Int J Adv Manuf Technol (2016) 83:561–574
DOI 10.1007/s00170-015-7552-x

ORIGINAL ARTICLE

A real-time surface interpolator methodology for precision CNC machining of swept surfaces

Kevin M. Nittler[1] · Rida T. Farouki[1]

Received: 10 March 2015 / Accepted: 3 July 2015 / Published online: 29 July 2015

Abstract A real-time surface interpolator is developed to machine a family of swept surfaces directly from their high-level procedural definitions. All the computations required for machining are performed in real time based on the exact surface geometry, including tool path planning, tool path interpolation, tool offsetting, and tool path step-over to achieve a prescribed scallop height. A G-code command (G05) is introduced to concisely communicate the precise surface geometry and all necessary process parameters to the controller. The swept surface interpolator offers profound accuracy and efficiency advantages over the traditional approach of generating voluminous piecewise–linear/circular tool path approximations as a preprocessing step. For example, in one instance, a 36,000-line piecewise-linear (G01) approximate part program file is replaced by a 3-line exact swept surface (G05) part program file. The methodology is verified by machining a variety of swept surface forms in aluminum and wax, using a 3-axis milling machine with the surface interpolator incorporated into an open-architecture CNC controller.

Keywords Rational swept surfaces · Real-time interpolator · Surface machining · G-codes · Tool paths · Cutter contact point · Tool offset · Scallop height

✉ Rida T. Farouki
farouki@ucdavis.edu

Kevin M. Nittler
kmnittler@ucdavis.edu

1 Department of Mechanical and Aerospace Engineering, University of California, Davis, CA 95616, USA

1 Introduction

Modern CAD/CAM technology for CNC machining is based on two disparate and fundamentally incompatible paradigms: the precise rational parametric surface descriptions employed in geometric design, and the G-code standard (ISO 6983) used in machine tool programming, which relies on data-intensive piecewise-linear/circular approximations to curved tool paths.

In traditional CNC milling, large areas of simple surfaces are generated by the flank or end of a cylindrical tool, using relatively few tool path segments specified by standard G-codes. On the other hand, machining of "free-form" surfaces that must satisfy stringent functional or stylistic requirements is an increasingly important problem. In this context, the tool cannot generate a large surface area with a single motion. Instead, approximate methods must be employed, the most common being the use of a ball–end mill executing many successive tool path passes separated by a small "step-over" distance, so as to approximate the free-form surface to a desired accuracy and surface roughness. This *contour machining* strategy, which is used extensively in the biomedical, die and mold, and aerospace industries, is a mathematically and computationally demanding problem because of the need to plan a sequence of tool paths to efficiently cover an entire surface area while maintaining the prescribed surface accuracy, implementing tool offsets, and accommodating the kinematic complexities incurred by 5-axis machining.

In modern CAM-based CNC machining work flow, illustrated in Fig. 1, a part with free-form surfaces is typically generated as a NURBS-based solid model in a CAD system, and then passed to a CAM system (separate from or integrated with the CAD system). The CAM program performs calculations based on the parametric surface geometry to

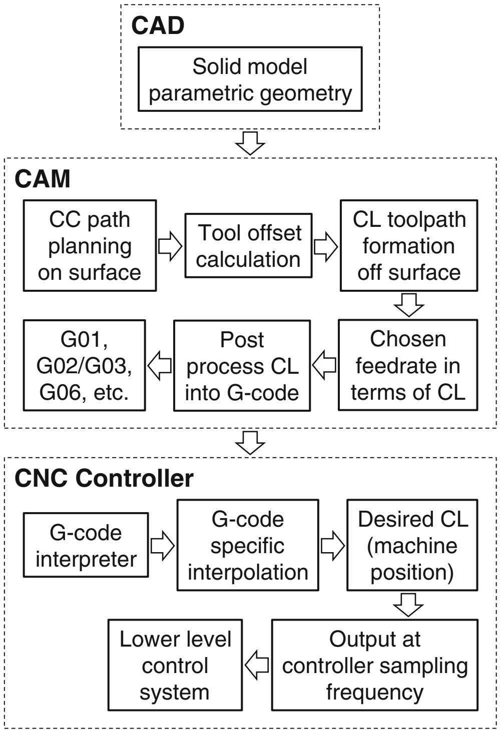


Fig. 1 Traditional workflow for CAM–based CNC machining

generate initial cutter contact (CC) paths on the surface, which are transformed into cutter location (CL) tool paths through tool offsetting. This tool path CL data is then converted to appropriate G-codes by means of a post processor, which is controller or even machine specific, depending on the machine complexity. The controller must then execute[1] long streams of short G-code motions that define data-intensive approximate tool paths for machining the part surfaces.

The deficiencies of the G-code system, as currently implemented, have been extensively documented [11]. The reliance upon linear (G01) segments and sometimes circular (G02/G03) segments to approximate general curved tool paths results in loss of accuracy, enormous part programs and data rates, acceleration discontinuities, aliasing effects at high feedrates, etc. In surface machining, these effects are exacerbated by the need to employ large numbers of closely spaced tool passes to ensure a desired surface accuracy. Moreover, G-code part programs entail an irreversible loss of precise surface geometry information. For example, part programs must be re-generated from scratch when a different tool is chosen, since they incorporate no information on the surface normal variation which is required for the tool offset.

[1]To improve performance, some controllers provide "on-the-fly" smoothing functions that perform real-time spline approximations of long G-code sequences.

The generality of the G-code approximation method offers a functional but obviously sub-optimal approach to surface machining. Compared to the significant advances made in precision machine tools, controller technology, and the ability of CAM systems to generate complex surface tool paths, the prevailing methodology for communicating such tool path information to the machine controller remains rather archaic and primitive.

Between the extremes of simple surfaces and general free-form surfaces, there exists a versatile and important family of *swept surfaces*, generated by using a *sweep curve* to impose a continuous sequence of intuitive geometrical transformations on a *profile curve* [12]. The range of transformations that can be invoked, while ensuring *exact* compatibility with the rational parametric geometry representations of CAD systems, has been vastly expanded [8] by exploiting the distinctive features of *Pythagorean-hodograph* (PH) *curves* [3] as sweep curves. Such surfaces have important applications in the functional design of dies and molds, ducting, variable blends between intersecting free-form surfaces, and also in stylistic and ergonomic design contexts.

This paper presents an approach to overcome the limitations of traditional G-code path planning, in the context of the compelling class of swept surface forms introduced in [8]. A real-time CNC interpolator is developed, whose input is just the high-level procedural definition of such a surface (specified by PH sweep and profile curves), and a modest set of ancillary data defining the sweep operation, tool radius, machining tolerance, etc. The methodology is demonstrated by machining a selection of swept surfaces that illustrate the versatility of the possible sweep operations. The part programs are extremely compact, based on novel G-codes to ensure that the controller always uses the exact surface definition, and all machining computations are performed in real time based on this exact form. This real-time *swept surface interpolator* eliminates the voluminous, approximate G-code part programs that limit the accuracy and efficiency of traditional surface machining methodology.

The plan for the remainder of this paper is as follows. Section 2 reviews some relevant background information concerning the current state of the art in real-time CNC interpolators and surface machining. The diverse family of rational swept surfaces recently introduced in [8], on which the present study is focused, is then briefly reviewed in Section 3. The algorithmic details of the swept surface interpolator methodology are presented in Section 4. Section 5 describes its implementation on a 3-axis CNC milling machine with an open-architecture software controller, and the outcome of machining experiments on a selection of swept surface types. Finally, Section 6 summarizes the key results of this study and identifies topics for further investigation.

2 Background

Compared to the substantial literature on real-time curve interpolators, the study of *surface* interpolators has received relatively little attention. Koren and Lin [13] proposed a real-time interpolator for 5-axis machining of bicubic surface patches, which was further refined by Lin [16]. Lo [20] formulated a surface interpolator with a particular emphasis in cutter contact (CC) point path planning. Tsai et al. [24] and Cheng and Tseng [1] implemented NURBS surface interpolators, while Wang et al. [26] proposed a NURBS interpolator with more sophisticated acceleration/deceleration control.

