Direct Process Control Using n-Dimensional NURBS Curves

Robert M. Cheatham, W. Edward Red[1] and C. Greg Jensen

Brigham Young University, [1]ered@et.byu.edu

ABSTRACT

Direct control allows CAD/CAM applications to pass native part geometry directly to a machine tool for part processing. Although highly efficient for 3-axis machines, one of direct control's advantages is in implementing new algorithms for 5 and 6-axis machining, where the tool orientation is a more complex function of the surface geometry. This paper introduces a new extended n-dimensional NURBS vector that incorporates tool orientation into control parameters. By including state control parameters such as feed rate and spindle rpm, the n-D NURBS becomes the first mathematical representation to incorporate all required machining information.

Keywords: numerical control, motion planning, NURBS, manufacturing

1. INTRODUCTION

Direct Machining And Control (DMAC) [1, 8] is a new method of controlling machine tools. Developed at Brigham Young University, Direct Control is designed to make manufacturing processes CAD centric as opposed to the current ASCII-based, M&G code centric process. Like some other control concepts disclosed in recent years [9, 10], the DMAC concept provides for an open architecture PC software controller. One of the unique differences with Direct Control is that the process planning software (CAD/CAM systems, robotic simulation software, coordinate measuring software, process optimization software, etc.) resides on the same computer as the controller.

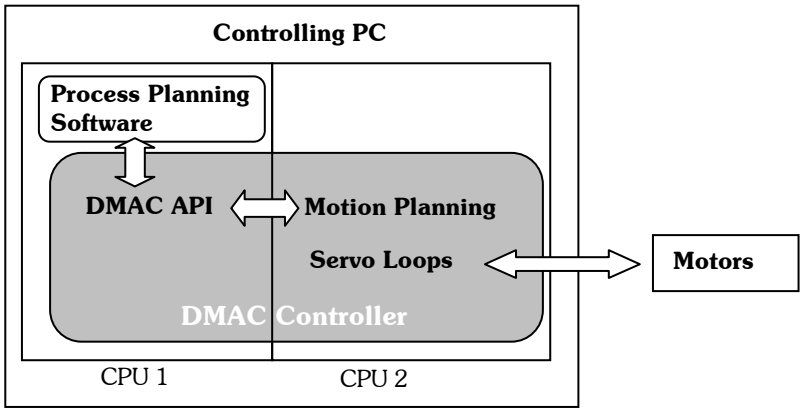


Fig. 1. DMAC Architecture

The DMAC controller consists of a dual processor computer running two operating systems, currently Windows and VenturCom RTX. One processor runs Windows and the process planning software while the other runs RTX and the DMAC control software (Fig. 1). Thus the process planning software can be linked directly to the machine tool. Information in the form of tool paths, I/O commands, machine states, and sensor readings are passed between the process planning software and the controller via function calls and shared memory. This gives real time decision-making ability to process planning software and factory control software so that changes can be made at the process plan level while the job is in-process. Since manufacturing operations are controlled directly from the process plan there is no dissociation from the original part file. No translation is necessary so the data remains in its native format and no error is introduced.

Contrast this with the current industry standard of M&G code. Controllers adhering to M&G code standards accept process steps and sequences in the form of ASCII block formatted commands. These process steps consist of mechanism commands, such as speed, coolant flow, and tool changes, and movement commands that consist of tessellated geometry – line segments and arcs. Tessellated moves may not be tangent, requiring blending to achieve smooth motion. For contouring operations in particular, some controllers refit the tessellated tool paths with splines as a preprocessing operation.

A weakness in this methodology is that once the ASCII files are generated, they are no longer associated with the original process plan. No information or changes can flow to or from the original process plan without regeneration of the ASCII file. This is strictly a unidirectional data flow. Another weakness is that the tessellated geometry does not reflect the native geometric data in the process plan, since many process planning systems store their tool paths as splines such as NURBS (see section 3.1). Upon postprocess, these splines are tessellated to work with the M&G code standard. The refitting of the tool path introduces error and further dissociates the manufacturing operation from the original CAD model and process plan.

