PED-Vol. 64, Manufacturing Science and Engineering
ASME 1993

CNC INTERPOLATORS: ALGORITHMS AND ANALYSIS

Y. Koren and C. C. Lo
Department of Mechanical Engineering
University of Michigan
Ann Arbor, Michigan

M. Shpitalni
Laboratory for Computer Graphics and CAD
Faculty of Mechanical Engineering
Technion
Haifa, Israel

ABSTRACT

CAD systems today interpolate general curves by dividing each curve into many straight-line segments which are downloaded to the CNC. Determining the number of lines to be transferred from the CAD to the CNC poses a conflict between the desired precision of the part and the feedrate fidelity. The current method results in severe variations in the feedrate, leading, in turn, to variations in the surface smoothness and a substantial increase in machining time. These problems are caused by the acceleration/deceleration at the ends of each segment. Moreover, the problems are inherent in the CNC interpolator, as is thoroughly discussed in this paper. These problems can be solved by the development of curve interpolation algorithms for CNC. In this paper, a real-time interpolation algorithm for curves presented in their parametric forms is proposed and compared with the existing CAD interpolators. Analysis shows that with this new interpolator, a constant feed is maintained along the cut and the machining time is as expected. In addition, the amount of geometric information transferred from the CAD system to the CNC is reduced by orders of magnitude. Moreover, the contour errors caused by the new interpolator are much smaller than those caused by conventional CAD interpolators.

Keywords: CNC, CAD, Interpolators, Curves.

INTRODUCTION

Most computer aided design (CAD) systems provide the designer with tools for defining two- and three-dimensional curves and surfaces. By contrast, conventional computerized numerically controlled (CNC) machines generally support only the functions of straight line and circular interpolations [e.g., Koren, 1983]. This gap between the well developed theory of representing curves and surfaces in CAD systems and the limited capabilities offered by the numerical control interpolators imposes severe difficulties on rapid and accurate machining of surfaces and curves, indicating an obvious need for a general curve interpolator for CNC machines.

In this paper, various possibilities for implementing curve interpolators on CNC machines are analyzed, and a new scheme is suggested. First, the current method used for machining of curves is discussed, and the difficulties associated with it are explored. Then, the need for a real-time interpolator for general curves is presented. Finally, a new scheme is proposed for real-time interpolators for general curves presented in parametric form.

Figure 1 shows the current method for machining a part. CAD systems are used to define the geometry of the part, and the CNC must then drive the machine tool to machine this geometry. The part geometry is transferred to the CNC by means of a part program consisting mainly of motion commands. These motion commands must be translated in real time by the CNC interpolator into a special form before being sent to the control loops for execution (i.e., driving the machine tool).

The motion commands must fit the interpolator capabilities so that the interpolator can translate them in real time. That is, a typical CNC can process only straight line and circular arc motion commands. Thus, in order to drive the machine tool along a curve, the curve must be broken into a set of line segments which approximates the curve to a desired accuracy (i.e., tolerance). The segmentation of the curve into line segments is performed in the CAD system by off-line calculations.

In summary, the current procedure for machining curves is as follows:

(a) The designer defines a shape by a set of geometrical entities. An entity may be a line, an arc, or a curve.
(b) If the entity is a line or a circular arc, it is transferred directly as such to the CNC by the part program.
(c) If the entity to be machined is a curve, it is first divided into line segments in the CAD system. The resulting line segments are then transferred to the part

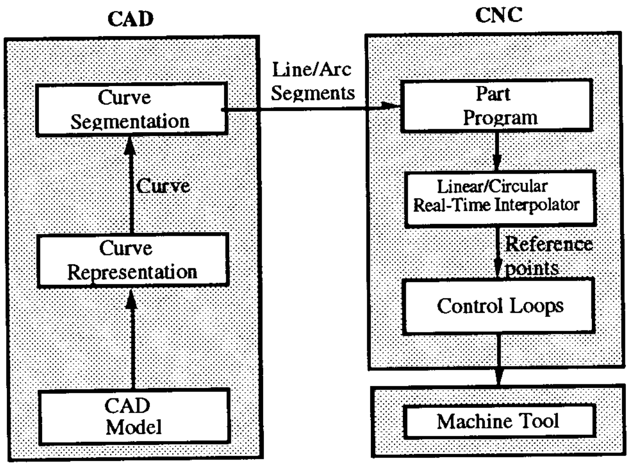


FIG. 1: CURRENT METHOD FOR MACHINING OF CURVES.

program. Their original curve is not even known to the CNC.

OPTIMAL SEGMENTATION OF CURVES

Inherent in the current method for machining of curves is a conflict regarding the number of line segments into which a curve should be divided by the CAD system. On the one hand, the number of segments should be *maximized* for two reasons:

(1) To better approximate the curve and reduce the contour error, and

(2) To minimize the effect of segmentation which causes discontinuities in the first derivatives along the path. These discontinuities, in turn, lead to deterioration of the smoothness of curves and surfaces and necessitate additional treatment (e.g., polishing).

The difficulty in this maximization, however, is the huge number (millions) of segments required for machining of surfaces. Therefore, on the other hand, the number of segments should be kept to a *minimum* for the following reasons:

(1) Each segment is treated by the CNC as an individual line. If many short segments are transferred, the tool may never reach the desired feedrate due to the automatic acceleration and deceleration applied at the beginning and end of each segment (feature of the control). The result is that the feedrate along the curve is not constant which, in turn, deteriorates the surface finish (in milling) or part dimension (in laser cutting). In addition, the machining time is increased because the mean feedrate is less than the desired feedrate.