The focus of this study is on developing a real-time interpolator for the class of rational swept surfaces introduced in [8], including implementation on a 3-axis mill with an open-architecture controller, and a demonstration of the machining of several representative surfaces (the emphasis is mainly on *finish machining*, although the methodology can also be used for rough cuts). The goal is to avoid the inter-related problems incurred by discretizing surface tool paths into numerous short linear G01 path segments, including:

1. High data rates that scale with both surface area and desired accuracy in both the tool path and stepwise directions
2. The inherent inaccuracy of the discretized tool path approximation that results in both geometrical and feedrate errors
3. The irrecoverable loss of surface geometry information that occurs when only one-dimensional tool path data is communicated

The high data rates (and to some extent the tool path error) can be addressed by more sophisticated (i.e., parametric curve) real-time interpolators, which can accommodate variable feedrates [10]. Some capability is already available in industrial CNC controllers, e.g., the NURBS interpolator in the FANUC G06.2 protocol [2] and spline interpolator in the Siemens BSPLINE command [21] but has not seen widespread use. Some controllers also offer "on-the-fly" spline smoothing of traditional G code part programs to ensure a smoother path execution, although this entails yet another layer of approximation.

The methodology proposed herein is based on the Pythagorean-hodograph (PH) curves [3]. The distinctive feature of a planar[2] polynomial PH curve $\mathbf{r}(\xi) = (x(\xi), y(\xi))$ is that its hodograph (derivative) components satisfy, for some polynomial $\sigma(\xi)$, the Pythagorean condition

$$x'^2(\xi) + y'^2(\xi) = \sigma^2(\xi). \tag{1}$$

$\sigma(\xi)$ defines the *parametric speed* of $\mathbf{r}(\xi)$—i.e., the derivative $\mathrm{d}s/\mathrm{d}\xi$ of arc length s with respect to the curve parameter ξ. For polynomials $\alpha(\xi), \beta(\xi)$, the solutions to Eq. 1 are [3] of the form

$$x'(\xi) = \alpha^2(\xi) - \beta^2(\xi), \quad y'(\xi) = 2\alpha(\xi)\beta(\xi),$$
$$\sigma(\xi) = \alpha^2(\xi) + \beta^2(\xi). \tag{2}$$

Since $\sigma(\xi)$ is a polynomial, PH curves admit exact algorithms for many basic computations that otherwise require approximations. The tangent $\mathbf{t}(\xi)$, normal $\mathbf{n}(\xi)$, and curvature $\kappa(\xi)$ of a PH curve are rational functions of the parameter, and the offset curves $\mathbf{r}_d(\xi) = \mathbf{r}(\xi) + d\,\mathbf{n}(\xi)$ at each distance d [7] are rational curves. The indefinite integral of $\sigma(\xi)$—the cumulative arc length $s(\xi)$—is also a polynomial. Consequently, with a PH curve, the *interpolation integral* (incurred in computing the reference point parameter values by a real-time CNC interpolator) admits analytic reduction for feedrates that are constant or dependent on time, arc length, curvature, etc. [5, 6, 9, 25].

A further advantage of PH curves is the possibility [8] of vastly extending the range of rational surfaces that can be exactly generated through intuitive sweep operations, in which a *sweep curve* is employed to specify a continuous family of transformations acting on a *profile curve*. With a PH sweep curve, it becomes possible to impose coordinated transformations, such as moving the profile curve along the sweep curve while orienting it in the sweep curve normal plane, or scaling it by a function of the sweep curve arc length, while guaranteeing an exact *rational swept surface* as the outcome.

The goal of this study is to develop a real-time interpolator that cuts the rational swept surfaces described in [8] directly from their exact definitions, i.e., the profile and sweep curve specifications, the sweep transformation type, and a modest set of machining parameters. To accomplish this, a novel G-code structure is proposed to communicate such surface definitions directly to the controller, so it can generate motion commands from them.

The proposed approach is in the spirit of the STEP-NC AP238 standard—a slowly progressing alternative to the G-code standard (ISO 6983), and a component of the larger STEP standardization movement. A recent overview of STEP-NC may be found in [11]. A key intent of this standard is to furnish the controller with more information about the part geometry, including the complete solid model if possible. Although the methodology presented here is in the more readily implemented G-code form, it is clearly just as applicable as a component of the STEP-NC approach. Liang

[2] For brevity, we restrict our attention here to *planar* PH curves—further details, including the extension to spatial PH curves, may be found in [3].

and Li have demonstrated this with a G-code surface interpolator [14] that was subsequently updated to the STEP-NC context [15], employing the same core methods but with a STEP-NC interpreter instead of a G-code parser as the front end.

3 Swept surface constructions

The real-time surface interpolator developed herein focuses on the rational swept surfaces introduced in [8] and briefly described below. A swept surface $\mathbf{R}(u, v)$ is generated by invoking a sweep curve $\mathbf{s}(v)$ to specify a continuous family of transformations that act on a profile curve $\mathbf{p}(u)$. We assume that $\mathbf{p}(u)$ are $\mathbf{s}(v)$ are planar PH curves, but the methodology can be extended to spatial curves, and for a rational surface it is only essential that $\mathbf{s}(v)$ be a PH curve. Since the basic definition admits a remarkable variety of swept surfaces, ranging from simple translational/rotational surfaces to geometries generated by multiple simultaneous transformations, the surface interpolator will be demonstrated on a representative sample of swept surfaces.

A rational surface patch $\mathbf{R}(u, v) = (x(u, v), y(u, v), z(u, v))$ with $(u, v) \in [\,0, 1\,] \times [\,0, 1\,]$ is specified by its homogeneous-coordinate polynomials $W(u, v)$, $X(u, v)$, $Y(u, v)$, $Z(u, v)$ such that

$$x(u, v) = \frac{X(u, v)}{W(u, v)}, \quad y(u, v) = \frac{Y(u, v)}{W(u, v)},$$
$$z(u, v) = \frac{Z(u, v)}{W(u, v)}.$$

Let $x(u)$, $y(u)$, $z(u)$, $x'(u)$, $y'(u)$, $z'(u)$, $\sigma(u)$, $s(u)$, $f(u), \ldots$ and $x(v)$, $y(v)$, $z(v)$, $x'(v)$, $y'(v)$, $z'(v)$, $\sigma(v)$, $s(v)$, $f(v), \ldots$ define *generalized coordinates*[3] for the PH profile curve $\mathbf{p}(u)$ and sweep curve $\mathbf{s}(v)$, comprising the coordinate components, their derivatives, the parametric speed, the arc length function, and certain user-specified polynomial functions. In the most general terms, a rational swept surface $\mathbf{R}(u, v)$ is defined by specifying its four homogeneous coordinate components $W(u, v)$, $X(u, v)$, $Y(u, v)$, $Z(u, v)$ as sums of products of the profile and sweep curve generalized coordinates.

The generalized coordinates can be extended to encompass higher-order quantities, such as curvature. However, even without such an extension, the basic definition admits a bewildering variety of possibilities. It is therefore desirable to focus on examples corresponding to geometrically intuitive sweep transformations of obvious practical interest. If $\mathbf{p}(u)$ and $\mathbf{s}(v)$ are PH curves, their generalized coordinates are all *polynomial* functions of their respective parameters, and the swept surfaces defined in the above manner are *rational* surfaces, exactly compatible with prevailing CAD geometry representations.

Let $(\mathbf{i}, \mathbf{j}, \mathbf{k})$ be unit vectors in the (x, y, z) directions. For brevity, we will illustrate the swept surface constructions for input curves in "canonical form"—namely, a profile curve $\mathbf{p}(u) = x(u)\,\mathbf{i} + z(u)\,\mathbf{k}$ in the (x, z) plane, and sweep curve $\mathbf{s}(v) = x(v)\,\mathbf{i} + y(v)\,\mathbf{j}$ in the (x, y) plane, with $u \in [\,0, 1\,]$ and $v \in [\,0, 1\,]$. For machining, it may be necessary to transform the resulting surfaces to the machine coordinate system by an appropriate rotation/scaling/translation. Once a swept surface $\mathbf{R}(u, v)$ has been constructed, the unit surface normal vector may be defined in terms of its partial derivatives by

$$\mathbf{N}(u, v) = \frac{\mathbf{R}_u(u, v) \times \mathbf{R}_v(u, v)}{|\,\mathbf{R}_u(u, v) \times \mathbf{R}_v(u, v)\,|}. \tag{3}$$

For all the swept surface types discussed below, a closed-form expression for the surface normal can be found in [8].

