Development of a Generic Computer-Aided Process Planning Support System

C.F. Yuen[*], S.Y. Wong[#], and Patri K. Venuvinod[#]
* Hong Kong Institute of Vocational Education (Chai Wan)
# City University of Hong Kong

Abstract

Although there is significant industrial need for comprehensive computer-aided process planning systems (CAPP), most traditional CAPP solutions have been fragmented in nature. This is because each CAPP domain (assembly, machining, inspection, etc.) has been treated independently. This paper argues in favor of adopting 'feature-orientation' as the unifying theme and describes a Generic CAPP Support System along with the Geometric Feature Recognition algorithms involved. Finally, some case studies derived from diverse application domains are presented to illustrate the advantages provided by the approach.

Key words: Feature recognition, Computer-aided process planning, Case-based reasoning

1. Introduction

Computer-aided process planning (CAPP) has long been recognized to be an important component and enabler of Concurrent Engineering (CE). CE is facilitated by the availability of a computerized system for estimating production costs and times that, necessarily, includes a comprehensive and robust CAPP system.

An effective CAPP system needs to be comprehensive because the production of a product (and, even, a single part) often needs a large variety of processes. Each of these processes may be a potential participant in the production process. These have to be individually evaluated in the context of the particular design specification of the part/product under consideration. A major hurdle faced by developers of a comprehensive CAPP system is that the processes it has to address have usually very little in common. The nature of each process and, hence, the knowledge base needed in its evaluation and design are quite distinct. As a result, the very subject area of CAPP has tended to be highly fragmented where each process domain is tackled in an independent and parallel fashion (see Figure 1). Often, the resulting overall system is quite restricted in its scope and, even within this limited scope, there is considerable redundancy. In short, very little progress has so far been made with respect to meta-reasoning concerning the basic nature of process planning itself in terms data utilization and data processing methods so that one could hope to develop a comprehensive but more coherent and less redundant CAPP system. The present paper offers a solution that goes some way towards redressing this situation.

The proposed strategy is based on the following premise. The creation of any domain specific process plan typically involves two interacting thought processes: extracting relevant high level information from the part/product needs to be collected, and reasoning over it on the basis of the corresponding domain-specific knowledge-bases (DSKB). Of these, the latter are likely to have very little in common. Hence the concept of DSKB cannot be the key to the desired 'seamless' integration of CAPP. In contrast, *every* CAPP sub-module involves reasoning over the part/product specification (specifications of form, dimensions, tolerances, surface roughness, etc.). Hence, if one wishes to develop a CAPP strategy that is applicable to (almost) every domain-specific CAPP sub-system, the strategy used must recognize and exploit part/product information as the unifying theme.

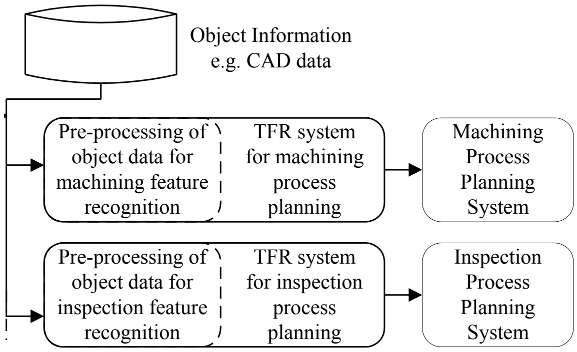


Figure 1. Traditional approach to the development of a comprehensive CAPP

An implication of the premise described above is that there *could* be a common 'front end' to every domain-specific CAPP sub-module that

- requires only the specification of the part/product as its input, and
- does not address issues requiring the use of a DSKB.

We will refer to this 'front end' as the *Generic CAPP Support System* (GCAPPSS).

This support system can be expected to act as the common starting platform for diverse domain-specific CAPP systems to be developed subsequently–thus reducing the overall effort needed in achieving a comprehensive CAPP system. The principal intent of the present paper is to argue in favor of the desirability, feasibility and utility of the concept of GCAPPSS. The rest of the paper is structured as follows. Firstly, arguments will be presented in favor of separating technological feature recognition (TFR)–a process implicit in any domain specific CAPP–from geometric feature recognition (GFR) and making the latter as the 'front end' of GCAPPSS. Next, certain complexities associated with technological as well as geometric feature recognition will be highlighted. This will be followed by a brief introduction of several GFR related algorithms developed by the authors.