Direct Control uses the native process geometry (splines, lines, arcs, etc.). This maintains the original manufacturing intent and also gives the process planning system an additional degree of control over the machine motion. Many spline generation methods have been developed for achieving smooth motion, but Direct Control allows the process planning software to choose a spline generation method that is best suited to the application.

2. MOTIVATION

In order to keep the process CAD centric, maintain bi-directional information flow, and avoid refitting errors, a Direct Control compliant controller needs to be able to follow the original NURBS curves generated in the process planning system. Since five-axis machining methods are increasingly represented in CAM systems, NURBS containing five-axis data must also be successfully followed. To solve this problem, we present in this paper a NURBS path following methodology suitable for Direct Control.

n-Dimensional (more than 3 dimensions) NURBSs are presented as a new entity type for representing all manufacturing tool paths. We show how concept opens new possibilities in manufacturing control. A section on speed control shows how we maintain speed along parametric curves. We then explain our methods of trajectory generation and how we limit the speed to maintain acceptable path and mechanism dynamics. An example is given showing the method in practice. Results are discussed and conclusions drawn as the reader is left to ponder future research and possibilities associated with direct control.

3. NURBS CURVES

3.1 Definition

A degree *m* NURBS curve is a vector valued piecewise parametric function

$$\mathbf{Q}(u) = \frac{\sum_{i=0}^{p-1} w_i \mathbf{Q}_i B_i^m(u)}{\sum_{i=0}^{p-1} w_i B_i^m(u)}; \qquad \mathbf{Q}_i \in \mathbb{R}^n \tag{1}$$

defined by *p* vectors $\mathbf{Q}_i$ (commonly referred to as control points or control vertices) with *p* corresponding positive real weights w_i, blending functions $B_i(u)$, and a non-decreasing set of m+p-1 real numbers known as the knot vector **K**. The blending functions are defined recursively by

$$B_a^b(u) = \begin{cases} \alpha B_a^{b-1}(u) + \gamma B_{a-1}^{b-1}(u) & b \neq 0 \\ 1 & b = 0,\ K_{a-1} \leq u < K_a \\ 0 & else \end{cases} \tag{2}$$

where

$$\alpha = \frac{u - K_{a-1}}{K_{a+m-1} - K_{a-1}} \tag{3}$$

$$\gamma = \frac{K_{a+m} - u}{K_{a+m} - K_a} \tag{4}$$

and the parameter u is defined on the interval $[K_{m-1}, K_{p-1}]$.

Drawing line segments between sequential control points yields a piecewise line known as the control polygon. The control polygon has many uses, one of which is analyzing the variation diminishing property. The variation diminishing property of NURBSs states that a general line for planar NURBSs, or a general plane for space NURBSs, will not intersect the curve more times than it intersects the control polygon. This means that the curve will not "wiggle" more than the control polygon.

For a more in-depth discussion of NURBS curves and associated algorithms the reader is referred to [3, 11].

3.2 n-D NURBS for Manufacturing

The use of NURBSs in CAx applications and manufacturing has typically been limited to planar and space curves. Use of NURBSs as 3-axis tool paths has allowed machine tools to achieve smooth motion and higher feed rates in the manufacture of freeform shapes. Recent papers have attempted to extend these benefits to 5-axis tool paths [5, 7].

We propose the use of additional dimensions in the control points to extend the benefits of NURBS to 5-axis methods and additional aspects of manufacturing processes. For example, the two angular components of spherical coordinates can be added to specify tool direction in five axis machining. Denavit-Hartenberg parameters [2] can be used to define motion for robots with orientable end effectors. Such parameters as amperage and element feed rate can also be specified for the operation of an attached welder. We will refer to these non-Cartesian dimensions as auxiliary dimensions and the parameters they represent as auxiliary parameters.