3.1 Scaled-rotation sweep

Let $\rho(v) = \sqrt{x^2(v) + y^2(v)}$ and $\phi(v) = \tan^{-1} y(v)/x(v)$ be polar coordinates for the sweep curve. Then the scaled-rotation swept surface is generated by continuously rotating $\mathbf{p}(u)$ by angle $\phi(v)$ about the z axis, while scaling it parallel to the (x, y) plane by the factor $\rho(v)$—see Fig. 2. This may be regarded as a generalization of the surfaces of revolution, which correspond to the special case $\rho(v) =$ constant (i.e., the sweep curve is a circular arc). The scaled-rotation sweep has the particularly simple closed-form expression

$$\mathbf{R}(u, v) = x(u)\,\mathbf{s}(v) + z(u)\,\mathbf{k}, \tag{4}$$

corresponding to the homogeneous coordinates

$$W(u, v) = 1, \quad X(u, v) = x(v)x(u),$$
$$Y(u, v) = y(v)x(u), \quad Z(u, v) = z(u).$$

This is an example of a set of rational swept surfaces [12] that depend only on the *coordinates* of $\mathbf{p}(u)$ and $\mathbf{s}(v)$, not their derivatives, parametric speed, arc length, etc. The $u =$ constant isoparametric curves are parallel to the (x, y) plane, at height $z(u)$ above it, with parametric speed $|\mathbf{R}_v(u, v)| = |x(u)|\,\sigma(v)$.

3.2 Oriented-translation sweep

The oriented-translation swept surface is generated by translating the profile curve by the coordinates of the sweep curve, while simultaneously orienting it in the sweep curve

[3] For brevity, the same symbols are used to denote generalized coordinates of both profile and sweep curves, the independent variable identifying the curve under consideration.

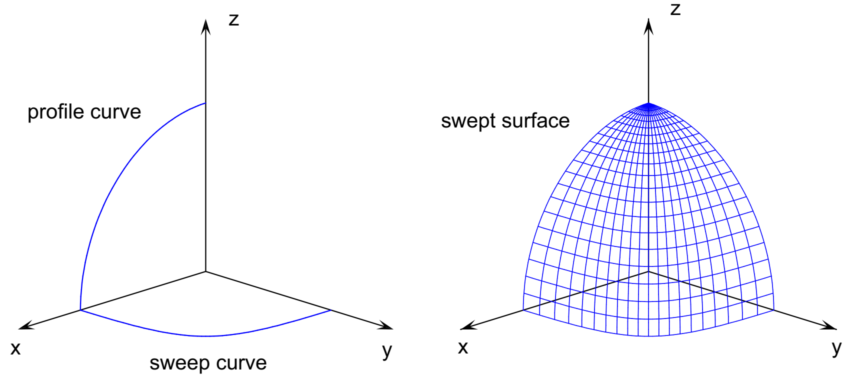


Fig. 2 An example of the scaled-rotation swept surface

normal plane (see Fig. 3). This is a fundamental and intuitive sweep transformation that has widespread applications in geometric design. The use of a PH sweep curve allows an exact rational surface outcome in instances that would otherwise necessitate approximation.

The oriented-translation swept surface may be expressed in the form

$$\mathbf{R}(u,v) \;=\; \mathbf{s}(v) + x(u)\,\mathbf{n}(v) + z(u)\,\mathbf{k}\,, \tag{5}$$

where $\mathbf{n}(v) = (y'(v)\,\mathbf{i} - x'(v)\,\mathbf{j})/\sigma(v)$ is the (rational) unit normal to $\mathbf{s}(v)$—the corresponding homogeneous coordinates are

$$W(u,v)=\sigma(v)\,, \quad X(u,v) \;=\; \sigma(v)x(v) + y'(v)x(u)\,,$$
$$Y(u,v)=\sigma(v)y(v) - x'(v)x(u)\,, \quad Z(u,v) \;=\; \sigma(v)\,z(u)\,.$$

The u = constant isoparametric curves are all parallel to the (x, y) plane, at height $z(u)$ above it, with parametric speed $|\mathbf{R}_v(u,v)| \;=\; \sigma(v)\,|\,1 + \kappa(v)x(u)\,|$, where $\kappa(v) \;=\; (x'(v)y''(v) - x''(v)y'(v))/\sigma^3(v)$ is the curvature of $\mathbf{s}(v)$.

3.3 Oriented-involute sweep

The *involute* to $\mathbf{s}(v)$ is the locus defined by $\mathbf{c}(v) = \mathbf{s}(v) - s(v)\,\mathbf{t}(v)$, where $s(v)$ and $\mathbf{t}(v) = (x'(v)\,\mathbf{i} + y'(v)\,\mathbf{j})/\sigma(v)$ are its cumulative arc length and unit tangent. The involute is a *rational curve* if $\mathbf{s}(v)$ is a PH curve, since $s(v)$ is a polynomial function and $\mathbf{t}(v)$ is a rational vector. Involute curves play a key role in the kinematics of gears, since they ensure "conjugate action" (i.e., a strictly constant angular velocity ratio) of meshing gears.

The oriented-involute sweep is similar to the oriented-translation sweep, except that the *involute* to the sweep curve is used to specify the continuous translation/orientation of the profile curve (see Fig. 4). It is defined by

$$\mathbf{R}(u,v) \;=\; \mathbf{c}(v) - x(u)\,\mathbf{t}(v) + z(u)\,\mathbf{k}\,, \tag{6}$$

with corresponding homogeneous coordinates

$$W(u,v) \;=\; \sigma(v)\,,$$
$$X(u,v) \;=\; \sigma(v)x(v) - x'(v)\,[\,s(v) + x(u)\,]\,,$$
$$Y(u,v) \;=\; \sigma(v)y(v) - y'(v)\,[\,s(v) + x(u)\,]\,,$$
$$Z(u,v) \;=\; \sigma(v)\,z(u)\,.$$

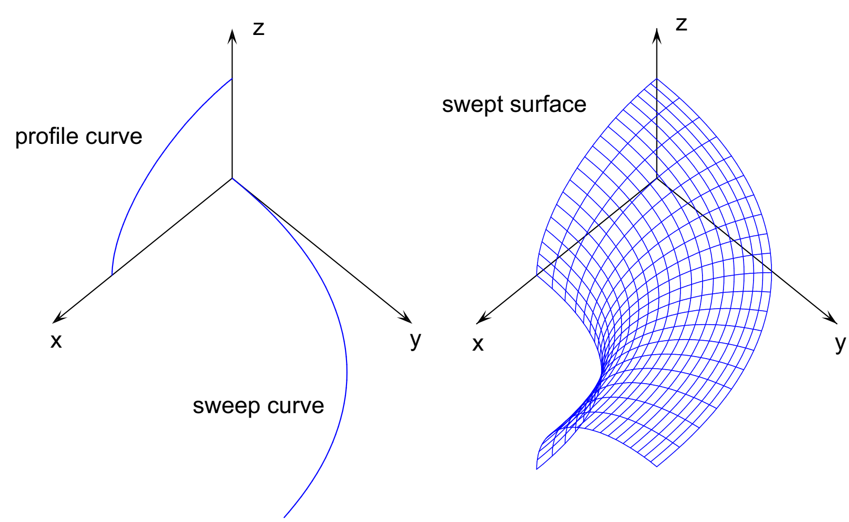


Fig. 3 An example of the oriented-translation swept surface

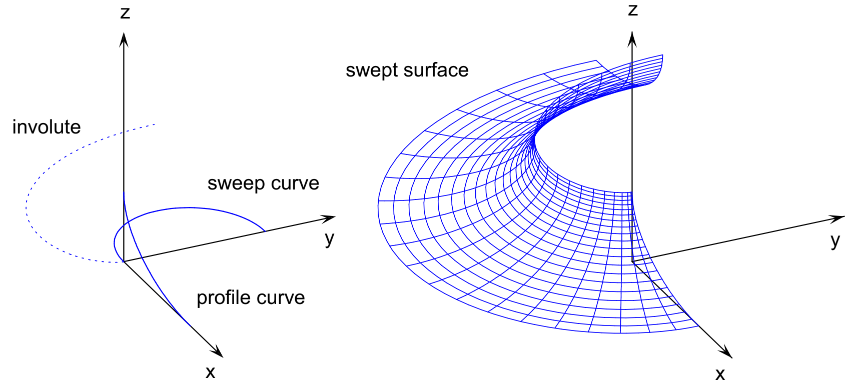


Fig. 4 An example of the oriented-involute swept surface

This defines a rational surface when $\mathbf{s}(v)$ is a PH curve. The u = constant isoparametric curves are parallel to the (x, y) plane, at height $z(u)$ above it, with parametric speed $|\mathbf{R}_v(u, v)| = |\,(x(u) + s(v))\kappa(v)\,|\,\sigma(v)$.