---

[*] Dept of Manufacturing Engineering, HKIVE (Chai Wan), Shing Tai Road, Chai Wan, Hong Kong.

[#] Dept of Manufacturing Engineering and Engineering Management, City University of Hong Kong, Tat Chee Avenue, Hong Kong.

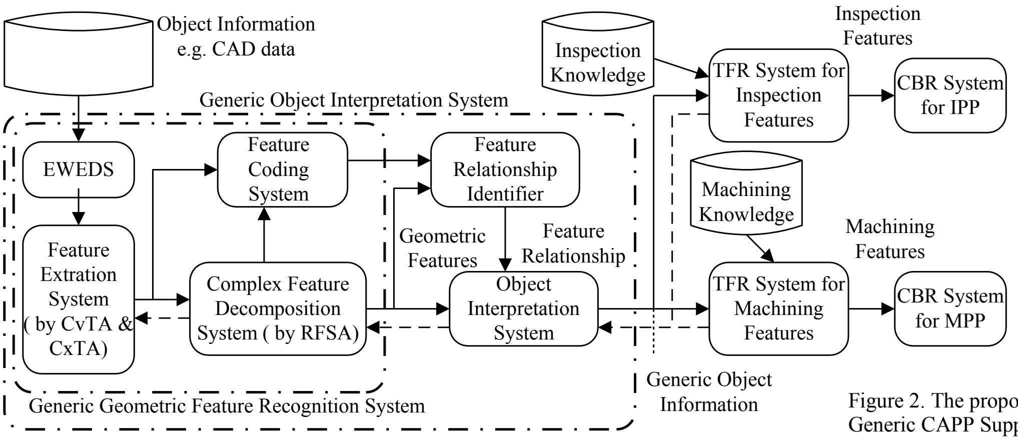


Figure 2. The proposed structure of the Generic CAPP Support System

These algorithms are not only capable of extracting and recognizing geometric features, but also decomposing complex features into simpler ones to facilitate the identification of all possible feature relationships. The information so gathered is then reorganized to yield a multi-layered part representation that facilitates multiple interpretations of the same part from different CAPP viewpoints. Finally, the implications of GCAPPSS with respect to downstream TFR and CAPP processes will be highlighted.

2. Geometric Feature Recognition: The 'Front End' of CAPP

Usually, when a process planner initiates a planning exercise, the first thing that (s)he examines is the part to be manufactured. This is typically done with reference to a visual image of the part and never the raw data of the part's computerized model. The planner then abstracts high-level information from the image. In particular, the technological features are first extracted. For instance, a machining process planner might be interested in recognizing cylindrical holes, if any, existing in the part so that (s)he can plan the corresponding drilling and reaming operations. These technological features form a subset of the part model that are of interest in the context of the specific process-planning domain. Hence, TFR cannot be used as the basis for GCAPPSS.

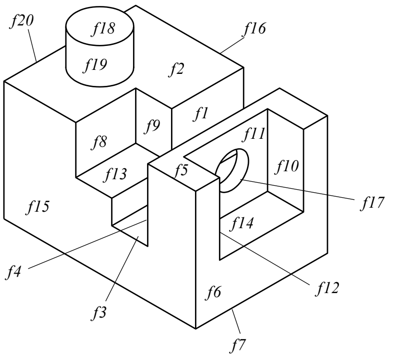


Figure 3 Example of a part with several features

Although technological features are domain specific, almost all of them seem to be rooted in certain geometric features implicit in the geometry of a part. For instance, while one is deciding whether there is a need to invoke drilling cycles (an activity of machining process planning (MPP)), or planning the measurement of a finished hole's diameter (an activity of inspection process planning (IPP)), the focus is on recognizing cylindrical depressions (essentially a geometric concept). Thus, geometric feature recognition (GFR) is the 'front end' of almost every domain specific TFR. Further, the data needed for GFR are derivable exclusively from the CAD model of the part/product, i.e., GFR does not need the invocation of extensive domain specific knowledge. Hence, it makes sense to build GCAPPSS around geometric feature recognition as illustrated in Figure 2.

3. On the Complexity of Part Feature Recognition

Whether it is technological or geometric, feature recognition itself is a complex and infinite domain problem. Given a set of simple features, there exists the possibility of their interacting in myriad ways to produce an infinite variety of features. Thus, the robustness of a given feature recognition system critically depends on how the system is structured with respect to the problems of complexity and infinite variety of features.

