Preliminary Draft

Feature-Based Process Planning for CNC Machining

by Elaine Cohen, Samuel Drake, Russell Fish, and Richard F. Riesenfeld

for the 1995 IEEE International Symposium on Assembly and Task Planning
(ISATP'95)
August 10-11, 1995
The Westin William Penn Hotel
Pittsburgh, Pennsylvania

UUCS-98-001

1 Introduction

Today CNC machining is used successfully to provide program-driven medium lot size manufacturing. The range of applicability of CNC machining should be greater: For small lot sizes such as prototyping or custom products, these machines should provide quick turnaround and flexible production scheduling. To set up for larger lot size production, the CNC machines can be used to construct small lots of production tooling, such as jigs, fixtures, molds and dies.

The CIM (Computer Integrated Manufacturing) initiatives of the last decade have done a great deal to integrate CAD (Computer-Aided Design with CAM (Computer-Aided Manufacturing). In fact, marketing has gone on to other buzzwords, as if CIM were a solved problem. So why are there still long delays between product design and production-ready CNC programs? In practice, CAD is often reduced to Computer-Aided Drafting, and CAM becomes Computer-Aided N/C toolpath layout. When CNC programming is done at the level of directing manually selected tools along individually specified motions, at user-specified feedrates and spindle speeds, it is still a laborious, time-consuming process. Worse yet, it must be done by an expert who understands the details of machining and CNC programming well. (These people are usually in short supply and it may be a long time before they can even start on a CNC program.) After using a CAM system to prepare a CNC program, further time must then be spent "proving out" the program in the context of a complete machine setup including fixtures, and there may be further delay in fabricating fixtures to hold the part.

Due to all of these factors (and more!) the critical path between design and production can be very long. We perceive that the delay in producing CNC machining programs is a great barrier to the greater use of CNC machining in prototyping, custom production, and tooling fabrication.

In the Alpha_1 project at the University of Utah Computer Science Department, we decided to look for "higher ground" for CNC programming. Our approach uses explicit feature-based manufacturing process plan models as a basis for automatic high-quality CNC code generation, including tool selection and feed and speed computation, as well as generation of tooling setup instructions for machine operators.

The Alpha_1 software system has been evolved over the last 15 years as a testbed for just this kind of problem. The foundation of Alpha_1 is a set of extensible object-oriented, parametric, geometric modeling and geometric computing libraries. Besides the traditional geometric entities, Alpha_1 uses NURBS (Non-Uniform knot vector, Rational B-Splines) to represent and compute with curves and trimmed-surface sculptured solids.

In the remainder of this paper, we will explain the concepts of feature-based process plans as developed in Alpha_1. The manufacturing algorithms of one Alpha_1 feature type, "general sculptured volumes", will be used as a high-level feature example. A complete process plan example spanning the range of features will be described. Finally, we will indicate some open areas for future work.

2 Feature-Based Process Plan Models in Alpha_1

In Alpha_1, we have provided a variety of manufacturing feature object types, including traditional manufacturing features such as holes and pockets, and non-traditional feature levels as well. Here, we will first describe the "process plan" object level, which organizes feature objects into a manufacturing plan, and then go on to talk about the various feature types and how they are manufactured.

2.1 Process Plan Models

An Alpha_1 *process plan* groups manufacturing feature objects into *stages*. Each stage corresponds to a manufacturing machine setup, so the process plan is like the traditional "routing" between manufacturing and other processes while producing a particular part. [Figure A: diagram of process plan, stages, machining operations.]

For CNC machining stages, each stage gives its "starting point", a properly positioned and oriented model of the initial stock or previous stage of manufacturing. A model of the fixture, vise, or chuck which will hold it during this machining stage may optionally be given. The stage also references a sequence of manufacturing feature objects which will be applied to the part as *machining operations*.