3.4 Generalized conical sweep

The generalized conical sweep may be viewed as an extension of the oriented-translation sweep, in which a scaling factor dependent upon arc length along the sweep curve is superposed on the translation/orientation transformations. For a PH sweep curve, the resulting surface is rational. We consider here just a simple (linear) variation of the scale factor, namely

$$c(v) = \frac{c_0(S - s(v)) + c_1 s(v)}{S},$$

where c_0 and c_1 are specified initial and final (positive) scale factors, at $v = 0$ and $v = 1$, and $S = s(1)$ is the total arc length. The expression

$$\mathbf{R}(u, v) = \mathbf{s}(v) + c(v)\,[\,x(u)\,\mathbf{n}(v) + z(u)\,\mathbf{k}\,], \tag{7}$$

defines the resulting swept surface, with homogeneous coordinates

$$\begin{aligned}
W(u, v) &= S\sigma(v)\,, \\
X(u, v) &= S\sigma(v)x(v) + [\,c_0 S + (c_1 - c_0)s(v)\,]\,y'(v)x(u)\,, \\
Y(u, v) &= S\sigma(v)y(v) - [\,c_0 S + (c_1 - c_0)s(v)\,]\,x'(v)x(u)\,, \\
Z(u, v) &= [\,c_0 S + (c_1 - c_0)\,s(v)\,]\,\sigma(v)z(u)\,.
\end{aligned}$$

An example is shown in Fig. 5—because of the scaling, the u = constant isoparametric curves are non-planar, with parametric speed

$$|\mathbf{R}_v(u, v)| = \sqrt{\sigma^2(v)\,[\,1 + c(v)\kappa(v)x(u)\,]^2 + c'^2(v)\rho^2(u)}$$

where $\rho(u) = \sqrt{x^2(u) + z^2(u)}$.

4 Real-time swept surface interpolator

The real-time interpolator in a CNC system computes a *reference point* (i.e., commanded position) in each controller sampling interval, from a given path geometry and feedrate. The *position error* (difference between commanded

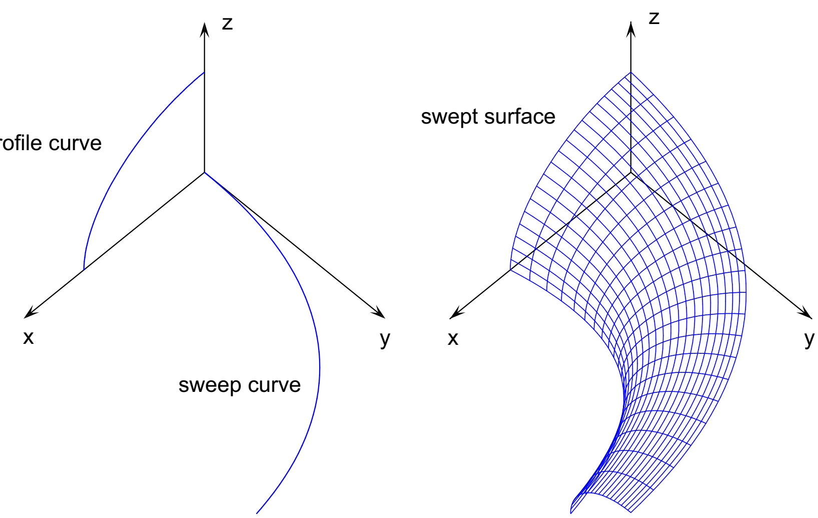


Fig. 5 A generalized conical swept surface with $c_0 = 1.0$ and $c_1 = 1.5$

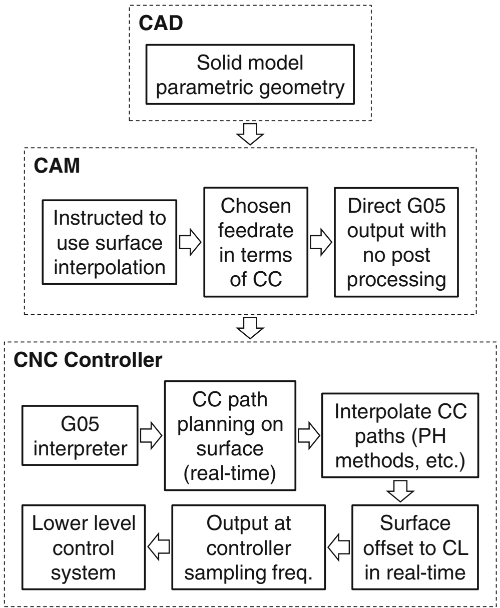


Fig. 6 Proposed workflow for the real-time surface interpolator

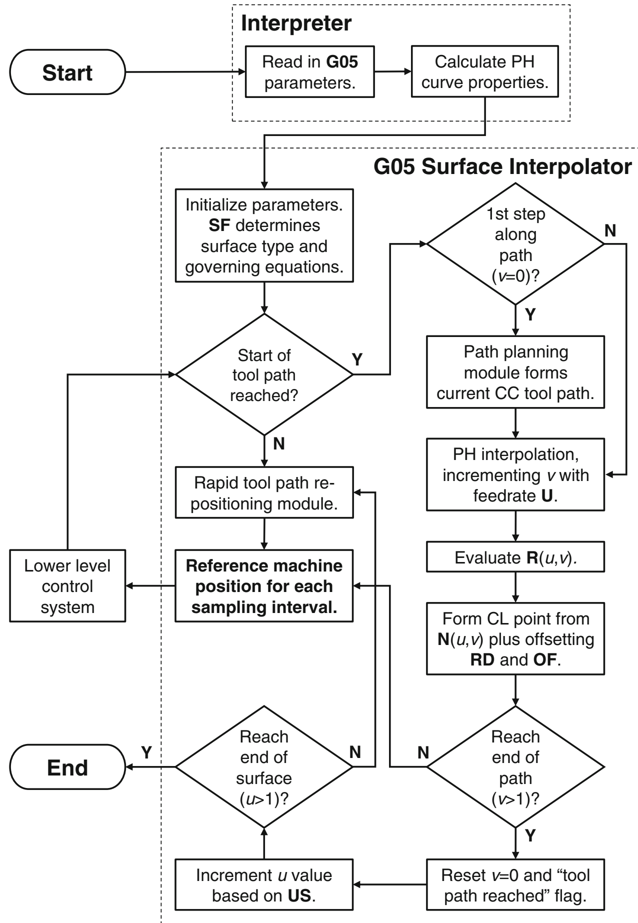


Fig. 7 Simplified logic flowchart for the real-time G05 surface interpolator using an iso-parametric tool path machining strategy

and actual position, measured by encoders on the machine axes) is the basic input to the controller. Traditionally, real-time interpolators were restricted to simple linear (G01) or circular (G02/03) path segments, but interpolator algorithms for free-from parametric curves are now more widely available.

A surface interpolator extends this function to a two-dimensional surface. Consequently, it entails a path planning function to guarantee a tool path coverage of the surface that suffices for a desired machined surface accuracy. Figure 6 outlines the proposed surface interpolator. Compared to traditional CNC machining workflow (see Fig. 1), it streamlines the overall process, performing most required computations directly in the controller using the exact surface description. It can also utilize real-time (e.g., position encoder) data, that is not available to off-line path planning strategies.

The G05 code has previously been adopted [6] to communicate PH curve data to a CNC system. In the present context, it is generalized to incorporate all the information required for a surface interpolator to machine a rational swept surface generated by PH curves. Figure 7 shows a simplified flow-chart for the interpolator. It employs a specific path-planning strategy, starting at the surface point $(u, v) = (0, 0)$ and following isoparametric curves in the v direction, with a step-over in the u direction at the end of each tool path. The entire surface is machined from just a single G05 command, based on a set of machining parameters specified within the G05 block. The interpolator has been implemented in the OpenCNC open-architecture controller.

4.1 G-code structure

A G-code part program consists of a series of lines or "blocks" containing one or more "words" that comprise an address character (specifying a function or setting to be executed) paired with a numerical value. A complete G-code command may be just a single word, or a multi-block segment of a specific order [2]. Although simple commands are common to all controllers, higher-level commands specific to a given controller and its G-code interpreter are also used. The upper part of Fig. 7 corresponds to the new G-code format and the interpreter that parses it. This sets up the real-time interpolator with the all the information needed to machine the entire swept surface.

The G05 code, adopted in [6] to specify PH curves, is extended here to define swept surfaces.[4] The conventions

[4]G05 was originally unused in the ISO 6983 standard, although some controllers assign functions to it.

