Математички Билтен
Vol. 39(LXV) No. 1
2015 (49-63)
Скопје, Македонија

ISSN 0351-336X
UDC: 519.713-74

SPEED CONTROL IN NUMERIC CONTROLLED SYSTEMS

Igor Dimovski[1], Samoil Samak[2], Dijana Cvetkoska[3],
Mirjana Trompeska[4], Filip Kochoski[5]

Abstract. In order to achieve higher speed (higher productivity at the same time), the modern way of managing numerical controlled systems includes Look Ahead algorithms with strong mathematical background. The purpose of these algorithms is generating a speed profile with which the tool will move along the programmed movement path. In this article will be described a method for speed profile generating whereby we will use numerical methods for differential computing, spline interpolation/ approximation and linear programming. For testing and view of the generated speed profiles we will use the programming package MATLAB.

1. INTRODUCTION

NC machines, being typical mechatronics products, comprise machine tools that have a mechanical component and a numerical control system that is an electrical component. In NC, the servo motor is used for controlling the machine tool according to the operation of a user and a servo motor drive mechanism for activating the servo motor. That is, NC means a control device that machines a target part by activating the servo motor according to commands. The NC combined with computer technology is called computerized NC or CNC (Computer Numerical Control). Theoretical overview and details about CNC are given by Suh et al [7].

In high speed machining, it is crucial to minimize the cycle time, which reduces costs, while preserving the quality and tolerance integrity of the part being produced (Heng, [2]). The challenge is to get balance between accuracy and productivity. In some areas the accuracy is more important, in other the later one. Our research concern mostly about filament winding machines. In this area, the speed is limited by technological process. Taking all constrains and limitations in consideration, in this paper we will focus on minimizing the winding time as main criteria.

In early works, as in Bobrow et al [4], the problem is formulated. Solutions in those, so-called phase analysis methods, yield satisfying results only in the case of simple toolpaths. In last two decades, many algorithms treating this problem are developed. Some of the researches (Erkormaz and Altintas [8]) concern about parameterization of

2010 *Mathematics Subject Classification.* Primary: 93C85
*Key words and phrases.*velocity, acceleration, jerk, speed profile, spline approximation, optimization

the input geometry and the influence of the parameter interpolation on the feedrate profile, fluctuations of feedrate and violation of the acceleration and jerk constraints. Type of interpolation is very important. Some of the modern trends, employing splines (Akima splines, Bezier splines, cubic splines and NURBS) as interpolation type is tested and compared against usual linear, circular and polynomial interpolation type by Msaddek et al [9]. Review of different methods for parameter interpolation types is given by Siu [5].

There are many algorithms developed for speed control problem. Most of them use so-called Look Ahead approach, so they are called Look Ahead algorithms, despite the phase velocity planning (VP) of the appropriate algorithm is the one which is Look Ahead phase.

Direct sampling methods are characterized by interpolation of point for every sample time, mostly using first or second order Taylor series, taking the feedrate calculated on various ways. For example, Lai et al [10] adjust the feedrate trough backtracking procedure if acceleration, jerk or chord error constraints are violated. Similar approach using bisection method and backtracking procedure is proposed by Heng [2] and Beudaert et al [3]. Main disadvantage of methods with backtracking is estimation of computational complexity witch is difficult in this case. In this paper will be explained the *heuristic method* we have developed and tested against two more methods.

VP (look ahead) is basically nonlinear optimization problem. Sencer et al [6] explain method for discretization the VP problem and solved it using numerical, nonlinear optimization method. Solving nonlinear programming problems result with large computational time. Fan et al [1] proposes another discretization approach, and linearization of the problem, transforming it into linear programming problem, which has predictable computational time.

In our research, we have implemented two methods from this class. The *nonlinear programming method and linear programming method* will be described below and the obtained results are compared between them and against the heuristic method.

Problem formulation is given in the section 2. Detailed explanations of the proposed methods are elaborated in section 3. Results are explained in section 4. Conclusion and directions for future work are given in section 5.

2. SPEED CONTROL

The speed of the axes is usually called feedrate, or shortly feed, in the machining literature.