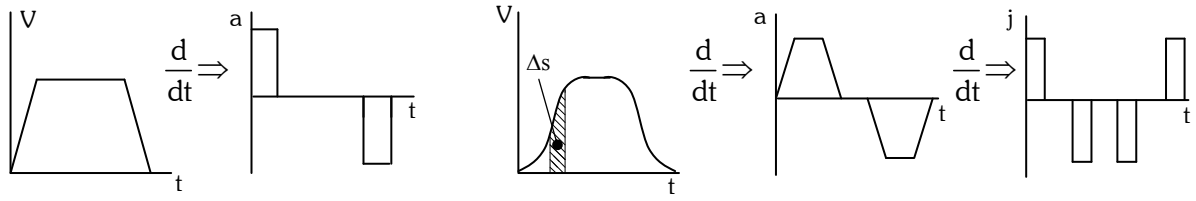
4. SPEED CONTROL

Traversing a path requires attention to be paid to several aspects of the manufacturing process. Following the path within the specified tolerance is critical. Mechanism dynamics require proper joint rate scheduling to maintain acceptable forces on the mechanism and tool. Unacceptable dynamics will damage a mechanism and also exceed process tolerances. Alternatively, conservative path planning causes a process to execute slowly, wasting time and resources. This section outlines our methods for optimizing the machining motion along NURBS paths.

4.1 Speed Profiles

Motion control in manufacturing is typically governed by speed. A desired speed is specified for a particular portion of a process plan. This speed cannot instantly be attained from a stationary position, so the speed must be ramped up slowly according to the dynamic capabilities of the machine. The plot of velocity versus time is known as a speed profile. Figs. 2(a) and 2(b) depict the well-known trapezoidal (constant acceleration) speed profile, which uses constant acceleration to change the speed. As the performance capabilities of machines and motors have improved, it has become necessary to use jerk-limited (s-curve) speed profiles, such as the constant jerk speed profile in Figs. 2(c), 2(d), and 2(e).

In modern digital control, movement is achieved by commanding the motors to move the joint axes to set points in P_i at specified time intervals. The frequency f of the set point commands is called the trajectory rate. To maintain a constant speed along a path, the change in curvilinear distance must remain constant between each set point. The curvilinear distance Δs between set points is found by $\Delta s = V/f$. When following a speed profile along a path, the distance between set points is determined by taking the area under the curve between time steps, as in Fig. 2(c).



a) Trapezoidal profile b) Trapezoidal acceleration c) S-curve profile d) S-curve acceleration e) S-curve jerk

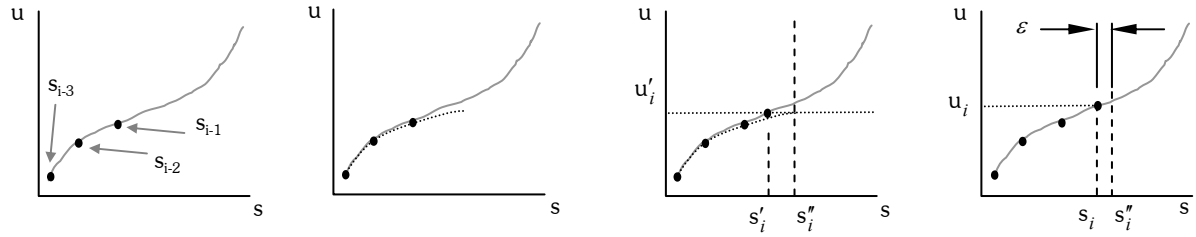
Fig. 2. Speed profiles and their derivative profiles

4.2 Guaranteeing Distance

To specify a point on a parametric curve, an appropriate parameter value u_i must be chosen. The difficulty lies in the relationship between the parameter value u and the length s along the curve (5). This relationship is highly nonlinear and does not, in general, have a closed form [4].

$$s(u) = \int_{u_0}^{u} \left| \frac{\mathbf{Q}(u)}{du} \right| du \tag{5}$$

We use the method of [13] to deal with the nonlinear parameter-length relationship. This method establishes a look-up table, Fig. 3(a), of parameter and length values using Gaussian Quadrature to calculate the length values. LaGrange interpolation, Fig. 3(b) is applied to the parameter and length values of previous set points to generate an approximating function for the parameter-length relationship. Given the desired length s_i'' along the curve of the set point currently being calculated, this function predicts the parameter value u_i', Fig. 3(c). A second Gaussian Quadrature operation calculates the actual length s_i' yielded by the predicted parameter value. The length error generated by the previous set point is then used to correct for the length error in the current set point, yielding the final parameter u_i, Fig. 3(d). The prediction and correction process may be repeated if the distance ε between the length s_i yielded by the parameter u_i and the desired length is greater than the allowable error.