Table 1 Address characters for the G05 swept surface interpolator

Address	Interpretation
G	Preparatory function
SF	Swept surface type
NS	Number of surfaces (SF08 multiple surface mode)
RA	Angle for rotational transformation of surface
CI, CF	Initial/final scale factors for SF05 surface
F	Feedrate type
U, V, W	Feedrate parameters
RD	Tool radius
OF	Surface offset distance
US	Scallop height parameter
SH	Degree of PH curve (sweep curve)
SX, SY, SZ	End–point coordinates (sweep curve)
SA, SB, SC, SD, SE	$\alpha(\xi)$ Bernstein coefficients (sweep curve)
SP, SQ, SR, SS, ST	$\beta(\xi)$ Bernstein coefficients (sweep curve)
PH	Degree of PH curve (profile curve)
PX, PY, PZ	End–point coordinates (profile curve)
PA, PB, PC, PD, PE	$\alpha(\xi)$ Bernstein coefficients (profile curve)
PP, PQ, PR, PS, PT	$\beta(\xi)$ Bernstein coefficients (profile curve)

chosen here are arbitrary, and can be modified to any implementation, such as STEP-NC. Table 1 summarizes the address characters used in the G05 swept surface definition. The SF address identifies the sweep type, communicating to the controller how the surface should be constructed (see Section 3) from the profile and sweep curves. At present, SF01 corresponds to the scaled-rotation sweep, SF02 the oriented-translation sweep, SF04 the oriented-involute sweep, SF05 the generalized conical sweep, and SF08/SF09 are used to delimit the multi-surface mode. The F address indicates the feedrate type, with U, V, W being the numerical parameters that specify[5] the feedrate variation [6].

The addresses defining PH curves are carried over from [6], but with S and P prefixes to identify sweep and profile curves: these curves are defined by their end points and the polynomials $\alpha(\xi)$, $\beta(\xi)$ in Eq. 2. This data allows all relevant curve information, such as the parametric speed and arc length polynomials, to be determined [3]. All the non-PH curve data is consolidated in a single block, followed by the sweep curve and profile curve blocks. This yields a compact data structure defining the exact swept surface and relevant machining instructions. As an example, the following G05 command[6] is for a generalized conical sweep (which requires the additional CI, CF parameters).

```
G05 SF05 CI1.0 CF0.5 U30.0 RD0.25 OF0 US0.0005
SH5 SX-1.5 SY-1.5 SZ0 SA0 SB-0.534 SC-1.190
    SP1.684 SQ1.291 SR1.190
PH5 PX0 PY3.0 PZ1.5 PA-1.190 PB-1.291 PC-1.684
    PP-1.190 PQ-0.534 PR0
```

An equivalent part program using G01 tool path approximations can involve thousands or tens of thousands of blocks—see Section 5 below.

Solid models of real-world parts typically consist of many surface patches, and it is preferable to employ tool paths that span multiple adjacent patches, instead of machining each patch individually. To accommodate this, a multi-patch mode of the G05 swept surface command can be invoked with the SF08 word, and terminated with the SF09 word. The address NS indicates number of swept surface patches (of any type), defined sequentially in the order they are to be machined. The RA parameter is used to define surfaces in a canonical coordinate system, which are then transformed to the model coordinates. An example command for two surface patches (omitting most numerical values for brevity) is listed below.

```
G05 SF08 NS2 U RD OF US
SF02 RA90.0 SH5 SX SY SZ SA SB SC SP SQ SR
PH5 PX PY PZ PA PB PC PP PQ PR
SF01 RA0 SH5 SX SY SZ SA SB SC SP SQ SR
PH5 PX PY PZ PA PB PC PP PQ PR
SF09
```

4.2 Tool path planning

Unlike traditional CNC machining practice, path planning for the G05 swept surface command is performed by the real-time controller (see Figs. 6 and 7), rather than as an off-line computation. Typical path-planning strategies for surface machining employ iso-planar, iso-parametric, and iso-scallop-height tool paths [20]—each amounts to holding a specific quantity constant between successive tool path passes in the stepwise direction.

Machining with a ball-end tool always leaves "scallops" (raised ridges) of excess material between consecutive tool/surface contact paths. For finish machining, the scallop height above the desired surface is a primary indicator of the machined surface accuracy and smoothness. Iso-planar tool paths lie on families of equidistant parallel planes, and are suitable for 3-axis roughing operations—but not finish machining because they do not take account of the variable

[5]If only U is specified, it defines a constant feedrate value.

[6]For brevity, the numerical data is truncated here (and in subsequent part programs) to three decimal places. In practice, it is preferable to use a much higher degree of precision.

scallop heights they incur. In iso-scallop-height machining, an initial tool path (often a boundary curve of the surface) is chosen, from which successive parallel tool paths on the surface are calculated, so as to maintain a constant prescribed scallop height between them [17, 18, 23].

Iso-parametric tool paths on a surface $\mathbf{R}(u, v)$ correspond to continuously varying one parameter (v, say) with the other (u) held constant at successive discrete values (with an adjustable step-over Δu between tool paths). This is obviously the simplest approach, since no further computation is required to determine the tool paths. Although isoparametric curves are not, in general, intrinsic geometrical features of a parametric surface, in the context of the swept surface types considered here, they are often amenable to an intuitive geometrical interpretation, facilitating an estimation of the scallop height variation that can be used to limit [20] the step-over increment Δu.

For simplicity, this preliminary study focuses on isoparametric tool paths in swept surface machining, with conservative step-over increments to ensure the satisfaction of prescribed scallop height bounds. Because of its complexity [23], and the difficulty of ensuring complete surface coverage, the development of an iso-scallop surface interpolator is deferred to a future study. In fact, for many of the swept surfaces considered here, the isoparametric tool paths are in fact also good approximations to iso-scallop-height paths.

The swept surface interpolator employs tool paths along the v (i.e., sweep) direction, with increments Δu to define a step-over between each tool pass. The scallop height is specified through the US address (u-direction scallop) to control the finish and accuracy of the machined surface—typical values for finish machining are 0.001 to 0.01 mm (0.00005 to 0.0005 in).

The *normal curvature*[7] κ of a surface $\mathbf{R}(u, v)$, in the direction of a curve identified on it by specifying the surface parameters as functions $u(\xi)$, $v(\xi)$ of another parameter ξ, is defined by

$$\kappa = -\frac{Lu'^2 + 2Mu'v' + Nv'^2}{Eu'^2 + 2Fu'v' + Gv'^2}, \tag{8}$$

where $E = |\mathbf{R}_u|^2$, $F = \mathbf{R}_u \cdot \mathbf{R}_v$, $G = |\mathbf{R}_v|^2$, and $L = \mathbf{N} \cdot \mathbf{R}_{uu}$, $M = \mathbf{N} \cdot \mathbf{R}_{uv}$, $N = \mathbf{N} \cdot \mathbf{R}_{vv}$ are the first and second fundamental form coefficients [22]. To achieve a prescribed scallop-height h when using a tool of radius r (specified by RD), the step-over distance Δs can be estimated in terms of the normal curvature in a direction orthogonal to the tool path [17] as

$$\Delta s \approx \sqrt{8hr/(1+\kappa r)}. \tag{9}$$

[7] We adopt here the convention that κ is positive when the center of curvature lies on the "inside" of the surface, i.e., in the direction opposite to the surface normal $\mathbf{N}$.

For tool paths along the v direction, the normal curvature orthogonal to them is obtained with $u' : v' = G : -F$ in Eq. 8. The value (9) then represents a geometrical distance along the surface, locally normal to the tool path, and must be converted to a parameter step-over increment Δu in the u direction. Setting $\cos\theta = \mathbf{R}_u \cdot \mathbf{R}_v / |\mathbf{R}_u|\,|\mathbf{R}_v|$, a first-order approximation gives

$$\Delta u \approx \frac{\Delta s}{|\mathbf{R}_u|\,|\sin\theta|}. \tag{10}$$

From Eqs. 5 and 6, one can verify that the oriented-translation and oriented-involute sweeps have *orthogonal parameterizations*, i.e., $E = \mathbf{R}_u \cdot \mathbf{R}_v \equiv 0$, and satisfy $|\mathbf{R}_u| = \sigma(u)$. Consequently, $|\sin\theta| = 1$ and $|\mathbf{R}_u| =$ constant along a tool path in the v direction, so Eq. 10 defines a constant Δu. For the scaled-rotation and generalized conical sweeps, $|\mathbf{R}_u|$ and $|\sin\theta|$ are non-constant along the tool paths, but admit a simple (conservative) constant estimate for Δu that ensures satisfaction of the prescribed scallop height bound.