(2) A similar effect is caused by interpolators of the Reference-Word type (see below). With these interpolators, a large number of segments increases the feedrate variations, reduces the average feedrate, and increases the machining time.

(3) The CNC's memory is very small (and very expensive) compared to the number of segments to be stored for a part with complex surfaces.

(4) The communication load between the CAD and the CNC, that may cause consequent errors, should be reduced.

In practice, the approach of minimizing the number of segments has been adopted by industry due to memory and communication constraints. The minimum number of segments is dictated by the allowed tolerance and depends on the curvature and length of the curve. However, the resulting discontinuities along the curve necessitate additional treatment. Note that even with the minimization approach a huge number of segments (may reach millions) is still needed in order to maintain reasonable tolerance (e.g., 10 μm). Thus, the current method is not adequate for machining of curves and surfaces.

Consequently, a new CAD/CNC procedure should be developed which will enable high and constant feedrate along the curve. This concept is shown in Figure 2. In the proposed approach the CAD segmentation algorithm has been removed from the CAD system. Instead, a new interpolator which can interpolate general curves in real time has been added to the CNC. This enables the CAD system to transfer to the CNC only information about the curve; the CNC then interpolates it using the real-time curve interpolator.

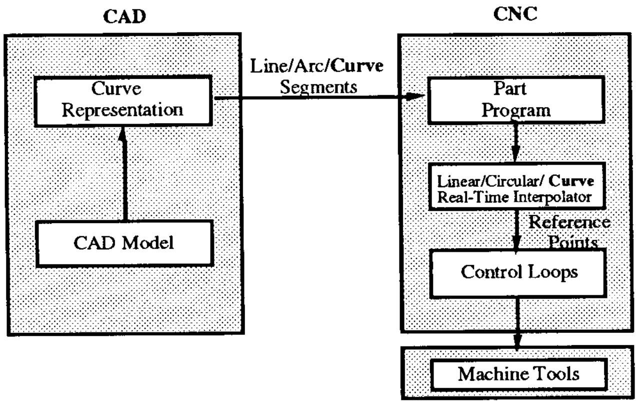


FIG. 2: SUGGESTED METHOD FOR MACHINING OF CURVES.

OFF-LINE CURVE INTERPOLATORS

One method for describing curves which treats the x and y axes symmetrically is the parametric form

$$x = x(u) \qquad (1)$$
$$y = y(u)$$

where u is an arbitrary parameter usually $0 \le u \le 1$. Most CAD/CAM systems use parametric forms to represent curves. The parametric form is very convenient for controlling multi-axis machine tools, where each axis is individually driven. Furthermore, the parametric form enables direct calculation of x and y as functions of u, and the extension from 2-D to 3-D is straightforward.

The selection of u is critical to the proper operation of an interpolator that calculates x(u) and y(u) in real time. Three conditions must be satisfied in the selection of u:

1. The parameter u must be independent of the geometrical parameters of the curve.
2. The representation of x(u) and y(u) must be explicit for u to allow direct, fast calculation of x(u) and y(u).
3. Successive u's must divide the curve uniformly, i.e., into equal segments Δs. Since each segment is machined during one sampling period T, this uniform subdivision of the curve guarantees cutting at a constant feedrate (i.e., velocity) V, where $V=\Delta s/T$.

While the first two requirements are usually satisfied, the third condition is satisfied only if u is a linear function of s, where s is the curve arc length, and then the position is represented as x(s) and y(s). This condition is easily satisfied for straight lines and circles, as shown below:

For a straight line given by its two endpoints (x_1, y_1) and (x_2, y_2):

$$x(s) = x_1 + \frac{(x_2 - x_1)s}{\sqrt{(x_2-x_1)^2+(y_2-y_1)^2}}$$

$$y(s) = y_1 + \frac{(y_2 - y_1)s}{\sqrt{(x_2-x_1)^2+(y_2-y_1)^2}}$$

for $0 \le s \le S\left(=\sqrt{(x_2-x_1)^2+(y_2-y_1)^2}\right)$.

For a circle with center at (x_1, y_1) and radius of a:

$$x(s) = x_1 + a\cos(s/a)$$
$$y(s) = y_1 + a\sin(s/a)$$

for $0 \le s \le \frac{\pi a}{2}$. However, satisfying the third condition in the general case is more complex.

In order to determine x(s) and y(s) in the general case, the intrinsic equations of the curve must be solved [Faux & Pratt, 1979]. The intrinsic equation of a curve defines the relationship between its arc length s and the angle φ between the tangent at point P and the X-axis. The curvature at point P is obtained from the intrinsic equation by the formula:

$$\kappa = \frac{d\varphi}{ds} \tag{2}$$

where $\kappa=1/\rho$ and ρ is the radius of curvature at point P.

Alternatively, the curve may be described parametrically in terms of the arc length by the equations x=x(s) and y=y(s). The functions x(s) and y(s) are then related to φ by the equations

$$\frac{dx}{ds} = \cos\varphi \;;\quad \frac{dy}{ds} = \sin\varphi \tag{3}$$

If we differentiate these equations with respect to s, and substitute κ for $\frac{d\varphi}{ds}$, $\frac{dx}{ds}$ for $\cos\varphi$, and $\frac{dy}{ds}$ for $\sin\varphi$, we obtain the simultaneous differential equations:

$$\begin{aligned} \frac{d^2x}{ds^2} + \kappa(s)\frac{dy}{ds} = 0 \\ \frac{d^2y}{ds^2} - \kappa(s)\frac{dx}{ds} = 0 \end{aligned} \tag{4}$$

These two second-order equations represent an interpolation algorithm and can in principle be solved to determine x(s) and y(s) for any given curvature function $\kappa(s)$. Appropriate numerical procedures have been described by Adams [1975)]. For a general curve, however, these are time consuming calculations that fit only off-line applications. Other solutions should be sought.