a) Previous 3 lengths b) LaGrange interpolation curve c) Predicted parameter d) Corrected parameter

Fig. 3. Parameter prediction/correction process

5. MOTION CONTROL

Lookahead methods ensure that path motion does not cause the machine joints to exceed their motion limits. Generally, controllers process a sequence of moves to see if a downstream move vector changes direction too quickly, or precipitates a sudden change in path curvature. Paths where the curvature changes rapidly limit motion because of the machine's jerk capabilities. Once the lookahead discovers such problems, backward recursion is used to modify the upstream motion accordingly.

Machine tool controllers often compete on the number of blocks (tessellated move segments) that can be buffered for the lookahead algorithms. Newer controllers may refit a block move sequence to a higher order polynomial such a NURBS and then process the NURBS directly. If the NURBS path specifies tool orientation change also, requiring a machine with orientation axes, a second NURBS might be used to represent the tool orientation change.

In general, 5 or 6-axis machines are challenged when tool orientation dynamically changes along complex paths. For example, to use advanced algorithms like curvature matched machining [6], modern controllers require that the contouring moves be pre-processed into a sequence of small linear segments with proportional tool orientation change over each segment. The controller can then blend these moves together. One of our objectives is to show how an n-D NURBS entity can be directly processed in the DMAC controller. The next two sections consider entity lookahead methods as applied in the DMAC controller.

5.1 Lookahead Method

In the DMAC controller the lookahead methods process a sequence of whole moves (entities) rather than a large block of small moves that have been tessellated from the part's complex curves. These entities mathematically represent the actual geometry of the process plan

Where typical lookahead buffers may process from 200 – 1000 blocks, DMAC's entity buffer usually processes 8 to 10 entities at a time, generating and updating the speed profile as necessary. Nevertheless, the summed path length of the buffer entities (lines, arcs, NURBS, n-D NURBS) will generally exceed that of many tessellated move buffers, providing a greater lookahead capability without the need for additional computing hardware.

The entity lookahead is made possible by exploiting the variation diminishing property of NURBSs. Since the curve cannot wiggle more times than its control polygon, the variation of its shape is limited. If the curve is sampled at an appropriate interval, the general shape of the curve will be evident, as shown in Fig. 4. This allows for a fixed number of sample points and a fixed amount of preprocessing for each knot interval. The number of sample points depends on the desired accuracy and is an implementation issue.

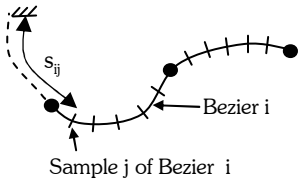


Fig. 4. Beziers and sample points

As moves are processed by the lookahead, the following information is determined at each sample point: 1) path length (s_{ij}); 2) tangent vector relative to a part frame; 3) curvature at each sample point; 4) path value $\mathbf{Q}_{ij} = \mathbf{Q}(u_{ij})$.

5.2 Path Limitations

Path dynamics consist of such variables as speed, acceleration, jerk, and for n-D NURBSs tool paths, changes in auxiliary parameters. Path dynamics affect the results of the manufacturing process and the life of the tool. For instance, a robot end effector may be subject to damage by high acceleration or jerk, or a cutting tool may be subject to chatter or sustain damage if loads are applied too quickly. The desired and allowable values of the path dynamics variables are called path parameters.

Given the path parameters, the lookahead algorithm performs a series of calculations to reduce the allowable path speed over sample points on each Bezier. The lookahead limits the path speed over each sample segment by 1) estimating the allowable centripetal speed given the sample curvature κ_{ij} and the maximum allowed radial acceleration, a_m and 2) limiting the jerk when the curvature changes (and thus the acceleration) from one sample point to the next. For simplification, we drop the Bezier subscript i from the following equations.