The concept of technological features is meaningful if and only if the specific application domain is well defined. This is one reason why a part can have multiple interpretations in terms of features. Likewise, an unambiguously extracted feature itself can have multiple interpretations. The latter situation arises mainly for two reasons: geometric and technological. The possibility of multiple interpretations of a part feature can be illustrated with reference to the part shown in Figure 3. An example of the variability in geometric interpretation is a part feature of the form of a 'staircase'. Do faces *f1, f3, f4*, *f8*, *f9* and *f13* form a single depression feature? Or, is it made up of a corner pocket that is adjacent to a slot? Even if these ambiguities are resolved, further variability in feature interpretation is possible in the downstream stage of TFR. Since geometric feature recognition is the core process to be captured by the GCAPPSS, we will start with the problem of coping with the complexity and infinity of geometric feature recognition. Some concepts outlined in this paper have already been reported in [1-5].

4. The Taxonomy and Morphology of Polyhedral/ Cylindrical Features

The basic issues concerning the taxonomy of geometric features can be illustrated by viewing the part shown in Figure 3 as a collection of *faces, edges* and *vertices*. Note that faces *f17* and *f19* are *cylindrical* whereas the rest are *flat*. Note also that the edge resulting from the intersection between faces *f19* and *f2* is *concave* whereas that between *f19* and *f18* is *convex*. Finally, all vertices are formed through the intersection of three edges. The part shown in Figure 3 consists of one cylindrical *protrusion* feature (view faces *f2*, *f18*, and *f19* together). This feature can be called a *primitive* feature since it does not appear to be possible to decompose it into a set of simpler features. The part also consists of what appear initially to be three distinct *depression* type features. The first two of these are the open cylindrical *hole* (zoom in the vicinity of face *f17*) and the *edge-pocket* composed of faces *f5*, *f6*, *f10*, *f11*, *f12*, and *f14*. Evidently, these two features are primitive in nature. The third immediately evident depression feature is that formed by faces *f1*, *f2*, *f3*, *f4*, *f5*, *f8*, *f9*, *f13*, *f15*, and *f16*. However, this feature cannot be a primitive feature since it seems to be possible to decompose it into a *slot* (view faces *f1*, *f3* and *f4* together) and a *corner-pocket* (see faces *f8*, *f9* and *f13*). Such decomposable features may be called *complex* features.

Consider now the morphology of a geometric feature irrespective of whether it is primitive or complex. A high level of interpretation of a feature will have to consider the relationships amongst the faces and edges of the feature. From this viewpoint, a named geometric feature may be defined as '*that portion of an object in which a group of faces and edges possesses a specified set of attributes (often expressed as a set of dimensional and orientation relationships) such that the portion can be distinguished from other portions of an object'*. This suggests that a part face can have different meanings with respect to different features. For instance, it is the cylindrical nature of face *f17* that points to the existence of the cylindrical hole in the part illustrated in Figure 3. Such a face may be called a *root face*. In contrast, face *f17* merely bounds the edge pocket. Hence it may be called a *boundary face*. Traditional GFR strategies had included only the root faces in the definition of a geometric feature. This had meant that one could recognize a named feature but not how it is related to other features belonging to the same part. This is a serious limitation from the point of view of TFR.

For instance, while inspecting the depth of a slot, one would need to utilize the boundary face as the datum face and the base face (a root face) as the face to be probed. This problem is resolved in the present paper by including both root and boundary faces within the definition of every geometric feature. The inclusion of root and boundary faces leads to three possible types of interactions between a pair of features: BB, RB, and RR interactions as explained in Figure 4. The identification of these interactions is important in TFR.

5. Geometric Feature Recognition System: The Core of GCAPPSS

Our approach to geometric feature recognition assumes that a reasonably complete CAD file of the part using a Boundary Representation (BRep) standard is available. In general, the data contained in such a file can be divided into two parts: coarse and fine. *Coarse data* pertains to information such as adjacencies between faces and concavity or convexity of edges. Much of the reasoning involved in GFR is carried out mainly over coarse data. Entities such as dimensions, tolerances, and surface roughness values belong to the class of *fine data*. These data gain particular significance during downstream TFR/CAPP processes.