The common approach in off-line interpolation is to divide the curve into equal increments Δu, to determine the corresponding $[x(u_k), y(u_k)]$ and to drive the tool along a straight line between $[x(u_{k-1}), y(u_{k-1})]$ and $[x(u_k), y(u_k)]$. Obviously, a larger number of segments results in a better tracking of the curve. However, in the previous section, we claimed that with reference-word interpolators, dividing a curve into a large number of segments reduces the average feedrate, increases the feedrate variations, and increases the machining time.

Although this observation is extremely important, it was never discussed in the literature. Consequently, CAD/CNC programmers are not aware of it, thus causing many problems in production of parts with complex surfaces. To facilitate the explanation of this point, let us use an example of a CNC system with a resolution of BLU=0.01 mm and interpolation sampling period of T=0.01 sec. A straight line (which simulates here a general curve) has to be machined in one axis at a feedrate of V=4 mm/sec (i.e., 4 BLU/sampling-period).

When the CNC linear interpolator receives the input, in terms of the length of the line l and the feedrate V=4, it starts to issue control commands of "move 4 BLUs" at every sampling period T. The accumulated velocity commands is the position reference. The last command in the line might be shorter, to complete the production of exactly l units. For a line of 22 BLUs, for example, the interpolator produces the position reference shown in Fig. 3a. The last incremental position is smaller. Figure 3b shows the corresponding velocity command.

During the first five sampling periods, the velocity command is V=4BLUs/0.01 sec=4 mm/sec, but it is only 2 mm/sec in the last period. The smaller velocity in the last period is typical.

Now let us examine the effect of curve segmentation for a straight line of l = 55 BLUs. The selection of a short line enables us to draw the results. We analyze four cases:

(a) The line is produced as one segment of 55 BLUs.
(b) The line is divided into three segments, two of 18 BLUs and one of 19 BLUs (2 x 18 +19 = 55).
(c) The line is divided into six segments, five of 9 BLUs and one of 10 BLUs (5 x 9 + 10 = 55).
(d) The line is divided into 11 segments, each of 5 BLUs.

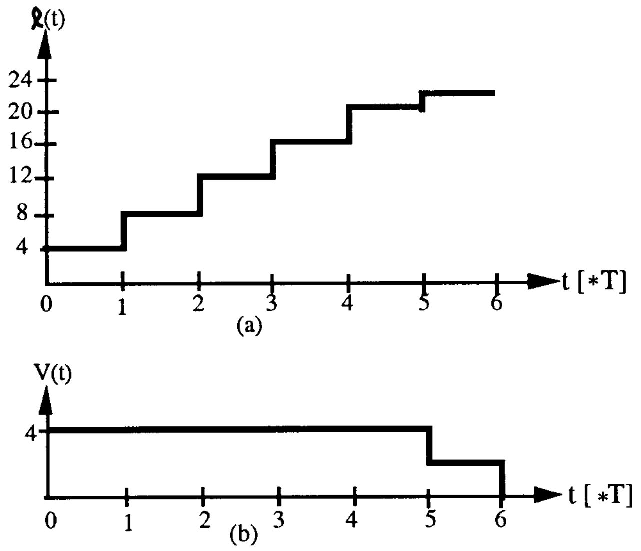


FIG. 3: (A) POSITION COMMAND ISSUED BY LINEAR INTERPOLATOR, AND (B) THE CONSEQUENT VELOCITY COMMAND.

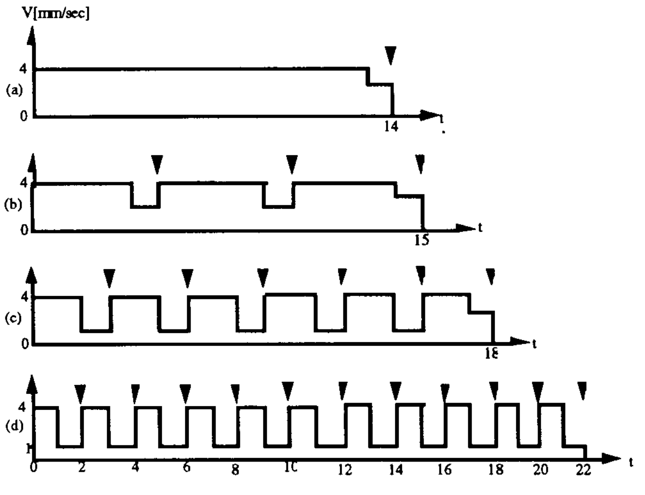


FIG. 4: VELOCITY COMMANDS ISSUED BY THE CNC INTERPOLATOR VERSUS NUMBER OF SAMPLING PERIODS.

The corresponding velocity commands are depicted in Fig. 4. If the length of a segment is not integer multiples of 4, its last incremental position is smaller than 4, which means that the velocity command is reduced during the last sampling period. Figure 4 shows that a division of a line into a larger number of segments increases the feedrate variations and reduces the average feedrate, which, in turn, increases the machining time. The machining time of the 55 BLU, as shown in Fig. 4, is 14T if the whole line is treated as a single segment (Case a). The machining time increases to 15T if the line is divided into 3 segments (Case b); to 18T if the line is divided into 6 segments (c); and, to 22T if 11 segments are used. The situation is even more complex when several axes are involved in the motion and especially for general curves.