Fig. 1 shows the whole procedure for the speed control process. The input to this procedure is a tool path generated from the computer-aided manufacturing (CAM) and the aim of the procedure is to create a feedrate profile to follow this starting geometry with respect the drive constraints. In order to get smooth movement of the machine drives the first thing we need to do is to modify the starting geometry and get the input geometry, the geometry that is input to the velocity planning process. The aim of the

velocity planning process sometimes called feedrate interpolation is to generate optimized feedrate profile that will follow the starting geometry with the given tolerance and satisfy velocity, acceleration and jerk constraints of each drive. Finally the output of the velocity planning process, the feedrate profile is needed to be sampled to axis set points with respect to simple time.

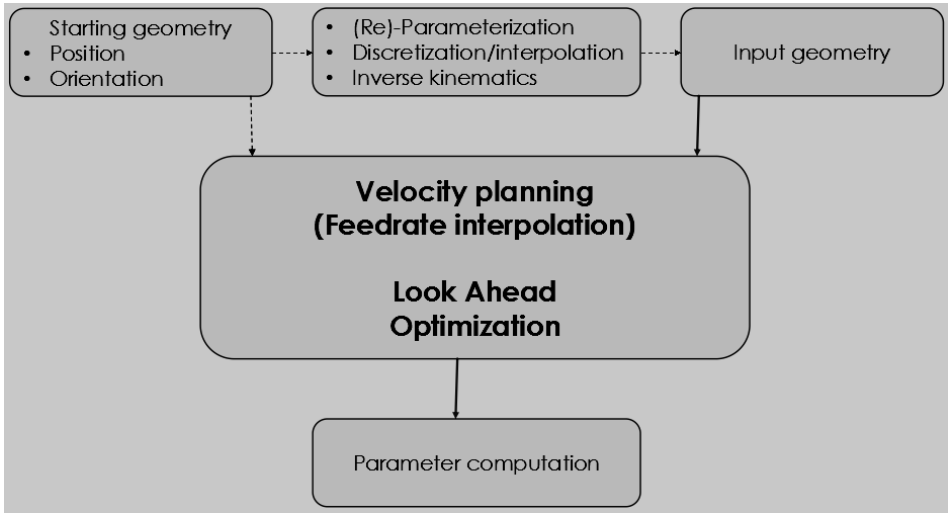


Fig 1. Speed Control Scheme

2.1. FROM STARTING GEOMETRY TO INPUT GEOMETRY

A typical several axis motion command in workpiece coordinate system that is produced from the CAM/CAD as a starting geometry is given by a sequence of discrete positions of the machine tool along the path. Each tool position is defined by three Cartesian coordinates of its center $P=[P_x\ P_y\ P_z]$ and angular orientation vector of the tool axis $O=[O_x\ O_y\ O_z]$. Because this sequence of discrete positions and orientations that define the starting geometry are given in workpiece coordinate system we use inverse kinematics to translate them in machine coordinate system. After this step is done the tool path is represented as discrete drive positions.

However this description of the tool path consist of line segments that can cause displacement, velocity, acceleration and jerk discontinuities during the velocity planning process and therefore we need to parameterized the given discrete drive positions from the starting geometry in to continuous function that is at least C^1 continuous. Therefore the sequence of drive positions are fitted to a cubic, quantic, NURBS, shape preserving or B-Splines in order to interpolate the intermediate cutter positions as the tool travels along the path. Tool path parameterization is important task of the speed control process because with this task we obtain a mathematical representation of a tool path such that the position coordinates of the tool tip can be computed in terms of an independent variable called the spline parameter. The most

important requirements of the tool path parameterization module are to generate splines that are geometrically continuous and to accurately describe the machining geometry.

In some of the algorithms for generating feed profile, tool-paths are generally parameterized with respect to their arc-length (s). The tool positions define the pose of the tool expressed as a function of path displacement (s) as:

$$\eta(s) = [x(s), y(s), z(s), a(s), c(s)], s \in [0, L] \quad (1)$$

where, L is the length of machine tool path. This is so-called *arc-length* parameterization.

Second, more general approach for tool path parameterization is when the parameter is in the interval [0,1] and does not depend on input geometry:

$$\eta(u) = [x(u), y(u), z(u), a(u), c(u)], u \in [0,1] \quad (2)$$

This parameterization is used when we describe the starting geometry as a spline, B-spline, NURBS and other curves that are at least C^1 continuous and the parameter need to be in [0,1] interval. These kinds of tool paths that are not parameterized according to their arc-length require an additional transformation from the spline parametric space to the arc-length displacement along the curve. Arc-length positions at each time step are converted to spline parameter values with the mapping defined by *u(s)* and substituted into the parametric definition of the curve such that: $\eta(u) \rightarrow \eta(u(s))$. For this additional mapping between spline parameter and arc-length of the tool path, we need to devote special attention because it is very important for the feedrate interpolation, especially when we calculate the geometric derivatives applying the chain rule.