The coarse data contained in the CAD file are first processed to explicitly identify basic information concerning the faces, edges and vertices. The data structure adopted at this stage is the Extended Winged Edge Data Structure (EWEDS) described in [1, 2]. One could also include other information needed for identifying the nature of interaction and the connection face between a pair of features. The next stage is to extract all possible geometric features implicit in any given EWEDS file.

Among the early GFR systems attracting widespread attention were those utilizing expert system (e.g., [6]) where a production rule had to be written for recognizing each defined feature in advance. Obviously, this bottom-up approach was incapable of coping with the possibility of infinite variety of geometric features. Further, owing to the absence of structured guidance for rule writing, the process of writing the rules itself became an 'expert' task. Subsequently, Joshi initiated interest in graph theory based GFR strategies [7] for recognizing polyhedral features. In particular, he developed the concept of Attributed Adjacency Graph. In fact, Joshi's exploitation was mimicking the view of cognitive psychologists [8] that human cognition of an object involves partitioning it in the vicinity of concave edges. We extended Joshi's approach to permit the recognition of polyhedral as well as cylindrical features by developing the concept of the Multi-Attributed Adjacency Graph (MAAG). Once a feature MAAG was defined, it was a trivial step to write the production rule for recognizing the feature. However, the problem arising from writing a new recognition rule for each new feature remained unresolved. We therefore abandoned the expert system strategy and attempted to solve the problem of coping with the complexity and infinity of GFR purely by algorithmic means.

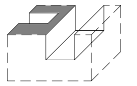
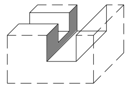
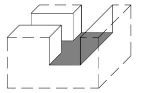
BB Interaction - The two features have a common boundary face(s)	
RB Interaction - The boundary face of a feature is the root face of the other	
RR Interaction - The two features have a common root face	

Figure 4. Possible types of interactions between features

The first issue we addressed was the extraction of complex features (i.e., of features that could be decomposed into a set of simpler but adjacent features) and those that are primitive (i.e., those that are not complex). This goal was found to be achievable through repeated application of Concave Triggering Algorithm (CvTA) that could extract *every* depression type of feature and a complementary Convex Triggering Algorithm (CxTA) that could extract *every* protrusion type of feature.

These algorithms enable our GFR system to extract each geometric feature separately before proceeding to its recognition. This means that, instead of anticipating, our system *discovers* each new feature as and when the system encountered it in practice. Once a feature has been discovered and labeled, the system stores it away in a feature library thus permitting continuous extension of the feature recognition range of the system. This approach also helps reduce the volume of data required for identifying each individual feature. However, this approach, by itself, was unable to facilitate multiple interpretations of complex features. To solve this problem, we proceeded to develop the concept of *primitive template feature* (PTF).

A PTF is a primitive feature (one that cannot be meaningfully decomposed into a set of simpler features) with a standardized code (the template) tagged on to it. Thus, a PTF is analogous to an atom. Some atoms can be varied to form isotopes. Atoms can interact with other atoms in myriad ways to form elements and compounds. Conversely, one can obtain various levels of insight into the nature of a given compound by decomposing it at various levels into its constituents. In a similar fashion, a PTF can be varied to form a *Variation of a PTF* (VPTF). A finite set of PTFs or VPTFs can mutually interact to form a seemingly infinite set of complex features. Conversely, a previously extracted complex feature can be subjected to multiple interpretations by decomposing it at various levels. The authors have compiled an apparently complete set of PTFs applicable to any part with polyhedral and cylindrical features. Next, we developed a formal system for encoding the template of a given polyhedral or cylindrical feature in a compact manner. The coding system is illustrated in Figure 5. This coding system has enabled the development of a 'Coding Algebra' that facilitates the analysis of feature interactions in a formalized manner. Further, it has been found to be useful in keeping track of how the geometry of a part would change as it is being progressively machined. This observation points to one advantage derivable by including an advanced GFR module at the core of the GCAPPSS.