To further clarify the point, assume that the curve

$$\begin{cases} x = 11.9u^3 - 29.8u^2 + 32.9u + 5.0 \\ y = 47.6u^3 - 41.7u^2 + 16.55u + 2.5 \end{cases} \quad 0 \le u \le 1$$

is stored in a CAD system and has to be machined on a CNC at a velocity of V = 1.2 m/min = 20 mm/sec. Also assume that the sampling time of the CNC is 0.01 sec. In the CAD system, the parameter u is incremented uniformly in equal increments Δu -- this is the segmentation of the curve. The more segments in the curve, the smaller the contour error, but larger deviations from the average feedrate are expected. The results are shown in Fig. 5.

As a consequence, we see that very fine segmentation causes severe problems in maintaining a uniform and desired V. This and the other reasons discussed in Sec. 2 above show the importance of a reliable real-time CNC interpolator.

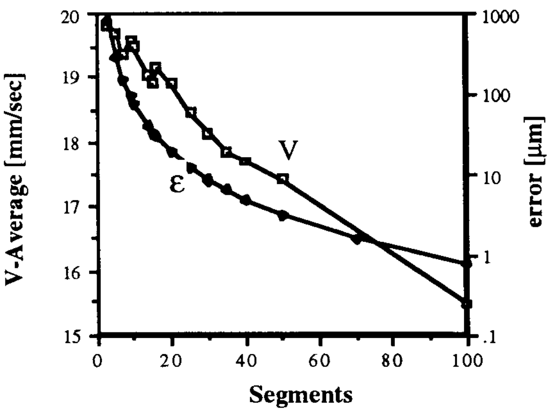


FIG. 5: THE EFFECT OF SEGMENTATION ON THE CONTOUR ERROR AND AVERAGE FEEDRATE.

TYPES OF CNC INTERPOLATORS

Two main kinds of interpolators exist in CNC systems.

(1) Reference-Pulse Interpolators which continuously calculate the increment to the next reference point at the resolution level (pulse mode), e.g., the DDA (Digital Differential Analyzer) method [Koren, 1976; Koren & Masory, 1981, Papaioannou, 1979].

(2) Reference-Word Interpolators which calculate the next reference point based on constant sampling time T [e.g., Masory & Koren, 1982].

The selected interpolation method depends on the control method. The proposed interpolator in this paper is of the more common method -- the reference-word interpolator.

Several researchers have developed real-time interpolators for curve generation. Koren [1976] proposed an interpolator for standard parabolic curves; Sata et. al. [1981] developed a real-time interpolator for cubic Bezier curves which can, in fact, interpolate any cubic function. This interpolator can of course interpolate general parabolic curves by using only a 2nd order Bezier curve. Makino [1988, 1991] developed special interpolators for high-speed machines. Stadelmann [1989] proposed a high-order interpolator for complex spatial geometry that requires a given velocity profile. In this interpolator the transition between two successive segments is continuous. However, this interpolator does not guarantee that all interpolated points lie on the curve. Recently Chou and Yang [1991, 1992] proposed a new interpolator for curves represented in their parametric forms. Their interpolator indeed interpolates the curves correctly but it does not fit real-time applications. They solved the equation that relates the time t to parameter u, namely t=f(u), while for real time realization the parameter u should be related to time, namely u=g(t). Solving u=g(t) requires a mathematical development that is given in this paper.

Proposed interpolation schemes may be differentiated by the method used to represent the curve. Two representation methods exist with corresponding interpolation schemes. The first scheme deals with the implicitly defined curve f(x,y)=0 [Lo, 1992]; and, the second scheme deals with the parametric curves $\bar{r}(u)=x(u)\bar{i}+y(u)\bar{j}$. (The extension to 3-D is also discussed in this paper.)

Note that in the particular case where y can be expressed explicitly as a function of x, namely, y=f(x), the situation is equivalent to the case of parametric representation. In these cases, the function can be expressed as:

x(u)=u
y(u)=f(u)

In this paper only interpolators for curves presented in their parametric form are discussed. The straightforward approach for real-time interpolators might be the one utilized in CAD systems: *Increment the parameter u uniformly, namely in equal small increments Δu, and calculate the corresponding x_{k+1} and y_{k+1} at each sampling period T.*

The approach, however, has two main drawbacks:

(1) The length of the steps Δs_k, where $\Delta s_k=\sqrt{(x_{k+1}-x_k)^2+(y_{k+1}-y_k)^2}$, is not equal, but the tool traverses them in equal time intervals T. Consequently, the feedrate along the curve is not constant.

(2) The optimal size of the increment Δu is not known. If Δu is too small, the resultant Δs_k's are too small as well, and the system slows down. If Δu is too big, the resultant Δs_k's are too big and the position accuracy is not maintained.

As a consequence, a real-time interpolator cannot be based on a uniform segmentation of the curve according to u. CAD systems, however, are based on equal segmentation of u to represent a curve by lines. But, in CAD systems, the lines are relatively longer and a velocity command is issued for each line, whereas in real-time interpolators the velocity is obtained automatically by $V_k=\Delta s_k/T$.

This situation motivated the development of a new real-time interpolation method for general curves expressed in their parametric form. This method requires only small computation time and therefore fits real-time applications.

The task of the real-time interpolator for a general curve can now be specified.

Given: (1) *a curve* $\bar{r}(u)$,
(2) *the desired feedrate V[mm/sec] along the curve,*
(3) *the sampling time interval T[sec], and*
(4) *the current reference point on the curve* $R_k(x_k,y_k)$,

find the next reference point to which the tool should be driven in T seconds while maintaining a constant feedrate V.