Centripetal check:

$$v_j = \sqrt{a_m / \kappa}_j \leq v_f \text{ (desired feed rate)} \tag{6}$$

Jerk check for $\kappa_{j+1} \neq \kappa_j$*:*

$$j_j = (a_{j+1} - a_j)/t_i = v_j (v_j^2 \kappa_{j+1} - v_j^2 \kappa_j)/(s_{j+1} - s_j) \leq j_m \text{ (max jerk)} \tag{7}$$

$$v_j \leq \sqrt[3]{j_m (s_{j+1} - s_j)/(\kappa_{j+1} - \kappa_j)} \leq v_f \tag{8}$$

5.3 Mechanism Limitations

Mechanism dynamics may further limit the path speed over the sample points. Both branching mechanisms with more than one rotational joint and serial mechanisms can complicate the lookahead due to: 1) $R\theta$ motion amplification, wherein a translation joint must move a greater distance than the tool tip due to the reorientation of the tool; and 2) motion singularities which cause sudden increased velocities in certain joints.

$R\theta$ motion amplification is a problem for many 5-axis machine tools because the rotational joints are often serially attached to the end of an *X-Y-Z* set of axes, or separately to a branched base table. Offset motion of the tool with rapid tool reorientation amplifies the motion of the translational joints is shown in Fig. 5. Compare δx to ΔX to see the motion amplification. This phenomenon makes 5-axis machining or any other movement with rapid tool reorientation difficult from the viewpoint of motion planning.

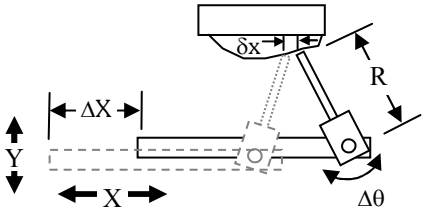


Fig. 5. Rθ motion amplification

A similar problem occurs when two rotational axes are present and the tool orientation change is to occur in the plane of the axes; see the A-C combination in Fig. 6. The tool is to change orientation from the $\mathbf{z}_1$ unit direction to the $\mathbf{z}_2$ unit direction as defined by the **e** vector normal to the instantaneous A-C plane, see Fig. 7. C must rotate by 90° while A changes by angle θ. Small θ will result in excessively large rotational rates for the C axis.

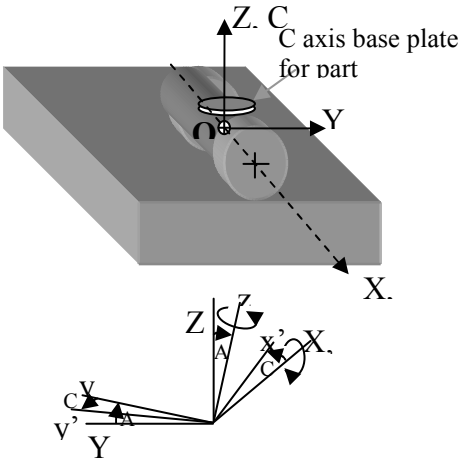


Fig. 6. A-C Table

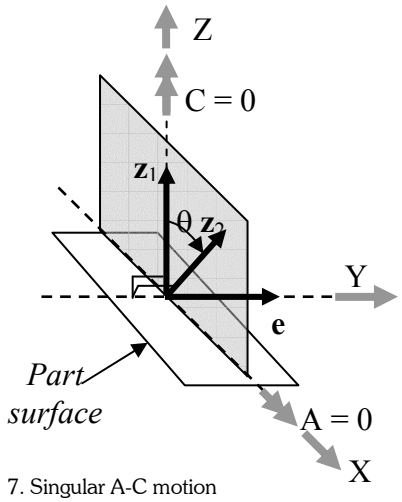


Fig. 7. Singular A-C motion

DMAC uses two lookahead methods to resolve possible singularity and amplification problems: 1) inverse kinematics lookahead; and 2) Jacobian lookahead. Moves are delimited by frames (represented as homogeneous transformations) that describe the tool orientation relative to the path at the move endpoints. Inverse kinematics are applied before the move is made to discover suddenly large changes in joint values, characterizing singularity or near-singularity. Both motion amplification and singularity problems can be discovered in this way. The procedures then modify the motion parameters for the move.