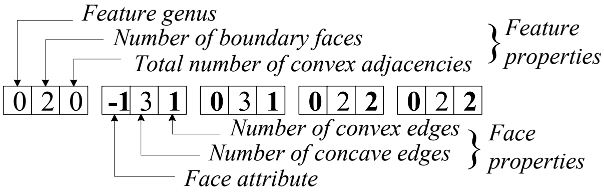


Figure 5. The coding system used in coding feature templates

Another noteworthy feature of our GFR system is the inclusion of a Root Face Segmentation Algorithm (RFSA) that is capable of facilitating the decomposition of a complex feature into simpler features as well as regrouping them in different ways to permit multiple interpretations. This flexibility was achieved by an eclectic use of different root face segmentation heuristics such as the Shortest Concave Distance Rule, the Edge for Same Parametric Plane Rule and the Normal Projected Plane Rule.

6. Feature Relationships and Part Representation

The previous section has described how the generic GFR system extracts the features implicit in the part model, recognizes them, and decomposes them if necessary. The role of the Feature Relationship Identifier (FRI) is to list the possible geometric relationships between each pair of previously extracted and recognized features. In particular, FRI examines whether each interaction is of the BB, RB, or RR type. Besides, FRI also identifies the common boundary and/or root face(s) between a pair of interacting features.

As with many exercises involving human reasoning, the downstream processes such as TFR and the execution of CAPP itself can be expected to occur in three stages. The first stage consists of breaking down the given problem into a set of sub-problems that can be tackled with greater ease. In the second stage, each sub-problem is individually solved in an appropriate manner with the aid of the relevant domain specific knowledge base. The third stage entails the integration of these piecewise solutions to develop the global solution to the complete problem. Each of these stages may require multiple iteration. The final solution may not be unique. Hence there is a need to represent part information in a systematic but flexible manner. In GCAPPSS, this is done by combining the outputs from the Feature Relationship Identifier and the Generic GFR System and reorganizing the resulting data in a multi-layered manner. Figure 7 illustrates our multi-layered representation of the part shown in Figure 6. In the highest layer, the part is represented as a Multi-attributed Feature Relationship Graph where each large/shaded node represents a feature extracted by the triggering algorithms. This layer provides an overview of the part in terms of its geometric features. Each node in the highest layer is associated with a set of attributes indicating the feature class, the feature code, and the lists of root and boundary faces. The arcs in the highest layer carry information concerning the relationship between the two end-nodes (features). Different types of arcs are used to represent different types of interactions between a pair of features. The arcs have a smaller node called 'connecting face node' that records the properties (including fine data such as the dimensions) of the connecting faces. However, in order to reduce the visual complexity of the graph, common connecting face nodes are merged. For instance, note that face *f24* is the boundary face of all the holes in the part in Figure 6, i.e., they exhibit a BB-interaction. Therefore, a single connecting face node (*f24*) is used to represent all BB-interactions. Likewise, a single connecting face node (*f10*) is used to represent RB-interactions of the complex protrusion feature (*F1*) with the four holes at the corners.

To simplify the representation of types of interaction, the arcs consist of two segments joined together through a connecting node. This node stores information concerning the interacting faces (the face number, attributes, dimensions, tolerances, etc.) so as to facilitate downstream TFR/CAPP processes. Each segment of the arc is a simple line or one carrying an arrowhead. A simple arc segment indicates that the face is a boundary face of the feature while one carrying an arrowhead indicates that it is a root face of the feature.