This implies that the next reference point $R_{k+1}(x_{k+1},y_{k+1})$ must lie on the desired curve and must obey $\|R_{k+1}-R_k\|=VT$.

In the following section, a new approach for real-time reference-word interpolators is proposed. Such an interpolator fits the case of parametric representation and is capable of dealing with general curves. We assume that the given curve is the curve along which the center of the tool must be driven. The new real-time interpolator is compared with the conventional interpolator.

INTERPOLATOR BASED ON UNIFORM-LENGTH SEGMENTS

The key idea for real-time interpolators of curves represented in parametric form is that the segmentation should be based on curve segments of equal length rather than on equal Δu's. Accordingly, the proposed method determines successive values of u such that the curve segments Δs_k (machined at each sampling period T) are constants, which, in turn, guarantees a constant feedrate (i.e., velocity).

The feedrate V(u) along the curve is defined by

$$V(u)=\frac{ds}{dt}=\left(\frac{ds}{du}\right)\left(\frac{du}{dt}\right) \tag{5}$$

or

$$\frac{du}{dt}=\frac{V}{ds/du} \tag{6}$$

where

$$\frac{ds}{du}=\sqrt{(x')^2+(y')^2+(z')^2} \tag{7}$$

and

$$x'=\frac{dx}{du}\quad ;\ y'=\frac{dy}{du}\quad ;\ z'=\frac{dz}{du}$$

Substituting Eq. (7) into (6) yields

$$\boxed{\frac{du}{dt}=\frac{V}{\sqrt{(x')^2+(y')^2+(z')^2}}} \tag{8}$$

A solution of Eq. (8) that is performed at short computation time is the heart of a real-time interpolator for curves given in parametric forms [Lo, 1992; Huang and Yang, 1992; Huang, 1992]. The solution of Eq. (8) for a constant V gives the required u(t). However, because the solution of Eq. (8) is difficult in the general case, we may use a recursive solution based on Taylor's expansion around t=kT:

$$u_{k+1}=u_k+T\dot{u}_k+\left(T^2/2\right)\ddot{u}_k + \text{higher order terms} \tag{9}$$

where $\dot{u}$ denotes $\frac{du}{dt}$ and $\ddot{u}$ denotes $\frac{d^2u}{dt^2}$.

If T is very small and the curve does not have small radii of curvature, even a first-order approximation is adequate:

$$u_{k+1}=u_k+T\dot{u}_k \tag{10}$$

This equation usually holds when

$$T<<2\left|\frac{\dot{u}_k}{\ddot{u}_k}\right| \tag{11}$$

Substituting Eq. (8) in Eq. (10) yields the equation of the proposed interpolator:

$$\boxed{u_{k+1}=u_k+\frac{VT}{\sqrt{x_k'^2+y_k'^2+z_k'^2}}} \tag{12}$$

In cases where Eq. (11) is not satisfied, we may use two terms of Taylor's expansion

$$u_{k+1}=u_k+T\dot{u}_k+\left(T^2/2\right)\ddot{u}_k \tag{13}$$

Equation (13) holds for small T, defined by

$$T<<3\left|\frac{\ddot{u}_k}{\dddot{u}_k}\right| \tag{14}$$

The expression for $\ddot{u}$ may be derived from Eq. (8)

$$\ddot{u}=\frac{d^2u}{dt^2}=-\frac{V^2\left(\frac{d^2s}{du^2}\right)}{\left(\frac{ds}{du}\right)^3} \tag{15}$$

where ds/du is given in Eq. (7), and differentiating Eq. (7) yields

$$\frac{d^2s}{du^2}=\frac{x'x''+y'y''+z'z''}{\sqrt{x'^2+y'^2+z'^2}} \tag{16}$$

where

$$x''=\frac{d^2x}{du^2}\ ;\ y''=\frac{d^2y}{du^2} \text{ and } z''=\frac{d^2z}{du^2}$$

The first derivative of u is given in Eq. (8)

$$\dot{u}_k=\frac{V}{\sqrt{x_k'^2+y_k'^2+z_k'^2}} \tag{17}$$

and the second derivative is given by substituting Eqs. (7) and (16) into (15).

$$\ddot{u}_k=\frac{-V^2(x_k'x_k''+y_k'y_k''+z_k'z_k'')}{\left(x_k'^2+y_k'^2+z_k'^2\right)^2} \tag{18}$$

Substituting Eqs. (17) and (18) into (13) yields [Lo, 1992; Huang, 1992]

$$\boxed{\begin{aligned}u_{k+1}=u_k&+\frac{VT}{\sqrt{x_k'^2+y_k'^2+z_k'^2}}-\\&\frac{(VT)^2(x_k'x_k''+y_k'y_k''+z_k'z_k'')}{2\left(x_k'^2+y_k'^2+z_k'^2\right)^2}\end{aligned}} \tag{19}$$

The explicit expressions of the first and second derivatives of x and y with respect to u are calculated and substituted into Eq. (19). The new proposed interpolator is based on calculating the new value u_{k+1} from Eq. (19), and using it in the next iteration to calculate $x(u_{k+1})$, $y(u_{k+1})$ and $z(u_{k+1})$.

In order to check the condition (11) at which the approximation (10) is valid, Eqs. (17) and (18) are substituted into (11), yielding

$$VT<<\frac{2\left[x'^2+y'^2+z'^2\right]^{3/2}}{\left|x'x''+y'y''+z'z''\right|} \tag{20}$$

If condition (20) is not satisfied, and Eq. (12) is used in the interpolation process, a feedrate error exists. Note, however, that the interpolated points are always on the curve. An interpolator based on Eq. (19) yields negligible feedrate errors for all practical cases.