Typically, the approximations (9) and (10) are sufficiently accurate for a smoothly parameterized surface, a modest scallop height h, and a normal curvature orthogonal to the tool path satisfying[8] $\kappa r \ll 1$. If greater accuracy is desired, one can regard Δs as the *geodesic distance* orthogonal to the tool path, and use a higher-order expansion [17] to obtain Δu from Δs.

As seen in Fig. 7, consecutive tool paths correspond to v increasing from 0 to 1, with u held constant. On completion of a tool path, the rapid tool path re-positioning module moves the tool to the beginning of a new path through high-speed linear motions, with v re-set to 0 and u incremented based on the scallop height calculation. Finishing passes are performed with "climb" milling for better surface quality, so the tool paths all start from the same side of the surface (rather than using a zigzag pattern).

4.3 Tool path interpolation

The interpolation of a parametric curve $\mathbf{r}(\xi)$ entails computing a reference point (commanded position) parameter value ξ in each controller sampling interval Δt, using the feedrate V and parametric speed $\sigma(\xi) = |\mathbf{r}'(\xi)|$. For a general parametric curve, no simple closed-form relation between the arc length and curve parameter exists, so approximate methods must be invoked. The simplest is to use a Taylor series to determine the parameter value ξ_{k+1} at time $t = (k+1)\Delta t$ from the value ξ_k at $t = k\Delta t$, namely

$$\xi_{k+1} = \xi_k + \frac{V}{\sigma}\Delta t + \frac{V}{\sigma}\left(\frac{\sigma V' - \sigma' V}{\sigma^2}\right)\frac{(\Delta t)^2}{2} + \cdots,$$

[8] When $\kappa r \geq 1$, the tool path exhibits a concave radius of curvature smaller than the tool radius, causing gouging—a smaller tool must be used to ensure gouge-free machining.

where primes denote derivatives with respect to ξ, $\sigma' = (\mathbf{r}' \cdot \mathbf{r}'')/\sigma$, and the coefficients are evaluated at sampling interval k. For a non-constant feedrate, the derivative V' must be chain-rule converted to a derivative with respect to a physically meaningful variable—details for feedrates dependent on time, arc length, and curvature may be found in [10]. For low feedrates and high sampling frequencies, the linear term alone may be sufficiently accurate.

For PH curves, the problem of determining ξ_{k+1} from ξ_k can be reduced analytically to computing the unique root of a monotone function. Since ξ_k is already a close approximation to this root, ξ_{k+1} is obtained to machine with just a few Newton–Raphson iterations—not only for constant feedrates but also feedrates dependent on time, arc length, curvature, etc. [5, 6, 9, 25]. For some swept surface types, the isoparametric tool paths are PH curves, so the PH curve real-time interpolators can be directly invoked. In other cases, they are more complex, so the Taylor series approach should be used. Expressions for the path parametric speed $|\mathbf{R}_v(u, v)|$ for each swept surface were given in Section 3. For simplicity, only constant feedrates are employed at present, but the method can readily be extended to variable feedrates [6].

4.4 Surface normal and tool offsets

In machining with a ball-end tool, the cutter location (CL) point corresponds to displacing the cutter contact (CC) point by the tool radius, in the surface normal direction. Although tool path calculations generate CC points, the machine controller operates in terms of CL points. Hence, in each sampling interval, a tool offset must be performed to obtain a commanded CL point from a known CC point. For each swept surface type, an expression for the surface normal can be derived [8]. By implementing these expressions in the real-time interpolator, the components of the surface normal—multiplied by the tool radius (RD) and any additional specified offset (OF)—are added to the CC point coordinates to obtain the CL point coordinates.

The CC–CL conversion is usually performed off-line in a CAM program, but implementing it in the real-time controller, based upon the exact surface definition, offers significant benefits. First, path interpolation becomes *tool independent*—the CC paths are always identical, regardless of tool size or shape, and the tool offset to obtain the CL paths is performed in real time. In traditional CAM methodology, a tool change entails a complete re-build of the CL paths *and* post-processing to generate a voluminous approximate G-code part program. In the surface interpolator, the RD value could simply be the tool radius value stored in the controller, as traditionally specified modally by T and D codes: adjustment for tool wear is then effortless.

Second, performing the offset in real time through the controller allows for the machining of *offset surfaces*. Using the OF parameter,[9] offsets from the original surface can be machined by simply changing one parameter. This capability is useful for roughing and semi-finishing passes, approaching the final finish cut at zero offset. It also allows for simple tolerance adjustments over an entire surface during manufacture. The OF value could be the offset value held in conjunction with the tool radius value stored in the controller. This offset capability, in combination with the tool radius offset, parallels the G41/G42 curve offset command, but generalized to surfaces.

A further advantage of having the controller calculate the exact surface normal at each CC point is that this information can be directly exploited in 5-axis position control. Generally, the two rotational degrees of freedom are referred to either the static machine reference frame, or the dynamic local surface normal. The real-time surface interpolator allows orientation based on the exact surface normal, but this information is lost in the discretized G-code part programs used in traditional machining, which specify only CL data. Some 5-axis surface interpolation schemes are described in [4, 14].

With real-time tool offset compensation, the CC path (rather than the CL path) is interpolated. This is an important distinction, since it influences how the interpolation interacts with the physical machining process [19, 20]. The interpolator generates machine reference points that induce the machine to realize a prescribed feedrate function V (often constant). With traditional CL programming, this feedrate refers to the cutter center, but with CC surface interpolation, it refers to the cutter contact point. The latter is much preferred in machining, since the CC path is where tool–material interaction actually occurs, determining important parameters such as the material removal rate. Non-zero curvature always incurs a velocity disparity between the CL and CC paths, which is particularly pronounced with large curvature and/or tool radius. CC path interpolation thus holds promise for improvements in the precise control of material removal rate, cutting forces, etc.

5 Implementation and results

The G05 swept surface interpolator was implemented in the MDSI OpenCNC open-architecture software controller, running on an off-the-shelf PC with a modest 500 MHz processor. The controller has a 1024 Hz position sampling

[9] Since they are additive, the RD and OF values are redundant. However, it is logically preferable to distinguish between the tool radius and an additional desired surface offset.

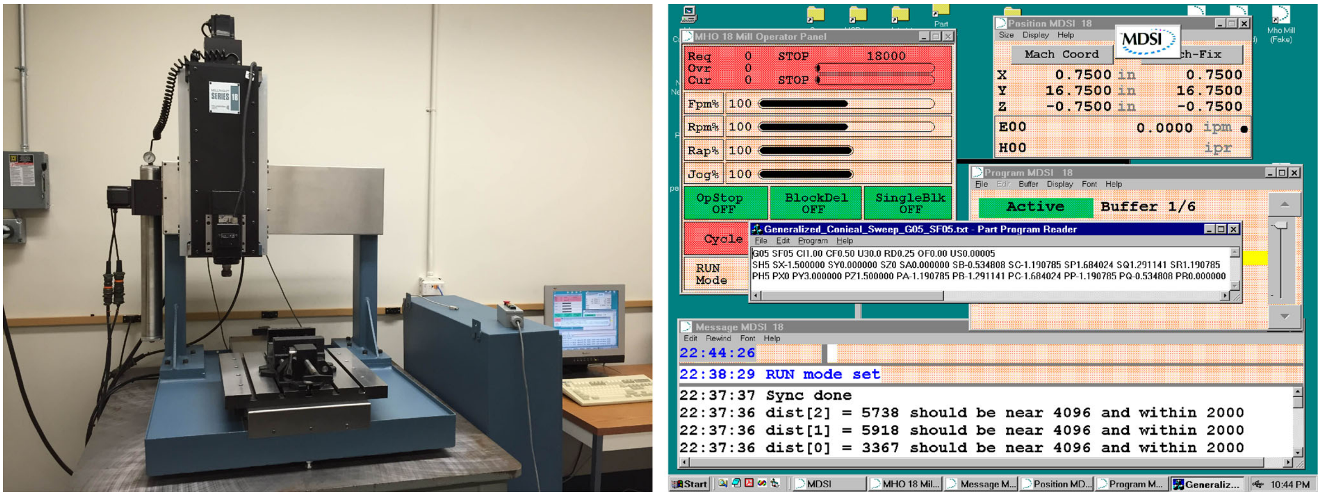


Fig. 8 *Left*: the 3-axis milling machine run by the OpenCNC controller. *Right*: the controller display of a G05 swept surface program, ready to run the machine through the custom interpolator routine integrated into OpenCNC

frequency, corresponding to a sampling interval of $\Delta t \approx 0.001\,s$, and drives a 3-axis milling machine (see Fig. 8). The surface interpolator was tested by cutting all the swept surface types described in Section 3, in both machining wax and 6061 aluminum, using the G05 command (Section 4.1). An example of the multiple-surface machining mode was also demonstrated.