When a singularity is discovered, it is simply necessary to reduce the path speed so that the rotational motion remains within limits. Unfortunately, for mechanisms that exhibit the $R\theta$ motion amplification shown in Fig. 5, inverse kinematics lookahead is insufficient, since the prescribed motion may cause *X-Y-Z* joint accelerations and jerks to exceed their limits due to $R\ddot{\theta}$ and $R\dddot{\theta}$ amplification. This problem is resolved by generating Jacobian equations for the mechanism, estimating X-Y-Z accelerations and jerks, then using the Jacobian to limit the path motion accordingly. Thus, use of the Jacobian also allows the lookahead to adjust for any other path-mechanism interactions including those caused by auxiliary parameters and their corresponding mechanism functions (auxiliary joints).

An example of these methods is shown in the following section.

5.4 X-Y-A-C (Base Branch) and Z (Tool Branch) Lookahead

This sample derivation uses a geometric solution approach, where the base branch of the mechanism is defined by an *X-Y-Z* base frame as shown in Fig. 6. This frame is colinear with the *X-Y-Z* joint axes when the mechanism is in its zero state. Given angles A, and C, we determine the orientation frame from the rotation sequence A – C as shown in Fig. 5. This generates the orientation frame **R**(A, C) of *x'y'z'* relative to *XYZ* as

$$\mathbf{R}(\mathrm{A,\ C}) = \begin{bmatrix} cosC & -sinC & 0 & 0 \\ cosA\,sinC & cosA\,cosC & -sinA & 0 \\ sinA\,sinC & sinA\,cosC & cosA & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{9}$$

To determine the *X*, *Y*, and *Z* joint values, rotate a part position **p** to position **P** in the base frame by

$$\mathbf{P} = \mathbf{R}\,\mathbf{p} \tag{10}$$

and determine the joint values by

$$X = O_x - P_x \tag{11}$$
$$Y = O_y - P_y \tag{12}$$
$$Z = P_z - O_z \tag{13}$$

where O_x, O_y, and O_z are the tool offsets relative to the X-Y-Z base frame.

We calculate the velocity of the part feature using the table branch equations:

$$\boldsymbol{v}_b = \boldsymbol{\omega}_b \times \boldsymbol{P} + \begin{bmatrix} \dot{X} \\ \dot{Y} \\ 0 \end{bmatrix} \tag{14}$$

$$\boldsymbol{\omega}_b = \dot{A}\,\mathbf{i} - \dot{C}\,sinA\,\mathbf{j} + \dot{C}\,cosA\,\mathbf{k} \tag{15}$$

The tool branch equations are

$$\boldsymbol{v}_t = \begin{bmatrix} 0 \\ 0 \\ \dot{Z} \end{bmatrix} \tag{16}$$

$$\boldsymbol{\omega}_t = \boldsymbol{0} \tag{17}$$

The tool relative to part velocities then become:

$$\mathbf{v} = \mathbf{v}_t - \mathbf{v}_b \tag{18}$$

$$\boldsymbol{\omega} = \boldsymbol{\omega}_t - \boldsymbol{\omega}_b \tag{19}$$

where $\mathbf{v} = (v_x, v_y, v_z)$ and $\mathbf{P} = (P_x, P_y, P_z)$. Equation (18) and its derivatives can be used in a lookahead method to estimate X, Y, and Z speeds, accelerations, and jerks due to excessive A and C motion. When excessive we reduce the move speed, acceleration and jerk settings to acceptable limits.

Applying (14), (16) and (18) and rearranging, we relate the amplified axis rates for X-Y-Z to the part position and rotational rates:

$$\dot{X} = -v_x + \sin A\,\dot{C}\,P_z + \cos A\,\dot{C}\,P_y \tag{20}$$

$$\dot{Y} = -v_y - \cos A\,\dot{C}\,P_x + \dot{A}\,P_z \tag{21}$$

$$\dot{Z} = v_z + \dot{A}\,P_y + \sin A\,\dot{C}\,P_x \tag{22}$$

where (v_x, v_y, v_z) are determined from the speed set for the move and the path tangent vector at the beginning of the move.