It is important to show how this interpolator behaves if applied to a circular arc in the X-Y plane.

$$x(u) = a \cos u$$

$$y(u) = a \sin u$$

for $0 \le u \le \frac{\pi}{2}$.

Differentiating these equations to obtain

$$\frac{dx}{du} = -a \sin u$$

$$\frac{dy}{du} = a \cos u$$

and substituting them in Eq. (11) results in:

$$u_{k+1} = u_k + \frac{VT}{\sqrt{a^2 \sin^2 u_k + a^2 \cos^2 u_k}} = u_k + \frac{VT}{a}$$

namely, the changes in u_k are independent of k, they are constant and equal to VT/a (where a is the radius of the arc). This means that a perfect circle is produced.

To evaluate the proposed interpolator, 2-D and 3-D cubic polynomial curves were simulated and compared with the conventional method, which approximates the curve by line segments for which Δu=constant. The evaluation is given below.

Example 1

The 2-D cubic curve

$$\begin{cases} x = -140u^3 + 90u^2 + 90u \\ y = -90u^2 + 90u \end{cases} \qquad 0 \le u \le 1$$

was simulated with V = 1.5 m/min, T = 0.01 sec.

The contour shape of this 2-D curve is shown in Fig. 6. The proposed interpolator was simulated and results in very small contour errors compared to conventional CAD interpolators, as shown in Fig. 7 and in Table 1. The variations in the velocity commands that sent to the control loops for the proposed interpolator are depicted in Fig. 8a for Eq. 12 and Fig. 8b for Eq. 19. Both are negligible for most applications. Adding the second derivative of u to the approximation results in slightly smaller contour errors and smaller velocity variations. The calculation time, however, is increased, which increases the lower bound on T. Therefore, for this example, the approximation based on Eq. 12 is adequate.

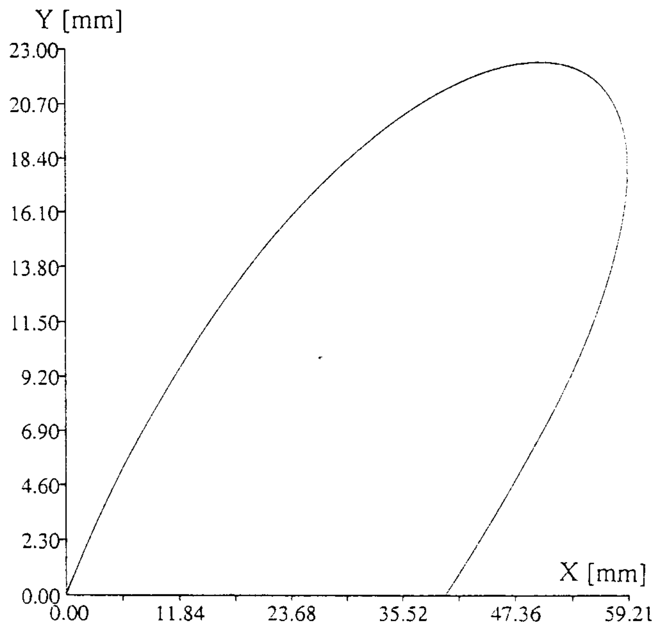


FIG. 6: A 2-D CUBIC POLYNOMIAL CURVE.

The segmentation in the conventional CAD interpolator, based on small equal Δu (e.g., Δu=0.02 for 50 segments), results in correspondingly short Δs segments (e.g., average of 1.9 mm). Because of the method by which linear interpolators operate, the short segments cause large feedrate errors, as shown in Fig. 9. As expected and as shown in Table 1, increasing the number of segments results in smaller contour errors, but increases the average deviation from the programmed feedrate (V=25 mm/sec in this example). This simulation does not include the acceleration/deceleration effect that even deteriorates the situation in off-line CAD segmentation.

In other CAD systems the approximation of the parametric curves by a set of line segments is based on assigning constant tolerance, rather than constant u [Loney & Ozsoy, 1987]. The curve in these cases is approximated by a smaller number of lines, but the fundamental problem inherent to off-line interpolation will exist. However, we made the comparison to the method utilizing constant Δu since it can be a candidate for real-time interpolators, whereas the constant tolerance method has high computation load and cannot be implemented in real time.

TABLE 1: INTERPOLATION OF A 2D CUBIC SPLINE.

Proposed Interpolator:			
Equation used:	Eq. 12	Eq. 19	
$\varepsilon_{max}[\mu m]$	2.6	2.5	
$\delta V_{av}/V$	0.2%	0.01%	
$\delta V_{max}/V$	1.7%	0.07%	
Conventional Interpolator:			
# of line segments	30	50	70
$\varepsilon_{max}[\mu m]$	61	22.3	11.2
$\delta V_{av}/V$	3.8%	6.6%	9.2%
$\delta V_{max}/V$	Arbitrary, and may reach up to 98%		