5.1 Detailed example surface

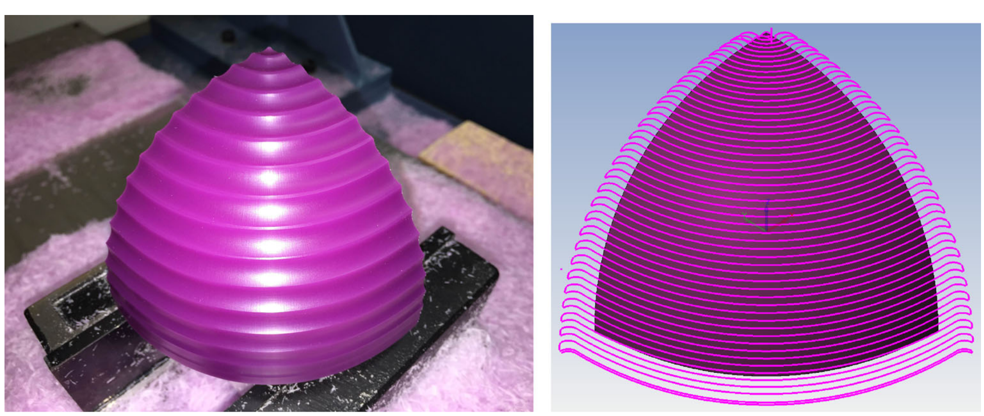
The machining of the scaled-rotation swept surface (Fig. 2) is described in detail here, and briefer descriptions for the other surface types are provided below. After fixturing and roughing using a traditional CAM-generated G-code part program, the workpiece is ready for semi-finishing, which is within the scope of the G05 command. By varying the OF and US values, the entire surface offset can be automatically machined to a prescribed scallop height. Figure 9 illustrates the outcome of a semi-finishing pass with constant scallop height generated by the G05 surface interpolator. Successively reducing OF to zero and US to the desired final scallop height provides a fast, intuitive means to generate an entire surface machining strategy without any CAM computation, post processing, or geometrical approximation.

Examples of the finish-machined surface, in both wax and aluminum, are shown in Fig. 10. A half-inch ball end mill running at 8000 rpm was used. The G05 command (in inches) for the finish machining is as follows.

```
G05 SF01 U30.0 RD0.25 OF0 US0.00005
SH5 SX-1.0 SY0 SZ0 SA0 SB-0.437 SC-0.972
    SP-1.375 SQ-1.054 SR-0.972
PH5 PX0 PY0 PZ2.95 PA-1.670 PB-1.811 PC-2.362
    PP-1.670 PQ-0.750 PR0
```

This was the *entire* information communicated to the controller, to machine the surface (the spindle speed was set manually). This command resulted in 323 tool path passes, with a largest chord length between interpolated points on the surface of only 0.000708 in, occurring at $\mathbf{R}(0.432886, 0.000110)$. The surface was finished to a very

Fig. 9 *Left*: result of a semi-finishing pass with a fixed scallop height for the scaled-rotation swept surface, machined in wax. *Right*: CAM-generated (G01) surface finishing tool paths—for clarity, the scallop height is greatly increased above the nominal 0.00005 in value



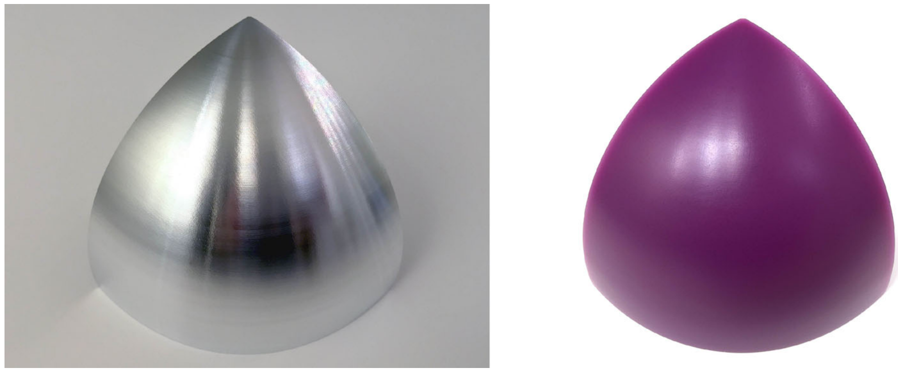


Fig. 10 The finished scaled-rotation swept surface with a very fine scallop height of 0.00005 in, as machined in aluminum 6061 (*left*) and wax (*right*)

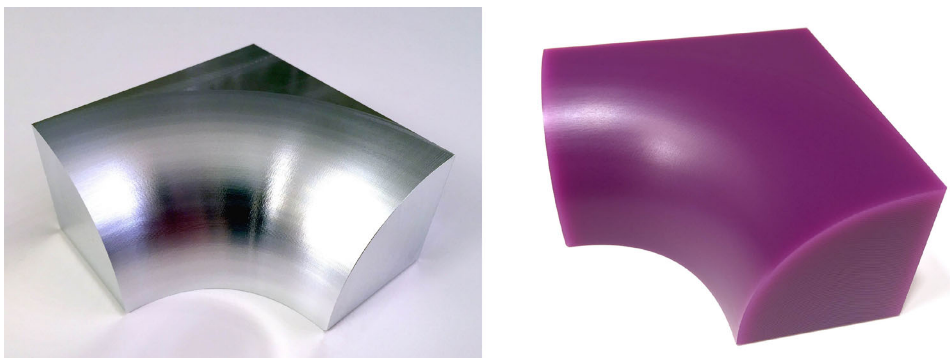


Fig. 11 The finished oriented-translation swept surface, using 240 passes with a 0.00005 in scallop height, in aluminum 6061 (*left*) and wax (*right*)

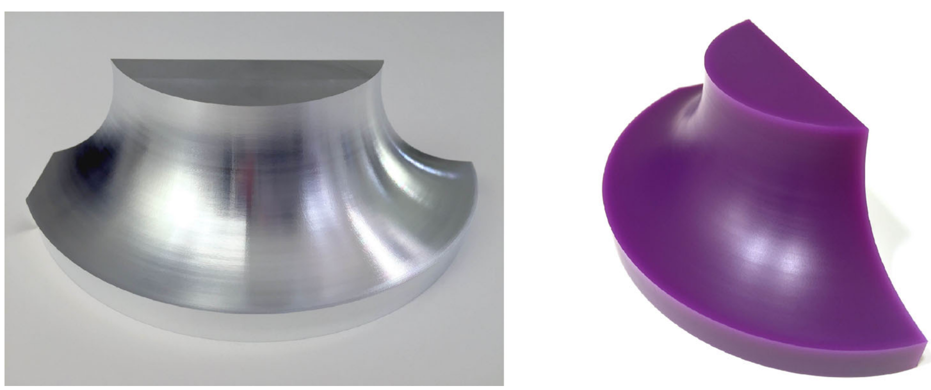


Fig. 12 The finished oriented-involute swept surface, using 240 passes with a 0.00005 in scallop height, in aluminum 6061 (*left*) and wax (*right*)

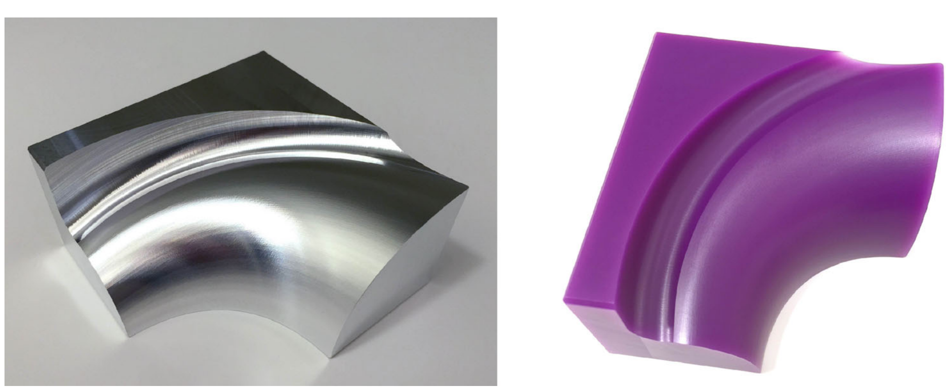


Fig. 13 The finished generalized conical swept surface, using 240 passes with a 0.00005 in scallop height, in aluminum 6061 (*left*) and wax (*right*)

Fig. 14 A composite of eight oriented-translation swept surfaces combined into a single G05 command in the SF08 multiple-surface mode (174 passes with a 0.00005 inch scallop height) in aluminum 6061 (*left*) and wax (*right*)

fine 0.00005 in scallop height which, for example, might be used for high-quality mold work before polishing.