The designer or manufacturing engineer using Alpha_1 for process planning may also specify a *machine class* for each stage. We currently support machine classes for 3-, 4-, and 5-axis milling machines, lathes with or without powered rotary tools ("live tooling"), and Coordinate Measuring Machine inspection. If the machine class is left unspecified on a machining stage, one is chosen algorithmically from the set of machine classes which are listed as available and also fit with the the geometry of the features. (For example, holes off the axis of a surface of revolution but still parallel to the axis require a lathe with live tooling, or transfer to a separate machining stage on a milling machine.)

With this minimal setup and sequencing information from the designer, the process plan model provides a context for automatic tool selection, spindle speed and tool feed rate calculations by the machining operation feature objects. CNC toolpaths for both roughing and finish cutting are generated by methods of the feature objects. For experienced manufacturing engineers and machinists who wish to have more control over tool selection and other low-level machining parameters, a mechanism of *process attributes* is provided to interactively modify the defaults for fine-tuning the process plan.

Of course, the real trick in an object-oriented process plan modeling environment is to choose a set of *parameterized features* that are are high-enough level to provide good design support via "conceptual chunking", while also supporting complete CNC code generation algorithms as methods of each feature object type. These features may be used during "ab initio" design, on portions of a product where the intended manufacturing method is clearly known by the designer. This is intended to support "concurrent engineering" of the product and manufacturing methods. A higher-level design can also be separately post processed by a manufacturing engineer, translating it into manufacturing feature objects and sequencing it into a process plan model.

As a starting point, we borrowed liberally from the traditional concepts of mechanical and manufacturing engineers, translating them into a layer of *conventional manufacturing features*, The parameters of the conventional feature objects are intended to be obvious, such as the center point, depth and diameter of a simple hole.

Many subtypes of hole features are supported (e.g. counter-bores, tapped holes, and so on) with options to allow chamfers, through holes, blind holes, and other common variations. There are a number of pocket features, from rectangular pockets through "profile" pockets which allow arbitrary spline boundary shapes. There are also several types of bosses, grooves, and slots, as well as pattern objects for creating instances of the basic features (e.g. linear, radial, rectangular, and point-list patterns).

Higher-level feature types include sculptured and product features. *general sculptured volumes* correspond to the idea of removing an arbitrary volume of material from the part. (More on this below.) *Product features* make the "conceptual chunking" of feature objects available on a higher level to the end-use environment, taking advantage of commonality of design and manufacturing methods within an

organization or industry. "Power users" or tool builders within the organization can define additional parameterized object types, building on the ones provided by Alpha_1. The manufacturing methods of each product feature object type which is defined can generate a sequence of manufacturing features, or a whole machining stage or process plan, as appropriate.

In the remainder of the paper we will concentrate on the generation of CNC machining instructions. The main output is of course CNC control programs. An informational summary of the single-tool machining operations and their tool assignments and feeds and speeds is another generated output. Finally, a tool list is generated for the machinist/operator who will set up the machine and run the CNC control program.

2.2 Hole Feature Manufacturing

While you may consider a hole to a simple feature of a part, consider that it may require several different tools to make it: center-drill, drill, and chamfer, for example. The algorithm used to automatically generate manufacturing information from the process plan for groups of features allows features to be broken into a sequence of lower-level *single-tool operations*, which are then re-ordered to optimize the manufacturing sequence - both in travel distance and to eliminate time-consuming tool changes.

The same hole feature might generate a variety of manufacturing instructions, depending on such things as availability of tools in the tool library and tolerances specified with the feature object. A through-hole is drilled to a different depth than a blind hole, and a tapped hole must be pre-drilled at a smaller size. A hole with a tight diameter tolerance is drilled undersize and then reamed to size, instead of just drilled.

2.3 Automatic Roughing and Finish Cutting of Conventional Features

The machining instructions which are automatically generated from the process plan information take advantage of the high-level feature descriptions to provide automatic roughing and finish cutting for conventional features like pockets and turned parts.