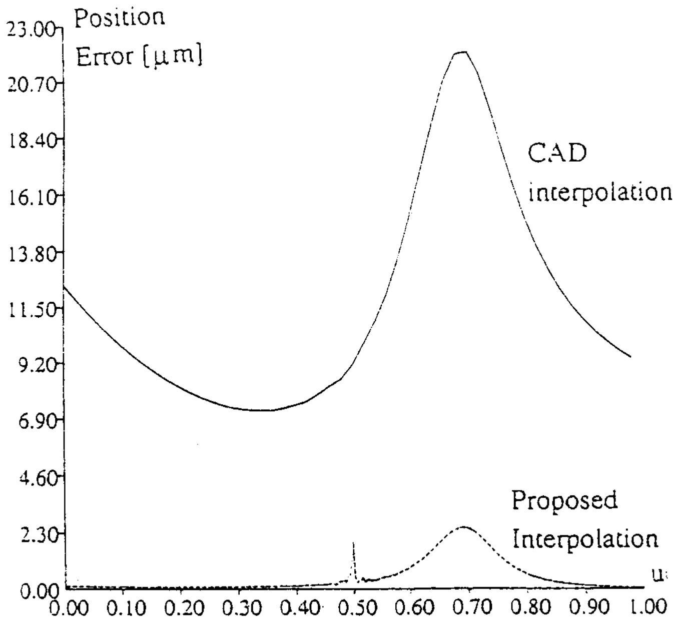
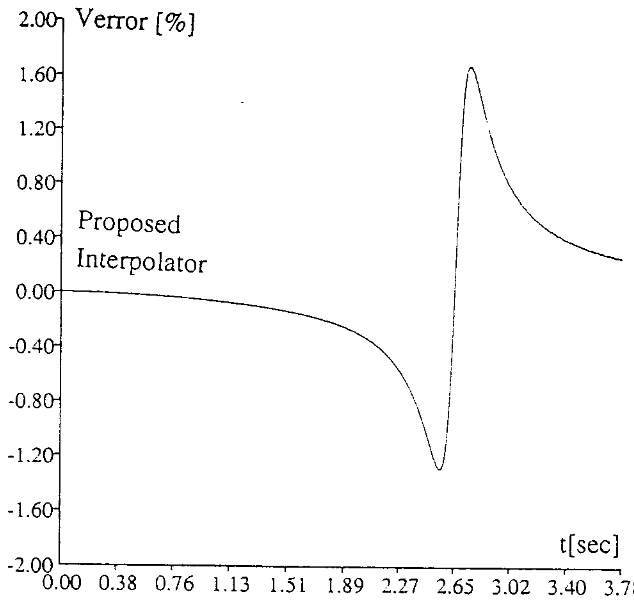


FIG. 7: CONTOUR ERRORS CAUSED BY THE PROPOSED AND CONVENTIONAL INTERPOLATORS.



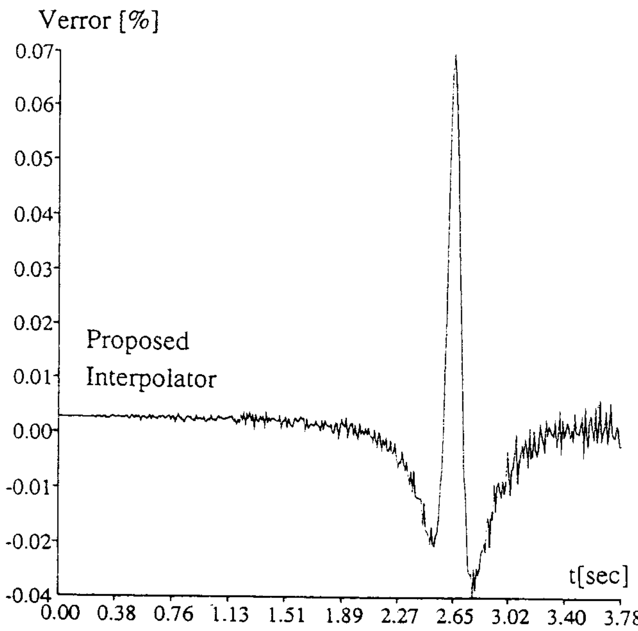


FIG. 8: VARIATIONS IN THE VELOCITY COMMANDS CAUSED BY THE PROPOSED INTERPOLATOR. (A) EQ. 12; (B) EQ. 19.

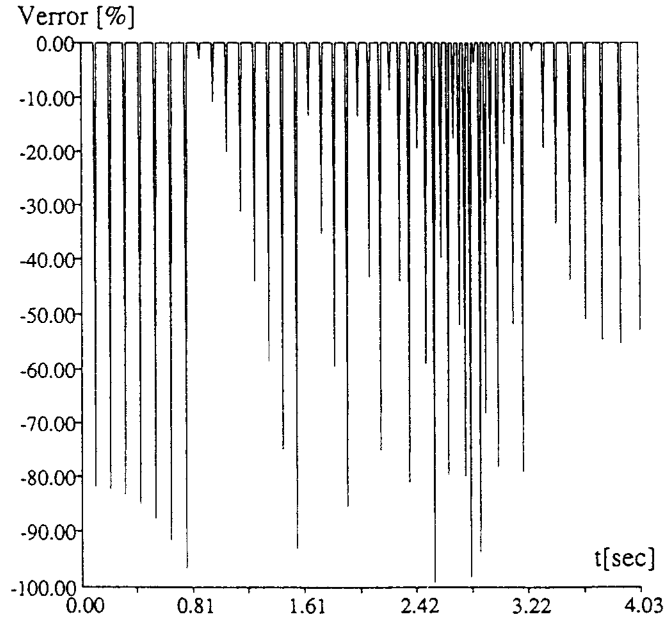


FIG. 9: VARIATIONS IN THE VELOCITY COMMANDS TO THE CONTROL LOOPS CAUSED BY CAD CONVENTIONAL INTERPOLATOR.

Example 2

The 3D cubic curve

$$\begin{cases} x = 11.9u^3 - 29.8u^2 + 32.9u + 5.0 \\ y = 47.6u^3 - 41.7u^2 + 16.55u + 2.5 \qquad 0 \le u \le 1 \\ z = 11.9u^3 - 5.95u^2 + 9.05u + 5.0 \end{cases}$$

was simulated with V= 1.2 m/min, T=0.01 sec.