To compare with traditional G-codes, an equivalent part program was generated in a commercial CAM program using the same scallop height (see Fig. 9—the scallop height is exaggerated for clarity). CAM programs typically allow adjustment to the tolerance of the tool path approximation, defined as the maximum allowed deviation of the discrete G01 linear segments from the exact surface. The tightness of this tolerance directly affects both the part program size and its accuracy (there is, of course, no equivalent for the G05 interpolator, since the exact surface geometry is employed).

Using the default 0.0005 in tolerance, the resulting part program based on G01 commands comprises *36,677 lines*, resulting in a *622 KB file size*. By comparison, the G05 part program consists of just *3 lines*, corresponding to a *0.242 KB file size*, which specifies the *exact* surface geometry rather than a mass of approximate tool path segments. Moreover, the program size for the traditional G-code method will scale approximately with the surface area (i.e., the square of the linear dimension) if the scallop height is held constant. The size of the G05 part program, on the other hand, is *independent* of surface size, scallop height, tool radius, and other relevant parameters—changing these parameters simply entails altering their numerical values, with no need for laborious re-calculation of tool paths or increase in data volume.

5.2 Other swept surface types

Figures 11, 12, and 13 show finish-machined results for the other swept surface types: the methodology closely follows that for the scaled-rotation sweep. Figure 14 illustrates an example of multi-patch swept surface machining—the part is modeled using eight oriented-translation swept surface patches, meeting with G^1 (i.e., tangent) continuity. Using the SF08 multi-surface mode, and rotational transformations based on the RA parameter, the machining can be performed continuously with paths that wrap around the entire part (which is more efficient than machining each patch individually). The interpolator proceeds to the next patch when the tool path on the current patch ends, and a step-over increment in the u parameter is applied after a complete pass around all the surface patches is accomplished. Continuity in the v parameter direction across contiguous patch boundaries is required, but otherwise, the patches may be generated by different sweep types, and need not necessarily form a closed loop. These examples are only preliminary demonstrations, and further functionality is necessary to achieve an implementation that is sufficiently versatile and robust for use in real-world applications.

6 Conclusion

Despite great advances in machine tool, controller, and CAM technology, the prevailing methodology for translating precise parametric surface geometry into motion commands remains primitive and cumbersome. Although data-intensive G-code tool path approximations have the advantage of universal applicability, and efforts have been made to mitigate their shortcomings in both CAM and controller algorithms, they continue to impose fundamental accuracy and efficiency limitations in precision machining of complex shapes.

A methodology was proposed to circumvent the shortcomings of G-code part programs, in the context of a versatile family of rational swept surfaces. Using Pythagorean-hodograph sweep curves, the variety of sweep operations ensuring a swept surface exactly compatible with prevailing CAD geometry representations is vastly expanded. The method is based on passing the exact surface geometry to the controller, encapsulated in a high-level procedural sweep definition. The controller can then efficiently perform all the required geometrical calculations in real time, including tool path generation subject to a prescribed scallop height. The methodology has been codified in a novel G05 command format, and its feasibility has been demonstrated through the machining of a representative family of swept surfaces using an open-architecture controller. Whereas a typical G05 part program uses just a few lines to *exactly* specify a machining operation, the equivalent approximate G01 part program may require tens or hundreds of thousands of blocks.

Many advantageous aspects of the swept surface machining methodology deserve further study. These include the exploitation of tool paths specified by cutter contact (CC) rather than cutter location (CL) data, the use of exact real-time surface normal data to accommodate CC–CL offset compensations for different tool geometries, suppression of material removal rate fluctuations through variable feedrates, and the utilization of multi-patch rational swept surfaces as a basic tool in mold and die design applications.

References

1. Cheng C-W, Tseng W-P (2006) Design and implementation of a real–time NURBS surface interpolator. Int J Adv Manuf Tech 30:98–104
2. FANUC Series 30i–MODEL A, Operator's Manual (2004), FANUC Co.
3. Farouki RT (2008) Pythagorean–hodograph curves: algebra and geometry inseparable. Springer, Berlin
4. Farouki RT, Han CY, Li S (2014) Inverse kinematics for optimal tool orientation control in 5–axis CNC machining. Comput Aided Geom Design 31:13–26
5. Farouki RT, Manjunathaiah J, Nicholas D, Yuan G-F, Jee S (1998) Variable feedrate CNC interpolators for constant material removal rates along Pythagorean–hodograph curves. Comput Aided Design 30:631–640
6. Farouki RT, Manjunathaiah J, Yuan G-F (1999) G codes for the specification of Pythagorean–hodograph tool paths and associated feedrate functions on open–architecture CNC machines. Int J Mach Tools Manuf 39:123–142
7. Farouki RT, Neff CA (1990) Analytic properties of plane offset curves. Comput Aided Geom Design 7:83–99
8. Farouki RT, Nittler KM (2015) Rational swept surface constructions based on differential and integral sweep curve properties. Comput Aided Geom Design 33:1–16
9. Farouki RT, Shah S (1996) Real–time CNC interpolators for Pythagorean–hodograph curves. Comput Aided Geom Design 13:583–600
10. Farouki RT, Tsai Y-F (2001) Exact Taylor series coefficients for variable–feedrate CNC curve interpolators. Comput Aided Design 33:155–165
11. Hardwick M, Zhao YF, Proctor FM, Nassehi A, Xu X, Venkatesh S, Odendahl D, Xu L, Hedlind M, Lundgren M, Maggiano L, Loffredo D, Fritz J, Olsson B, Garrido J, Brail A (2013) A roadmap for STEP–NC–enabled interoperable manufacturing. Int J Adv Manuf Tech 68:1023–1037
12. Hinds JK, Kuan LP (1978) Surfaces defined by curve transformations. In: Proceedings of the 15th numerical control society annual meeting & technical conference, pp 325–340
13. Koren Y, Lin R-S (1995) Five–axis surface interpolators. CIRP Ann 44:379–382
14. Liang H, Li X (2009) A 5–axis milling system based on a new G code for NURBS surface. In: IEEE international conference on intelligent computing and intelligent systems (ICIS), vol 2009, pp 600–606
15. Liang H, Li X (2013) Five–axis STEP–NC controller for machining of surfaces. Int J Adv Manuf Tech 68:2791–2800
16. Lin R-S (2000) Real–time surface interpolator for 3–D parametric surface machining on 3–axis machine tools. Int J Mach Tools Manuf 40:1513–1526
17. Lin R-S, Koren Y (1996) Efficient tool–path planning for machining free–form surfaces. ASME J Manuf Sci Eng 118:20–28
18. Lin Z, Fu J, Shen H, Gan W (2014) A generic uniform scallop tool path generation method for five–axis machining of freeform surface. Comput Aided Design 56:120–132
19. Lo C-C (1999) Real–time generation and control of cutter path for 5–axis CNC machining. Int J Mach Tools Manuf 39:471–488
20. Lo C-C (2000) CNC machine tool surface interpolator for ball–end milling of free–form surfaces. Int J Mach Tools Manuf 40:307–326
21. SINUMERIK 840D/840Di/810D Advanced Programming Guide (2002), Siemens Corporation
22. Struik DJ (1961) Lectures on classical differential geometry. Dover Publications (reprint, New York
23. Suresh K, Yang DCH (1994) Constant scallop–height machining of free–form surfaces. ASME J Manuf Sci Eng 116:253–59
24. Tsai M-C, Cheng C-W, Cheng M-Y (2003) A real–time NURBS surface interpolator for precision three–axis CNC machining. Int J Mach Tools Manuf 43:1217–1227
25. Tsai Y-F, Farouki RT, Feldman B (2001) Performance analysis of CNC interpolators for time–dependent feedrates along PH curves. Comput Aided Geom Design 18:245–265
26. Wang Y, Liu H, Yu S (2012) Curvature–based real–time NURBS surface interpolator with look–ahead ACC/DEC control. Math Comput Sci 6:315–326