Similarly, we estimate the X-Y-Z acceleration amplification from these equations:

$$\ddot{X} = -a_x + \dot{A}\,\dot{C}\,(\cos A\,P_z - \sin A\,P_y) + \ddot{C}\,(\sin A\,P_z + \cos A\,P_y) \tag{23}$$

$$\ddot{Y} = -a_y + (\sin A\,\dot{A}\,\dot{C} - \cos A\,\ddot{C})P_x + \ddot{A}\,P_z \tag{24}$$

$$\ddot{Z} = a_z + \ddot{A}\,P_y + \cos A\,\dot{A}\,\dot{C}\,P_x + \sin A\,\ddot{C}\,P_x \tag{25}$$

where (a_x, a_y, a_z) are estimated from the curvature properties of the curve at the beginning of the move.

We use only the third derivative terms to predict X-Y-Z jerk amplification caused by the A and C jerk:

$$\dddot{X} = \dddot{C}\,(\sin A\,P_z + \cos A\,P_y) \tag{26}$$

$$\dddot{Y} = -\cos A\,\dddot{C}\,P_x + \dddot{A}\,P_z \tag{27}$$

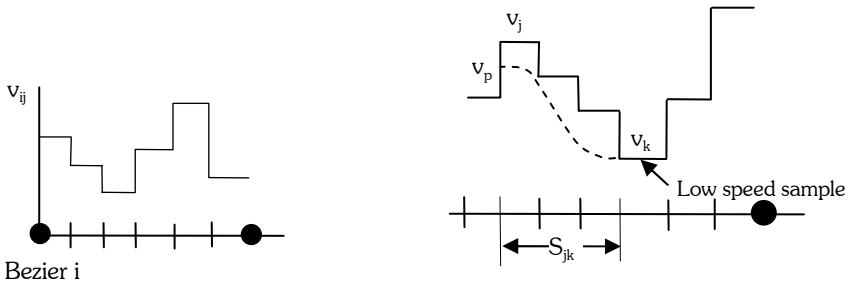
$$\dddot{Z} = \dddot{A}\,P_y + \sin A\,\dddot{C}\,P_x \tag{28}$$

5.5 Speed Profiler

Once we have determined each sample point speed, Fig. 8(a), we ensure that we can make the transition between the sample point speeds, given the sample distances. The problem occurs when a speed drops too fast for the allowable motion limits for the machine. To rectify this problem, we backwardly recurse from the last sample point to the first sample point, reducing the intermediate sample point speeds as necessary.

In the backward recursion we identify low allowable speed points preceded by higher allowable speed points – see sample points j and k in Fig. 8(b) (i subscript dropped). Given sample relative distance $S_{jk} = s_k - s_j$ and using the adaptive S-curve trajectory profiler developed by [12], we determine whether we can attain the desired sample speed

while retaining the ability to reduce the speed to that of the low point in the remaining distance. If not, we reduce the offending sample point speed. In Fig. 8 the dashed line is the backwards trajectory profile. We can only attain v_p in the distance S_{jk}; thus, we reduce v_j to v_p. This yields a dynamic trajectory profile that attempts to maintain the desired speed while keeping all path and joint dynamics within specified constraints.