The very general pocket and boss features, which have flat bottoms and straight walls, generate roughing and finish machining passes which handle arbitrary profile shapes, islands, draft angles, and rounded bottom edges. Notice that a boss feature, which leaves protruding islands of material, is exactly like an inside-out pocket feature, which removes the material in its interior. One difference is that CNC toolpaths for a boss must be limited by some external outline, such as as the outline of the stock material from which the part is being machined.

For turned parts, an automatic decomposition is done on the profile curve extracted from a surface of revolution feature object. Roughing and finishing passes are generated, with various tools used to machine different parts of the profile. Lathe cutting tools often will cut in only one direction: left-to-right, right-to-left, in a bore or across a face. This complicates the toolpath generation algorith somewhat.

In the next section, we will see how rough machining of sculptured features is in fact built upon the machining methods of conventional pocket features in Alpha_1.

3 Machining "General Sculptured Volume" Features

Since Alpha_1 contains a large variety of modeling operations which are often used to create highly sculptured surfaces, it provides an ideal environment for exploring the issue of process planning for parts which contain these complex surfaces. Most parts which contain free-form surfaces also contain a combination of conventional manufacturing features as well.

Sculptured volumes can be modeled in Alpha_1 using general sweep operations, shaping operations such as warping and bending, boolean set operations, and others. A sculptured volume in Alpha_1 is represented by a *shell*, which is simply a collection of surfaces together with the adjacency topology

which "sews" the surfaces together. The representation for the component surfaces in the shell is a trimmed NURBS of arbitrary order. [Figure 1: a blade gap volume]

Although only (portions of the) surfaces of the sculptured volume will appear on a finished part, the feature object, considered as a machining operation, actually specifies removing the volume of material enclosed by the surfaces. Thus, there are two families of algorithms which we have explored for generating the machining instructions for a sculptured volume feature: one for the roughing pass, and one for finishing.

We have implemented several algorithms for generating sculpturing toolpaths in Alpha_1, differing in the way they sweep out the interior and surfaces of the volume, as well as other aspects such as the choice of tool type and control of the tool orientation direction for 5-axis sculpturing. The simplest and to date most generally useful roughing and finishing algorithms are described next.

3.1 Volume Roughing Algorithm

A volume roughing algorithm is used to remove the bulk of the interior of a volume feature, leaving a specified allowance for finishing. It makes a series of planar section cuts of the volume, and then uses the conventional pocketing algorithms to generate manufacturing instructions for each of the sections in the following way:

- A series of "step planes" are constructed for the sculptured volume. The intersection curve of each plane relative to the shell surfaces is computed. These intersection curves are closed contours, which are machined exactly like conventional pockets with curve outlines. [Figure 2. Slice curves through the volume]
- An arc and line approximation of the contour curves for each slice is computed, based on a specified tolerance. [Figure 3. Arc and line approximation for one of the curves]
- The *skeleton* of the arc and line approximation curve is computed. The skeleton is the dual of the Voronoi diagram, which has been generalized to handle arcs as well as polygons. The skeleton provides a way to represent offsets from the boundary curve. [Figure 4. Skeleton for the same slice]
- The machining instructions which are generated plunge the cutter into the "center" of the skeleton (if it is a closed loop), cutting swaths spaced (by default) at half the cutter diameter from each other. The path is expanded towards the boundary each time around until only a distance equal to the roughing allowance plus the cutter radius is left. [Figure 5. Toolpath of one slice computed from skeleton] [Figure 6. Stack of toolpaths for the volume]

3.2 Volume Finishing Algorithm

The volume finishing algorithm is applied to each surface in the volume after the roughing cuts are completed. It uses a technique of adaptive offset iso-curves to provide valid, optimal toolpaths for the finishing.

A ball-end cutter is used for this machining operation, and this requires that the surfaces be offset by the cutter radius. For volumes which are the result of boolean set operations, offset surfaces for each of the shell surfaces can be computed and the booleans re-executed to re-trim the surfaces. This results in a new sculptured volume which is the offset of the original. The problem is reduced to computing offsets for the components of the boolean set operation. For simple features with analytic descriptions, exact offset methods are used. For complex free-form surfaces, an adaptive offset algorithm is applied. In an iterative process, a fairly simple (but not necessarily very accurate) offset algorithm is applied. Then, analytical spline operations are used to evaluate the error and to direct refinement of the surface to areas with large error. The process is repeated until the offset surface is within the specified tolerance.

