^4$	FPB (boundary)	n=4
$\sum^3$	FPS (face)	n=3
$\sum^2$	edges	n=2
$\sum^1$	vertices	n=l
$\sum^0$	NULL	n=0

Queries to the Underlying GSM

Two Boolean operators are used to make enquiries to the geometric solid modeller (**GSM**): Non-regularised Boolean intersection (usually represented as ∩) and, regularised Boolean intersection (represented as ∩*). Boolean intersection is commonly available in GSM such as CSG, B-Rep and, hybrid systems. These operators are used to obtain:

- **$C=A\cap^{*}B$.**
- **$D=A\cap B$.**

Other enquiries are *set membership* tests such as: "which feature does the face **F** belongs to?'", "is the entity **X** of the same type as entity **Y** ?", "what is the entity **W**? (a volume, face, edge, vertex)". Some of this information can be obtained from the FBM because it is usually kept in the FBM database as a reverse reference (pointers from the FBM towards some entities in the GSM data-structure).

The Classification Process

Thus, according to the result **C**, interacting entities can be classified into two types: Connected and, Disconnected.

- **Connected** interacting cases occur when **C is not NULL**. The word "connected" was chosen to emphasise that the connection between entities will only occur if an entity of the same *relative dimension* as the inputs is used to establish the relationship (and the same can be said of the regularised Boolean intersection). Connected entities can be distinguished between coincident *(conjoint)* or partially overlapping *(subjoint)* cases.
- **Disconnected** entities occur when **C is NULL** or, there is no relationship of the same *relative dimension* between **A** and **B**. Disconnected entities can be distinguished as separate *(disjoint)* and adjacent *(adjoint)* cases.

Conjoint connected cases are those where one entity is completely superimposed or inserted into another because the output of the Boolean operation is one of the original entities (**C = A** or **B**). Conjoint interaction occurs because the output coincides with one or both inputs. Con- joint cases can be further divided into:

- Cases where the inputs A and B exactly *match* each other (**C = A** and **C = B**, which means that **A** and **B** are the same).
- Where one entity is completely inserted into the other (**C = A** or **C = B** but, **A ≠ B** or simply, if they are **connected** <u>conjoint</u> but do not *match).*

Subjoint connected cases (the prefix "sub" when added to nouns refers to a thing, **C**, that is part of a larger one, the joint **A** and **B**, of the same *relative dimension),* also called *Overlapping* cases, occur when complex non-standard topologies arise. Such interaction could not affect the entity meaning itself but could have severe impact on downstream applications. For instance, if overlapping features (Figure 2) are not identified and represented properly they will result in redundant machining operations if they have the same volumetric removal intention. Subjoint connected cases can be sub- classified into:

- *Enter,* when one entity's end is completely inserted into another entity. An entity's end is of lower *relative dimension* than the entity itself. For instance, a feature's end is a face, in the same way as an edge's end is a vertex.
- *Cross,* when neither of the ends of an entity are inside the other at the same *relative dimension* (see Figure 2).
- A range of other cases that can be identified for pragmatic purposes is left here as a *General* sub-class for simplicity.

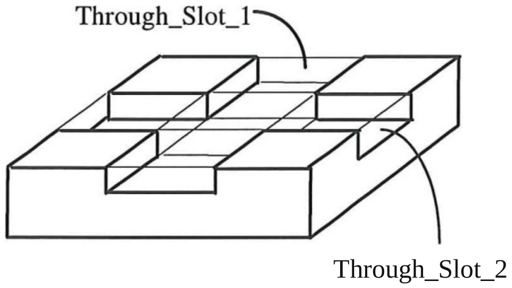


Figure 2: Cross subjoint connected VI case (Zhang's *collision,* Bidarra's *topological* interaction).

Disconnected interacting cases (represented as **disconnected**) occur when **C**, the Regularised Boolean Intersection result, **is NULL** and **D** is an entity of an inferior *relative dimension.* Two situations can occur: *adjoint* and *disjoint* disconnected interaction.

Disjoint interaction occurs where there is no intersection whatsoever, **C** and **D** are **NULL** and features are considered separate (the prefix "dis" is usually added to describe the opposite state of something). Disjoint cases can be:

- *Far* when entities are "really" distant from each other.
- *Near* when entities, although not touching, are close to each other and with no other entity in-between.

Conversely, *adjoint* (this word means next to each other, adjacent, touching) cases happen when **D** is **not NULL** and the input entities share a topological entity of lower *relative dimension*- the result **D** (Figure 3).