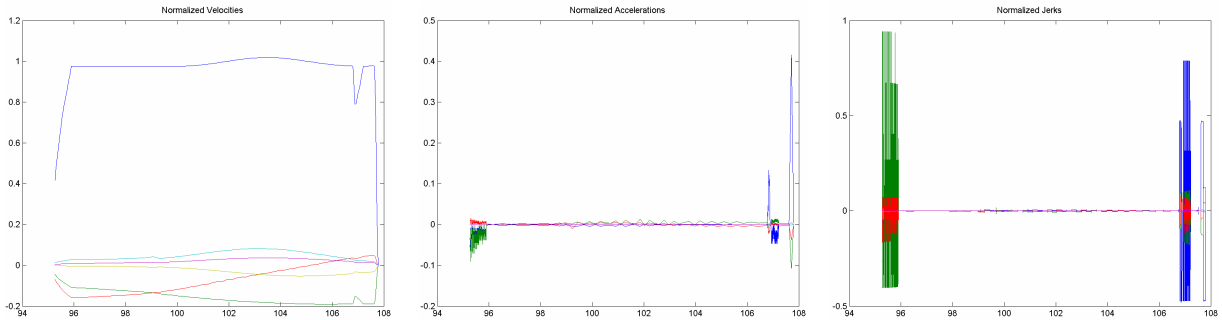


a) Limiting Sample Speeds b) Backward recursion from low speed sample

Fig. 8. Speed profile generation

6. RESULTS

We tested our methods on a machine with a kinematic configuration similar to that described in Section 5.4 and Fig. 6. Figs. 9(a-c) show the actual machine dynamics (normalized by their respective allowable values) as a function of time over two n-D NURBS moves used in curvature matched machining. The motion dynamics reflect path curvature changes over the surface along with $R\theta$ motion amplification and subsequent motion adjustment. The plots show the normalized dynamics of all five axes. The blue trace in Fig. 9(a) shows the actual feed rate normalized by the desired feed rate. The dip in velocity near the end of the movement results from an immature blending algorithm. Note that all dynamics are within their allowable values.



a) Normalized Joint Velocities and Path Velocity b) Normalized Joint Accelerations c) Normalized Joint Jerks

Fig. 9. Machine Dynamics along an n-D NURBS curvature matched machining tool path

7. CONCLUSION

We have shown the suitability of our method for following n-D NURBS tool paths. We are able to follow these complex tool paths while maintaining acceptable mechanism dynamics. While our example demonstrates the use of n-D NURBS for 5-axis machining it is not restricted to n being less that or equal to five, nor is it restricted to milling mechanisms. We invite other researchers to apply our mathematics and methods to devices having more degrees of freedom, and to devices with varying types of controllable components.

8. REFERENCES

[1] Bassett CP, Jensen CG, Bosley JE, Red WE, Evans MS. Direct Machining: A New Paradigm Machining Data Transfer. 5th Design for Manufacture Conference - ASME 2000 Design Engineering Technical Conferences. Baltimore, MD. paper no. DFM-1498. Sept. 10-13, 2000.

[2] Denavit J, Hartenberg. A Kinematic Notation for Lower-Pair Mechanisms Based on Matrices. Journal of Applied Mechanics. June 1955. p215-221.

[3] Farin G. Curves and Surfaces for CAGD, 5ed. Morgan Kaufmann. 2002.

[4] Farouki RT, Sakkalis T. Real Rational Curves are not 'unit speed.' CAGD. 1991. v8. p151-157.

[5] Fleisig RV, Spence AD. A Constant Feed and Reduced Angular Acceleration Interpolation Algorithm for Multi-Axis Machining. CAD. 2001. v33. p1-15.

[6] Jensen CG, Anderson DC. Accurate Tool Placement and Orientation for Finish Surface Machining. Journal of Design and Manufacturing. v3. p251-61. 1993.

[7] Jüttler B, Wagner MG. Computer-aided design with spatial rational B-spline motions. ASME Journal of Mechanical Design. 1996. v118. n2. p193-201.

[8] Li W, Davis T, Jensen CG, Red WE. Rapid and flexible prototyping through direct CNC. Computer-Aided Design and Applications. v1. n1-4. p91-100. 2004.

[9] OMAC, http://www.omac.org, OMAC Users Group.

[10] OSACA, http://www.osaca.org.

[11] Piegl L, Tiller W. The NURBS Book, 2ed. Springer Verlag. 1997.

[12] Red WE. Dynamic Optimal Trajectory Generator for Cartesian Path Following. Robotica. 2000. v18. n5. p451-458.

[13] Yang Z, Red WE. On-line Cartesian Trajectory Control of Mechanisms Along Complex Curves. Robotica. 1997. v15. n3. p263-274.