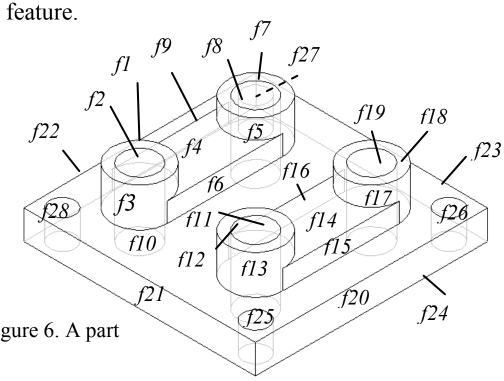


Figure 6. A part

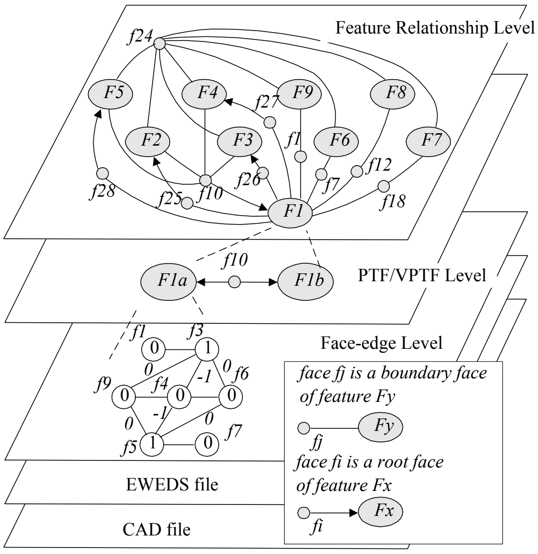


Figure 7 The multi-layered representation of the part shown in Figure 6

The second layer contains the feature MAAGs corresponding to each decomposable feature in the first layer. Since this layer is designed for further representation of a decomposable feature in the first layer, pointers are established to the appropriate features in the first layer. The nodes in this layer represent the PTFs or VPTFs that have been identified by using appropriate decomposition methods for the complex features. Again, these nodes contain information related to the decomposed features such as feature class, feature code and face lists etc., whereas the arc attributes include the facts of RR-interaction and the corresponding interacting faces.

The third layer contains the MAAGs corresponding to each of the features in layer 1 and 2. The nodal attributes include pointers to the face attributes listed in the EWEDS (see the fourth layer). Likewise, the edge attributes include pointers to the edge properties listed in the EWEDS.

7. Geometric Object Interpretation

GCAPPSS contains a Geometric Object Interpreter (GOI) that reasons over information contained at the highest level of the multi-layered part representation so as to identify commonalties amongst features. This may require the extraction of the boundary faces of each feature and then reasoning over them so as to identify feature groups having a common boundary face. Likewise, the system can identify feature groups that have the same feature class, geometric dimensions, orientation, etc. Identification of such groups is likely to be of interest in many domain specific TFR/CAPP stages. While generating the process plans, these steps facilitate the subsequent selection of proper instruments or tools as well as the design of the appropriate sequence of operational steps through retrieval from a finite database.

From the viewpoint of a specific TFR/CAPP domain, it is possible that the desired first and second layer representations of the part shown in Figure 6 might deviate (possibly significantly) from those shown in Figure 7. This is because the representation in Figure 7 has been derived from a purely geometric analysis. However, the GFR-based representation would normally be containing sufficient information for abstracting the representation desired by the TFR/CAPP stage. Therefore, all that the GOI needs to do is to effect this abstraction. This suggests that it is useful to have feedback links that propagate the desired configurations of the TFR to the GOI and then to the feature relationship identifier and the RFSA (see Figure 2). Further research is required to implement this feedback. However, it should be noted that the layers between the face-edge level and the CAD-file-level are all unique to a given part—they are not influenced by downstream TFR/CAPP processes.

8. Discussion

Consider how GCAPPSS might be useful in supporting different downstream TFR activities, we will start with a TFR concern of product designers as an example before proceeding to some examples relevant to machining and inspection process planning. Suppose a designer is interested in recognizing whether a given shaft (e.g., the shaft shown in Figure 8) is a splined shaft. This is the same as recognizing whether there exists a ‘multi-splined cylindrical feature’ on the part. This presupposes that the designer has already defined this particular type of technological (design or functional) feature. A definition that is likely to be appealing to the designer is that such a feature is characterized by ‘a series of slots that are regularly distributed over a convex cylindrical part feature and oriented parallel to the axis of the cylindrical feature’. Clearly, all characteristics embedded in this definition are retrievable from our multi-layer representation. For instance, there would be a countable set of nodes representing the ‘slots’ and a ‘connecting face nodes’ linking them. The latter node would be carrying the attribute that it is ‘convex cylindrical’ (here we may have to descend to layer 2). A simple rule or algorithm could then be written to capture this particular reasoning.

Consider now the concerns of machining process planning (MPP) of the same splined feature. If the designer had used a feature-based design system, the existence of a ‘multi-splined cylindrical feature’ is likely to have already been known. Otherwise, as in the case of the design oriented TFR, the machining-CAPP system would have to invoke the procedure for recognizing the splined feature. However, after such invocation, the CAPP system would need to obtain the dimensions of the ‘slot’ by descending down the multi-layered data structure to locate the required dimensions so that it can plan the necessary ‘slot’-milling cycles.