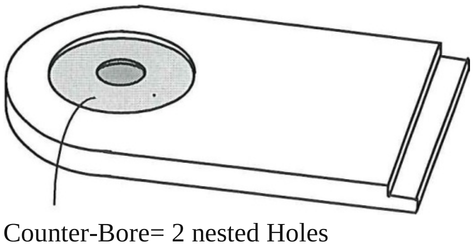
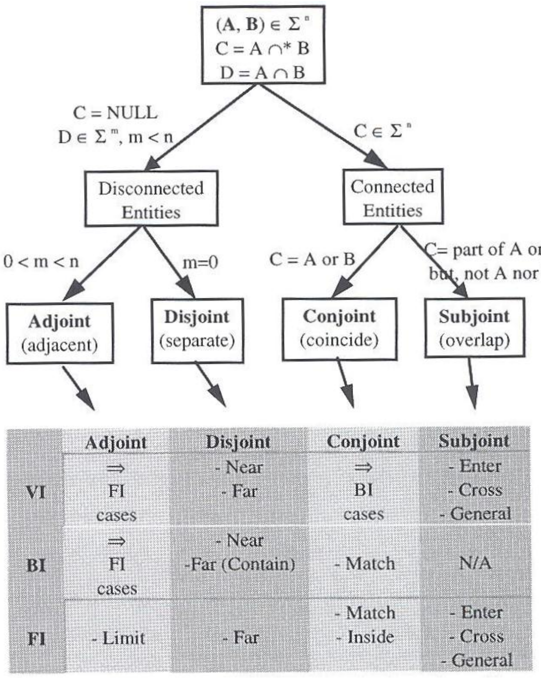


Figure 3: Adjoint disconnected VI (touch), inside conjoint FI case.

Disconnected VI cases can be further distinguished according to their "spatial inter-feature relationship" such as *parallelism* and, *coplanarity.*

The operations and *set membership* tests presented are reproduced in Figure 4 for clarity purposes. Each arc represents a test and each box represents either an operation or a status of the interaction. **A** and **B** are the joint entities, **C** and **D** the results of the operations and, **m** and **n** the *relative dimensions* of **D** and the inputs, respectively.



	Adjoint	Disjoint	Conjoint	Subjoint
VI	⇒ FI cases	- Near - Far	⇒ BI cases	- Enter - Cross - General
BI	⇒ FI cases	- Near -Far (Contain)	- Match	N/A
FI	- Limit	- Far	- Match - Inside	- Enter - Cross - General

Figure 4: The basic framework for classifying feature interactions.

The sub-cases most likely to occur at each leaf of the classification tree are presented in the table at the bottom of Figure 4. Some of them are links or pointers to a lower level of interaction. These are identified at the table by the symbol "→" and the interaction level. The arrows basically says that the classification can go deeper (if required) in order to distinguish between different cases that otherwise will be treated equally. Adjoint FI cases are identified as *limit* because they identify that one feature is actually being limited by another.

The interactions presented are not always commutative thus, the interaction relationship will have an *active* or *passive* response according to which input entity (**A** or **B**) was considered as a reference. Hence, interactions include *Crossing* or *Crossed*, *Inside* or *Outside*, *Limiting* or *Limited.* The exceptions are the commutative interactions: *Match*, *Near* and *Far.*

A "FRIEND" THAT UNDERSTANDS HOW FEATURES INTERACT

A DbF prototype system called **FRIEND** (an acronym for **F**eature-based **R**easoning for **I**ntent-driven **EN**gineering **D**esign) has been implemented with special attention being paid to representation validation of the feature-based models [3]. In order to carry out such validation processes a complete scenario of interactions is built identifying each interaction case between every single pair of features and, if required, its components (lower levels). The interaction case is of crucial importance because, together with the feature properties and parameters (rather than the GSM data), it is used as the "vocabulary" to express validity conditions and to verify the scenarios.

A first level of interaction scenario is analysed considering only FPV's as entities. At this stage, some reasonings can be already applied. If conjoint VI cases are encountered then FPB's are considered as the source for further classification. If adjoint VI or adjoint disconnected BI cases are encountered then, for practical reasons, these interaction cases are linked to many FI interaction relationships as required for each face of the feature's realisation but FI disjoint cases are discarded (because they are the most frequent ones and they do not add further information for the present reasoning).

Boundaries are considered to be closed sets of faces so, there is no way that two conjoint VI features would have a subjoint BI interaction (the intersection operation would return an open boundary) thus it is marked in the table of Figure 4 as Non-Applicable (N/A).

This scenario is then analysed by a knowledge-based system in search for compliance of the model with pre-defined general validity properties. If an invalid representation is detected the reasoning fires actions to revalidate the representation. **FRIEND** stores the interactions at each level in order to use this information in the reasoning. The classification structure gives crucial information to help the decision-making in these actions.

Using Feature Interaction

High levels of identified interaction act as filters or approximations for further low level reasonings and can be used promptly for some specific reasoning before lower level analyses are performed. Among adjoint VI cases there will be a possible merging operation (if a *matching* conjoint FI case happens) or a change on the feature's properties from "blind" to "through" (if an *inside* conjoint FI case occurs). Disjoint BI interactions, as an example of reasoning, means that one feature is contained within another and analysing their FPB will lead to a *near* or *far* case. The threshold between *near* and *far* should be computed by a separate "thin-wall reasoning". If *near* then it is possible that an "internal thin-wall" problem may have occurred and, if *far* (and if the feature happens to have no other interaction) it can be interpreted as a *hollow* in the part, which should be eliminated.