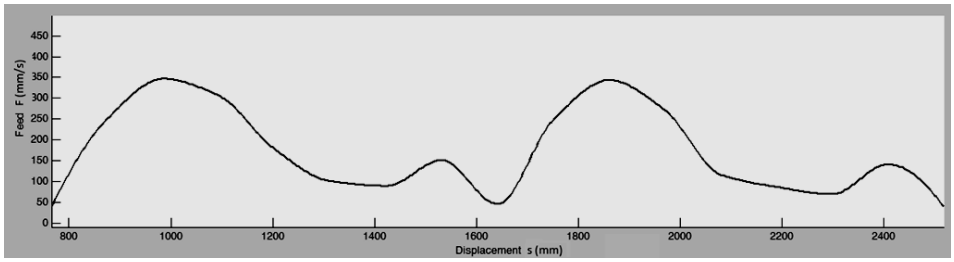
When the machine need to deal with complex workpieces, often happens the starting geometry to be expressed as a continuous curve like B-spline, NURBS or other kinds of geometric curves rather than discrete sequence of tool path positions. In this case first we need to do a discretization in order to get the discrete setpoint, and then do the inverse kinematics, parameterization/re- parameterization and interpolation to finally get the required input geometry.

2.2. VELOCITY PLANNING PROCESS

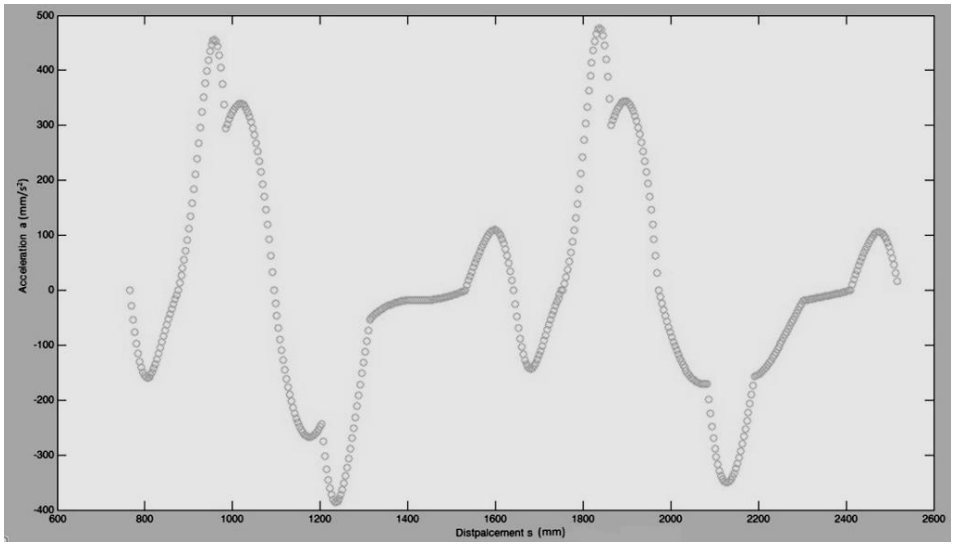
When multi-axis machine is programmed the goal behind velocity planning (look ahead) is a profile of tool speed - feedrate (appropriate acceleration and jerk) to be generated. There are different approaches for how the speed profile should be represented. The most common is the tool speed according to tool path $v(s)$, (Fig 2). Also common output of the velocity planning process is a feedrate profile according to spline parameter $v(u)$ and a speed according to the machining time $v(t)$.