It is not surprising that GCAPPSS appears to be capable of supporting the domain of machining-CAPP. This is because, as the structures of many current CAPP systems directed specifically at the machining domain seem to indicate, the process of ‘machining’ is inherently feature-oriented and hence the availability of GCAPPSS is likely to be beneficial. A similar situation seems to exist with respect to many other CAPP domains. For instance, the planning of assembly, molding as well as fixturing seems to start with a high level interpretation of the part being addressed. Could we generalize the observations by stating that *all* CAPP domains could be ‘feature-oriented’ so that GCAPPSS could emerge as the common ‘front end’ of a future comprehensive CAPP system?

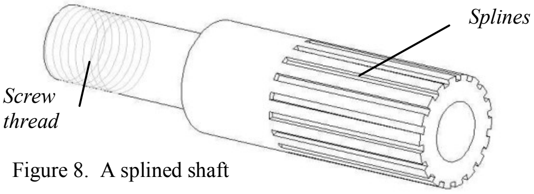


Figure 8. A splined shaft

Figure 7. A splined shaft

A potential problem with the above generalization relates to the possible existence of a CAPP domain that does *not start* with considerations related predominantly to part features. The resolution of this issue would require us to identify a CAPP domain that either has not been addressed in depth so far or appears (intuitively) to be not naturally feature oriented. The domain of Inspection Process Planning seems to meet this criterion. Firstly, a review of literature on CAPP reveals that, notwithstanding its great industrial importance, the domain of inspection using common metrological instruments has received little attention. (There is considerable literature related to inspection using a coordinate measuring machine.)

Secondly, when the authors interviewed a highly experienced inspection planner, it was not immediately apparent that his planning exercises were predominantly feature oriented. (However, when probed further, the planner admitted that it was not a bad idea to develop a feature-oriented inspection planning system.) Hence, it is worthwhile to briefly discuss the potential utility of GCAPPSS to inspection planning domain. Returning to Figure 8, consider how the splined feature could be inspected. Since the feature has already been recognized and the associated dimensions have already been extracted (as explained earlier), all that remains to be done is to extract the tolerances. Having abstracted all the necessary data, the inspection planning system can takeover the rest of the planning process.

Consider yet another example drawn from the domain of inspection. Suppose the inspection planning system is seeking to find the most suitable datum face for inspecting the majority of the dimensions specified for the part shown in Figure 6. This is easily done by reasoning over the feature relation graph (see layer 1 of Figure 6). This graph immediately reveals that face *f24* has the largest number of links as a boundary face to all other faces. One may therefore conclude that *f24* is the most suitable datum face while inspecting the part. Further, from the feature relationship graph of the object model, it is also possible to identify that the eight features are of ‘hole type’. Such information is easily extracted by the GOI through checking of the class of each feature. Further checking on the associated fine data may find that these ‘holes’ can be divided into several sub-groups in terms of hole dimension and hole orientation. The identification of such groups should be helpful in developing an efficient inspection plan.

At this stage, it is worth highlighting one general observation concerning the difference between machining planning and inspection planning. Machining seems to be in greater need of aggregate information (particularly at the complex feature level) since a canned machining cycle can realize several sub-features and the associated dimensions and tolerances in one go. Thus, much of the reasoning needed for machining can be conducted at the highest levels of part representation. In contrast, inspection often utilizes single point probing. Therefore, its planning requires reasoning that requires one to traverse from the highest layers to the lowest layers (fine data such as dimensions and tolerances are recorded at the lower levels). These observations indicate that it is advantageous to have a multi-layered data structure representing the part before one proceeds to the stage of CAPP. In fact, GCAPPSS was formulated with this consideration in mind.

The above discussion has argued in favor of adopting a feature-oriented approach while developing a comprehensive CAPP system. The discussion on GCAPPSS shows that such an approach is feasible at least to the extent of facilitating TFR. However, there is much to CAPP beyond mere TFR. How would the paradigm of feature-oriented CAPP affect each domain specific CAPP? For instance, what would be the effect, if any, on the way the domain specific knowledge base needs to be implemented? Answering these questions fully is beyond the scope of the present paper. However, one way of finding the answers is to engage in the development of a domain specific process planning system that maximizes the benefits derivable from the paradigm of feature-orientation. Our work so far suggests that it would be advantageous to utilize Case Based Reasoning (CBR).