It can be inferred that besides helping to obtain more reliable feature-based representation for further "Design for X" analysis some manufacturability, assemblability, etc. analyses can be anticipated and performed at **FRIEND's** stage because the feature interaction classification is very expressive and powerful. For instance, various "thin-wall", obstruction, precedence and, accessibility problems can be easily detected through the proposed feature interaction identification scheme. Now phrases like "a slot *VI entering* a pocket" is a valid, meaningful and measurable statement in that a FBM system can process and produce.

Discussion

The feature interaction classification presented here has several advantages: (*i*) It is a DbF-aware scheme and subsumes existing classifications (both from FeR systems as well as from narrow DbF domains). (*ii*) It adds a comprehensive coverage and a clarification of the *interference* and *interaction* terms to avoid mis-understanding. *(iii)* It is multi-level which facilitates its integration with hybrid GSM modellers and allows reasoning to be performed at all these levels. (*iv*) All levels share the same structure and concept of classification promoting the consistency of the scheme. This also avoids misunderstandings because there is no mixing up of entities at each level. (*v*) Its categories are well defined through simple rules using commonly available GSM Boolean operators and tests. (*vi*) No concave/convex nor planar/non-planar assumption is made to the minor detriment of efficiency but, many of the operations and tests can be quickly and accurately predicted using Bounding Boxes. (*vii*) The cases are as more detailed as required allowing specific actions to be taken for apparently similar cases.

CONCLUSION

A feature-based CAD system has been implemented with special attention paid to "representation validation" of the feature model and is concerned with various aspects such as functionality, morphology, manufacturability, etc. The validation of a feature-based model is driven by all sorts of interactions between features (not only adjoint and overlapping, see [2]) thus, identifying interaction cases is essential to the forthcoming validation analysis to be applied. Besides, feature interactions identify many engineering-oriented tasks in a Design-by Feature system. The interaction framework presented here has been adopted as part of a "vocabulary" used to express the validation reasoning in a prototype system called **FRIEND** (an acronym for **F**eature-based **R**easoning for **I**ntent-driven **EN**gineering **D**esign). It is accurate (even using Bounding Box data), powerful (identifies complex cases), elegant (easy to understand), structured (has a well-defined structure that repeats itself), multi-level (works at volumetric, boundary and, face levels), simple (uses simple GSM-based operators and tests) and requires almost no knowledge of the intricacies of the GSM representation schemes (although some efficiency can be lost because of this). Although the methodology was implemented for convex and planar Bounding Boxes of features, for efficiency reasons, it is general in nature and can be applied to concave and non-planar geometries.

REFERENCES

1. V. ALLADA and S. AMAND (1995) Feature-based Modelling Approaches for Integrated Manufacturing: State-of-the-art survey and future directions. International Journal of Computer Integrated Manufacturing 8(6), p. 411-440.
2. R. BIDARRA and J. C. TEIXEIRA (1994) A Semantic Framework for Flexible Feature Validity Specification and Assessment. (ASME) Computers in Engineering Conference'94 1, p. 151-158.
3. M. S. HOUNSELL and K. CASE (1996) Representation Validation in Feature-based Modelling: A Framework for Design Correctness Analysis and Assurance. Proceedings of the 12th National Conference on Manufacturing Research, p. 256-261.
4. T. D. MARTINO and F. GIANNINI (1994) The role of feature recognition in future CAD systems. IFIP Int. Conf. on Feature Modelling and Recognition in Advanced CAD/CAM Systems 1, p. 343-355.
5. F. G. MILL, J. C. SALMON, and A. G. PEDLEY (1993) Representation problems in feature-based approaches to design and process planning. International Journal of Computer Integrated Manufacturing 6(1-2), p. 27-33.
6. M. J. PRATT (1988) Synthesis of an optimal approach to form feature modelling. (ASME) Computers in Engineering Conference'88 1 p. 263-274.
7. J. J. SHAH and M. MANTYLA (1995) Parametric and Feature-Based CAD/CAM: Concepts, Techniques and Applications. John Wiley & Sons Inc.
8. R. E. D. SILVA, K. L. WOOD, and J. J. BEARMAN (1990) Representing and Manipulating Interacting Interfeature Relationships in Engineering Design for Manufacture. (ASME) Advances in Design Automation DE-32-1, p. 1-8.
9. K. F. ZHANG and H. H. ELMARAGHY (1993) Validity Check for a Functional-oriented Modeler. (ASME) Advances in Design Automation DE-65-2, p. 293-300.