The simulations results for a 3D cubic spline are similar to those for a 2D cubic spline. The results are summarized in Table 2.

Based on the above examples, as well as on many others that we tried, we may conclude the following:

(1) The conventional approach needs a very large number of line segments to guarantee a small contour error. Note that in the first (2-D) example the length of the tool path is about 94.5 mm, but 70 segments are needed to approximate it with a tolerance of 11 µm. This subdivision caused a decrease of about 9% in the average feedrate.

(2) The feedrate error caused by the proposed interpolator depends on the number of terms used to approximate the next spatial parameter u_{k+1}. If the requirement is not very strict (e.g., $|\delta V\max|/V < 3\%$), only the first term in Taylor's expansion is needed, and, in turn, the computation is very fast. In Example 1, the numbers were $|\delta V\max|/V = 1.7\%$ when only the first term was used, and 0.07% when the second term was used as well.

(3) In Example 2 (3-dimensional curve) even better results were achieved. The results are better both with the conventional and the proposed interpolator. The reason for this is that the 3-D curve has larger minimal radius of curvature. The proposed interpolator still offers higher performance.

TABLE 2: INTERPOLATION OF A 3D CUBIC SPLINE.

Proposed interpolator:			
Equation used:	Eq. 12	Eq.19	
$\varepsilon_{max}[\mu m]$	0.80	0.78	
$\delta V_{max}/V$	1.3%	0.035%	
Conventional Interpolator:			
# of line segments	10	30	60
$\varepsilon_{max}(\mu m)$	77	8.8	2.2

CONCLUSIONS

The amount of geometric information transferred from the CAD to the CNC must be minimized but must still enable the machining of any general curve. These conflicting requirements can be satisfied only by the development of real-time interpolators for general curves. The most common method to present curves is by their parametric forms. Accordingly, a new type of CNC interpolators which can handle general curves was introduced. The only assumption made is that the first two derivatives exist, as is the case for smooth curves. The performance of the interpolator, evaluated in terms of precision and feedrate deviations, is superior to all existing interpolation methods.

ACKNOWLEDGEMENTS

This research was partially sponsored by NSF grant #DDM-9114131 and by the Japan TS Research Fund through the Technion's Vice President for Research grant #033-0861. The paper was written while Yoram Koren was a Lady Davis Visiting Professor at the Technion. The authors acknowledge the support of the Lady Davis Foundation.

REFERENCES

Adams, J.A., 1975, "The Intrinsic Method for Curve Definition," *Computer Aided Design*, Vol. 7, No. 4, pp. 243-249.

Chou, J.J. and Yang, D.C.H., 1991, "Command Generation for Three-Axis CNC Machining," *Trans. of ASME, Journal of Engineering for Industry*, Vol. 113, August, pp. 305-310.

Chou, J.J. and Yang, D.C.H., 1992, "On the Generation of Coordinated Motion of Fine-Axis CNC/CMM Machines," *Trans. of ASME, Journal of Engineering for Industry*, Vol. 114, February, pp. 15-22.

Faux, I.D. and Pratt, M.J., *Computational Geometry for Design and Manufacture*, Ellis Horwood Publishers, 1979.

Huang, J.T. and Yang, D.C.H., 1992, "Precision Command Generation for Computer Controlled Machines," PED-Vol. 58, Nova, pp. 85-104.

Huang, J.T., 1992, "Design and Application of a New CNC Command Generator for CAD/CAM Integration," Ph.D. Thesis, UCLA.

Koren, Y., 1976, "Interpolator for a Computer Numerical Control System," *IEEE Transactions on Computers*, Vol. C-25, No. 1, Jan., pp. 32-37.

Koren, Y. and Masory, O., 1981, "Reference-Pulse Circular Interpolators for CNC systems," *Trans. of ASME, Journal of Engineering for Industry*, Vol. 103, Feb., pp. 131-136.

Koren, Y. and Masory, O., 1982, "Reference-Word Circular Interpolators for CNC systems," *Trans. of ASME, Journal of Engineering for Industry*, Vol. 104, Nov., pp. 400-405.

Koren, Y., 1983, *Computer Control of Manufacturing Systems*, McGraw-Hill, New York.

Lien, T.K., 1980, "Coordinate Transformation in CNC Systems for Automatic Handling Machines," *CIRP Manuf. Syst.*, Vol. 9, No. 1, pp. 49-60.

Lo, C.C., 1992, "Cross-coupling Control of Multi-Axis Manufacturing," *Ph.D. Thesis*, The University of Michigan, Ann Arbor.

Loney, G.C. and Ozsoy, T.M., 1987, "NC Machining of Free Form Surfaces," CAD, Vol. 19, No. 2, pp. 85-90.

Makino, H., 1988, "Clothoidal Interpolation - A New Tool for High-Speed Continuous Path Control," *Annals of the CIRP*, Vol. 37, No. 1.

Makino, H. and Ohde, T., 1991, "Motion Control of the Direct Drive Actuator," *Annals of the CIRP*, Vol. 40, No. 1, pp. 375-378.

Papaioannou, S. G., 1979, "Interpolation Algorithms for Numerical Control," *Computers in Industry*, Vol. 1, pp. 27-40.

Sata, T., Kimura, F., Okada, N. and Hosaka, M., 1981, "A New Method of NC Interpolation for Machining the Sculptured Surface," *Annals of the CIRP*, Vol. 30, No. 1, pp. 369-372.

Stadelmann, R., 1989, "Computation of Nominal Path Values to Generate Various Special Curves for Machine," *Annals of the CIRP*, Vol. 38, No. 1, pp. 373-376.