CBR is a knowledge-based technique that has gained prominence in many fields (including the field of design) in recent years. Schank [9] describes the nature of CBR as follows: “*Most people prefer not to have to think hard if they can help it. They will try to get by with whatever worked before, even if it is less than optimal. We believe that, roughly speaking, people’s everyday cognition consists of abut 90% retrieving of past solutions and only about 10% or less of actual novel problem solving. Because of our belief about the relative importance of retrieval, it follows that if one wants to understand what it makes to model human intelligence, one should focus on the type of processing that contributes the most to people's everyday behavior, namely retrieval and adaptations of old solutions*.”

The implementation of any CBR system requires one to define the basis on which the cases in the case library would be unambiguously delimited. Here, feature-orientation could be of great value. For instance, in the case of inspection planning, one could initially identify the most frequent set of features that are traditionally in the specific inspection domain and the GCAPPSS could be configured to extract this set. Next, previously developed inspection plans for each of these inspection features could be stored in a case library. When a new part arrives, it could be sent to GCAPPSS to create the multi-layered part representation. Each part feature so identified at the highest layer could then be matched against the feature cases in the feature library. If a good match is found, the inspection plan associated with the specific library feature could be retrieved and adapted (i.e., varied to match the requirement of the current feature). If no match is found, the current feature could be subjected to manual inspection planning. The new inspection plan could be stored in the feature case library as a new case. This approach will enable the CAPP system to learn from experience.

9. Conclusion

A major hurdle facing the development of a comprehensive CAPP system is that each CAPP domain is quite unique in terms of the analytical models and knowledge bases it utilizes. As a result, the field of CAPP has become highly fragmented. A unifying theme is needed to reverse this trend. One way of achieving this goal is to adopt the paradigm of feature-orientation as the unifying theme. This paper has explored one method of implementing this paradigm. The method involves the creation of a Generic CAPP Support System that acts as the ‘front end’ for all domain specific CAPP systems. The presence of such a ‘front end’ is helpful in avoiding unnecessary downstream redundancies. GCAPPSS itself consists of a generic Geometric Feature Recognizer, a Feature Relationship Identifier, and a Geometric Object Interpreter. The first unit invokes a powerful set of algorithms that enable feature extraction, recognition, coding, classification and decomposition. The output from this system enables a multi-layered hierarchical part representation that seems to facilitate the interpretation of feature relations and the object itself. The system facilitates downstream technological feature recognition and CAPP exercises irrespective of the process domain. The paradigm of feature orientation deserves further research. In particular, its effects on the knowledge structures used downstream need to be examined. Initial efforts in this direction have indicated that there exists a natural synergy between the paradigm of feature-orientation and case based reasoning.

References

[1] Wong S.Y., Automated Geometric Feature Recognition - An Essential Component for Automated Design for Assembly, M Phil thesis, City Polytechnic of Hong Kong, 1992.

[2] Venuvinod P.K. & Wong S.Y., “A Graph-based Expert System Approach to Geometric Feature Recognition”, *Journal of Intelligent Manufacturing*, 1995, Vol. 6, p155-162.

[3] Venuvinod P.K. & Yuen C.F., “Efficient Automated Geometric Feature Recognition through Feature Coding”, *Annals of CIRP* Vol.43/1/1994, p413-416.

[4] Yuen C.F. & Venuvinod P.K., “Geometric Feature Recognition: Coping with the Complexity and Infinite Variety of Features”, *International Journal of Computer Integrated Manufacturing*, Vol. 12, No.5., 1999, p439-452.

[5] Yuen C.F., Coping with the Complexity and Infinite Variety of Geometric Features: A Cooperative Geometric Feature Recognition System, PhD thesis, City University of Hong Kong, 1999.

[6] Henderson M.R., Extraction of Feature Information from Three Dimensional CAD Data, PhD thesis, School of Mechanical Engineering, Purdue University, West Lafayette, Indiana, May 1984.

[7] Joshi S. & Chang T.C., “Graph-based Heuristics for Recognition of Machined Features from a 3D Solid Model”, CAD Vol.20 No.2, Mar 1988, p58-66.

[8] Biederman I., “Human Image Understanding: Recent Research and a Theory”, *Human and Machine Vision II*, Academic Press, Inc., 1985, p13-57.

[9] Schank R.C. and Jona M.Y., “Issue of Psychology, AI, and Education: A Review of Newell’s Unified Theories of Cognition”, in *Contemplating Minds—A Forum for Artificial Intelligence*, The MIT Press, 1994.