2.2.1. FEED GENERATION

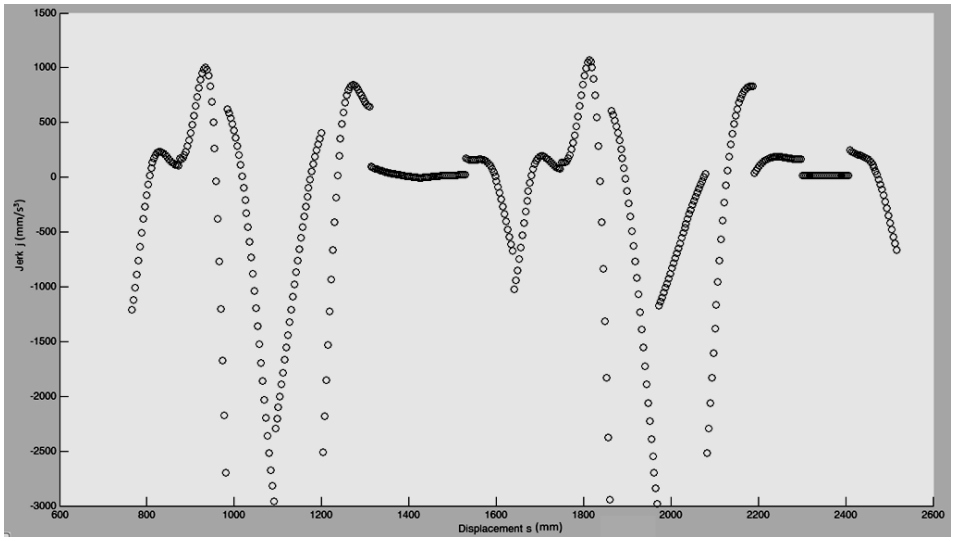
Feed generation characterizes the motion along the tool path in terms of the arc displacement $s(t)$, feed $\dot{s}(t)$, acceleration $\ddot{s}(t)$ and jerk $\dddot{s}(t)$ in the tangential direction. Or, if the input geometry is given in the term of formula (1) then velocity, acceleration and jerk profiles of each drive are evaluated as:



a) Feedrate profile along the tool path



b) Acceleration profile along the tool path



c) Jerk profile along the tool path

Fig. 2 Feed profile along the tool path

$$\begin{aligned}
\dot{\eta}(s) &= \frac{d\eta(s)}{ds}\dot{s}(s) \\
\ddot{\eta}(s) &= \frac{d\eta^2(s)}{ds^2}\dot{s}^2(s) + \frac{d\eta(s)}{ds}\ddot{s}(s) \\
\dddot{\eta}(s) &= \frac{d\eta^3(s)}{ds^3}\dot{s}^3(s) + 3\frac{d\eta^2(s)}{ds^2}\dot{s}(s)\ddot{s}(s) + \frac{d\eta(s)}{ds}\dddot{s}(s)
\end{aligned} \tag{3}$$

When more general parameterization is used, or when the tool path is represented as a continuous curve with respect to spline parameter $u \in [0,1]$ in the term of formula (2), with another application of the chain role we can obtain similar formulas for equations (3).

2.2.2. CONSTRAINTS

Because of the physical realization of the drives (motors, driving system, machine tool structure ...) the velocity, acceleration and jerk of each individual drive have to be limited. The jerk limitation is important to reduce the vibration due to the dominating vibratory mode of the axes.

As derivatives of the tool-path changes, the commanded path velocity, the feed, may violate the velocity, acceleration and jerk limits of active drives on the machine tool. The optimization constraints are chosen to ensure that the machine performs within the physical and control limits of its components and that the desired contouring accuracy during machining is maintained. For these reasons, constraints are imposed on the feedrate, and the velocities, motor torques, and jerks of all axes. The satisfaction of all imposed constraints is a common for all look ahead algorithms.

2.2.2.1. VELOCITY CONSTRAINTS

The velocities of all drives must not exceed their saturation limits:

$$V_{\max} = [v_{x\max}, v_{y\max}, v_{z\max}, v_{a\max}, v_{c\max}]$$

Velocity of the machine tool can be represented as vector valued parametric function (with respect to parameter w, where w can be arc-length parameter $s \in [0, \mathrm{L}]$ or spline parameter $u \in [0,1]$) such that the velocity of each of the machine drives are coordinates in the tool velocity function $V(w) = [v_x, v_y, v_z, v_a, v_c]$, where:

$$v_\tau(w) = \dot{\tau}(w) = \frac{d\tau}{dt} = \frac{d\tau}{dw}\frac{dw}{dt} = \tau_w(w)\dot{w}, \tau \in \{x, y, z, a, c\}$$