Once the offset surface approximations are generated, an appropriate toolpath for the offset surface must be computed. A set of adaptive iso-curves are generated to cover the offset surface. [Figure 7. Adaptive iso-curves vs. non-adaptive] The adaptive iso-curve generation algorithm has the following characteristics:

- Each iso-curve is spaced at the largest distance from adjacent iso-curves which is still less than a specified maximum distance. This maximum distance is selected as a function of the tool radius and the maximum scallop-height which is acceptable on the finished surface.
- Analytic spline computations are used to compute a bound on the distance between two iso-curves in the surface. If the distance is everywhere less than the specified maximum, then the two iso-curves are close enough. Otherwise the iso-curve can be subdivided at the positions where the distance becomes too large, and a new iso-curve inserted between the existing ones. This recursive subdivision process continues until enough iso-curves have been generated to form an appropriate toolpath for the surface.
- Iso-curves that are too close to their adjacent iso-curves are trimmed away as a post-process. Those that cross surface trimming curves are also trimmed away.

4 Manufacturing Example

As an example of the use of sculptured volume features in modeling and manufacturing a part, a one-piece compressor disk for a jet turbine engine is shown. [Figure 8. Raster image of the part] [Figure 9. Photograph of the manufactured part] The part has many sculptured surfaces, in combination with conventional features such as holes and surfaces of revolution. The solid model of the part is expressed by using sculptured volume features to remove the metal in the gaps between the compressor blades via boolean set operations. [Figure 10. Ring of blade gaps]

The process plan for machining the compressor disk consists of these stages: [Figure B: Diagram of compressor disk process plan]

- Mill the center hole and bolt holes in a slab of stock. [Figure 11. Slab with holes.] This is done on a 3-axis milling machine with the stock clamped down, and prepares for the turning operation.
- Turn the surfaces of revolution on an arbor fixture mounted in a lathe. [Figure 12. Surfaces of revolution turned from slab.]

  Initial setup includes making the arbor fixture with a process plan of its own. Before being mounted on the fixture, the slab of stock is rough-sawn using a bandsaw to an approximately round "wheel" shape. (This was more efficient in our prototyping environment than milling around the outline of the wheel.)
- Mill the gaps between the blades on a tilt-rotate (5-axis) table fixture. [Figure 13. Final part - repeat of Figure 8 or 9.]

5 Conclusions

In this paper, we have described the use of feature-based process plan models in Alpha_1, and the way in which they are processed into CNC manufacturing instructions which direct setup and operation of the production machinery.

There are of course many areas which are opened up for further work by these results. One is that the complete geometric model provides a basis for *global collision avoidance*. More complete models of the production machinery, tool holders, and so on are of course needed to support this. While we

have done work on simulating the machine model running the CNC control program to **detect** collisions between the cutting tool and the machine and gouging of the part by the tool, it is far more preferable to generate toolpaths which are known to **avoid** collisions and gouging.

We have also considered how this sort of process plan models can themselves be *generated algorithmically* from un-ordered sets of manufacturing features, by taking advantage of the common orientations of feature axes and other geometric information. Similarly, work can be done on methods to *translate* models constructed from design feature objects which do not themselves have direct manufacturing methods, producing derived manufacturing feature models that are directly manufacturable.

Finally, as we work our way up the "engineering food chain" from manufacturing engineering to earlier stages of the product engineering process, it seems clear that a similar approach using *design feature objects* will provide automated support for design engineering and detailing. Our approach is not to make the process automatic, so much as to provide high-level "power tools" to automate the time-consuming and error-prone aspects of the engineering process. In synergy with the work reported here on feature-based process planning for CNC machining, prototypes of the designed products could be rapidly produced by generating machining process plans, and production tooling and fixturing would follow as well.