Since each axis has its own limitation the velocity constraints are given as:

$$|\tau_w(w)\dot{w}| \le v_{t\max} \quad \tau \in \{x, y, z, a, c\}. \tag{4}$$

2.2.2.2. ACCELERATION CONSTRAINTS

The acceleration of all drives must not exceed their saturation limits:

$$A_{\max} = [a_{x\max}, a_{y\max}, a_{z\max}, a_{a\max}, a_{c\max}]$$

Analogously, as velocity, the acceleration of the machine tool can be represented as vector valued parametric function with respect the same parameter w:

$$A(w) = [a_x, a_y, a_z, a_a, a_c] = [\dot{v}_x, \dot{v}_y, \dot{v}_z, \dot{v}_a, \dot{v}_c]$$

where:

$$a_\tau(w) = \dot{v}_\tau(w) = \frac{d^2\tau}{dt^2} = \frac{d^2\tau}{dw^2}\left(\frac{dw}{dt}\right)^2 + \frac{d\tau}{dw}\frac{d^2w}{dt^2} = \tau_{ww}(w)\dot{w}^2 + \tau_w(w)\ddot{w}, \tau \in \{x, y, z, a, c\}$$

According to this and the acceleration limits for each of the axes, acceleration constraints are given using formula 5:

$$|\tau_{ww}(w)\dot{w}^2 + \tau_w(w)\ddot{w}| \le a_{\tau\max}, \tau \in \{x, y, z, a, c\} \quad (5)$$

2.2.2.3. JERK CONSTRAINTS

Also the jerk of all drives must not exceed their saturation limits:

$$J_{\max} = [j_{x\max}, j_{y\max}, j_{z\max}, j_{a\max}, j_{c\max}]$$

Analogously to the velocity and acceleration constraints, the jerk of the machine tool can be represented as vector valued parametric function with respect to the same spline parameter *w*:

$$J(w) = [j_x, j_y, j_z, j_a, j_c] = [\dot{a}_x, \dot{a}_y, \dot{a}_z, \dot{a}_a, \dot{a}_c]$$

where:

$$j_\tau(w) = \dot{a}_\tau(w) = \frac{d^3\tau}{dt^3} = \frac{d^3\tau}{dw^3}\left(\frac{dw}{dt}\right)^3 + 3\frac{d^2\tau}{dw^2}\frac{dw}{dt}\frac{d^2w}{dt^2} + \frac{d\tau}{dw}\frac{d^3w}{dt^3}$$

$$= \tau_{www}(w)\dot{w}^3 + \tau_{ww}(w)\dot{w}\ddot{w} + \tau_w(w)\dddot{w}, \tau \in \{x, y, z, a, c\}.$$

Taking into account jerk limits for each of the axes, for jerk constraints we have:

$$|\tau_{www}(w)\dot{w}^3 + \tau_{ww}(w)\dot{w}\ddot{w} + \tau_w(w)\dddot{w}| \le j_{\tau\max}, \tau \in \{x, y, z, a, c\}. \quad (6)$$

2.2.3. PROBLEM DEFINITION

The methods for speed control of a machine have to concern on both geometric accuracy and machine productivity. To ensure good machine productivity we need to provide that the machine drives will move with highest feedrate according to previously described constraints (4), (5) and (6). This way the machine will provide the shortest travel time along the tool path. So the aim of the feedrate optimization problem is to maximize the feedrate or to minimize the travel time and generally it's defined as the minimization of total travel time along the entire path:

$$\min_w \int_0^1 \frac{dw}{\dot{w}}. \quad (7)$$

Later in this paper we will do a comparison analysis of three different approaches for feed profile generation.

2.3. PARAMETER COMPUTATION

The output of the velocity planning process is a feed profile relative to same parameter. If that parameter is machining time we get feedrate profile v(t) and the job is done because we only need to calculate drive positions according to obtained feedrate profile. But, if the parameter of the obtained feedrate profile is the path length s, or the spline parameter u we need to do additional interpolation in order to map the machine time to the appropriate parameter. There are different ways to deal with this interpolation s(t) or u(t) like first and second Taylor expansions algorithms, algorithms that deal with integral equations and others. For all of them the common thing is that they have to deal with some difficulties during interpolation when the speed of the feedrate profile is very small.

3. VELOCITY PLANNING METHODS

To deal with the velocity planning process we have developed and implemented 3 different look ahead algorithms in order to compare their characteristics: non-linear programming, heuristic and linear programming method. In the next section we will discuss them in details.

3.1. NON-LINEAR PROGRAMMING METHOD

Regarding geometry obtained from the inverse kinematics the 'movement' for each axis is parameterized using arc-length parameterization and the movement is represented using shape preserving cubic spline for each active axis. Here we use shape preserving cubic spline instead of simple B-spline because we need to follow the shape of the path obtained from the inverse kinematics (Fig3). Like this, using inverse kinematics, fitting shape preserving cubic spline and arc-length parameterization we get the input geometry for this kind of velocity planning method.

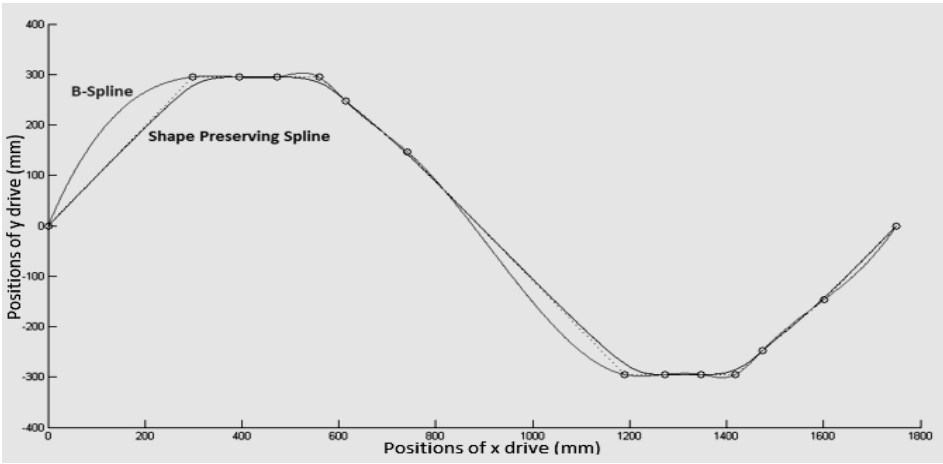


Fig. 3 Shape preserving v.s. B-spline

Next, with this method the feedrate profile is modeled as B-spline with respect to the tool displacement $\dot{s}(s)$. The only thing we do to obtain the feedrate profile that fulfills problem definition and constraint requirements is modulating B-spline control points by simply changing their position (Fig.4).

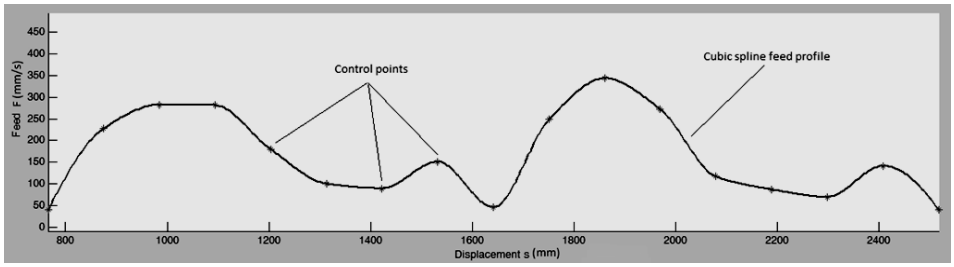


Fig. 4 Control points and feed profile using non-linear programming method

The basic of this method is moving the control points in order to minimize equation (7). To obtain the feedrate profile were used different optimization methods, like Dynamic Programming/Interior Point (DPIP) method and all different methods that can be called with MATLAB fmincon function. The results presented in section 4 are obtained from MATLAB Active-Set optimization method.

3.2. HEURISTIC METHOD

Similar as previous method after inverse kinematics is done, the input geometry for this heuristic method is represented as cubic spline with respect to tool path displacement for each axis. At the beginning the input geometry is segmented such as introducing a new spline parameter $u \in [0,1]$ for each segment. The extraction of the feedrate profile is done by modeling an S-curve with respect to time, on every path segment. By introducing the S-curves we ensure that by changing the invariable jerk the acceleration will be continuous. On Fig.5 is shown a kinematic profile of S-curve process that is usually divided in seven phases: phase with constant jerk, phase with constant acceleration, phase with constant jerk, phase with constant feedrate, phase with constant jerk, phase with constant acceleration and phase with constant jerk.

The basic and what gives strength to this algorithm is using of bisection search algorithm (Fig.6) in order to obtain a compatible feedrate. This compatible feedrate is derived from the kinematics compatibility conditions that checks if the segment has enough length to create S-curve with highest possible feedrate. If isn't long enough a dichotomy is used to determine the highest feedrate that will enable kinematic compatibility on each segment. Also this compatible feedrate determines the heuristic search space for obtaining a feasible feedrate that will satisfy velocity, acceleration and jerk constraints, and be declared as optimized feedrate. To obtain the final optimized feedrate once again is used a bisection search algorithm.

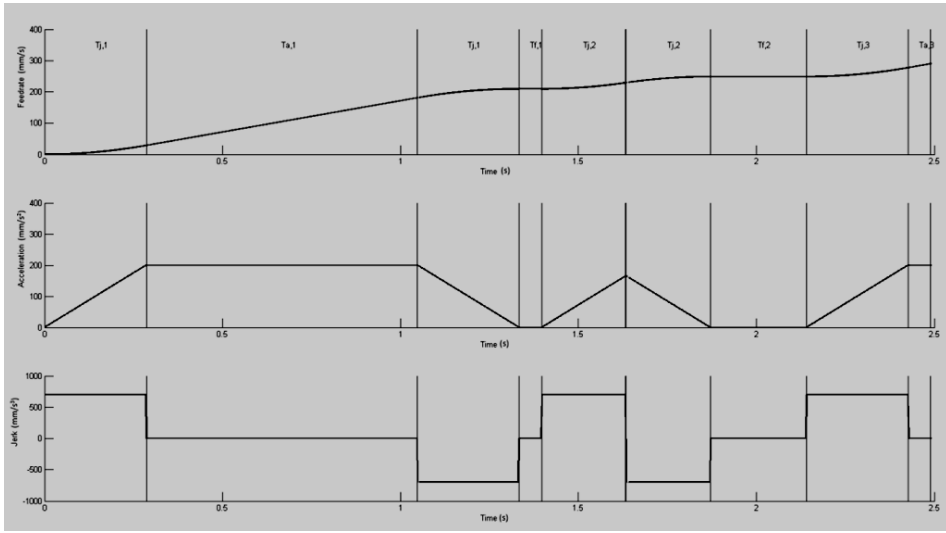


Fig. 5 S-curve feedrate profile

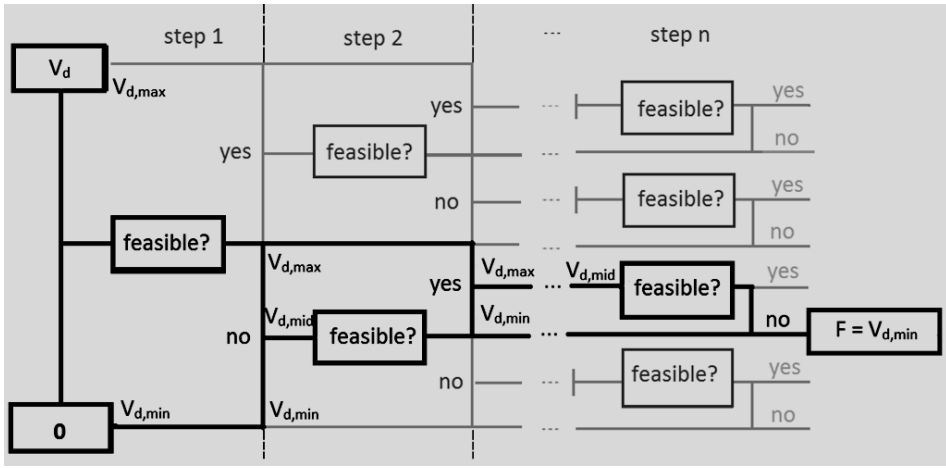


Fig. 6 Bisection search algorithm (dichotomy)

3.3. LINEAR PROGRAMMING METHOD

Here, each axis path obtained from the starting geometry is represented as cubic spline with respect to $u \in [0,1]$. The optimization is performed using MATLAB optimization method linprog. Since the problem as it is defined in section 2.2. is not linear, the first important thing done in this feed profile generating method is the linearization of the problem. To do this first a new variable is introduced:

$$q = \frac{(\frac{dx_1}{du})^2+(\frac{dx_2}{du})^2+...+(\frac{dx_n}{du})^2}{(\frac{dx_1}{dt})^2+(\frac{dx_2}{dt})^2+...+(\frac{dx_n}{dt})^2}$$

Using this, the problem is reduced to:

$$\min_q \int_0^1 \frac{du}{\sqrt{q}}$$

Subject to the constraints:

$$\begin{aligned} &0 \le q \le (\tfrac{v_{\tau\max}}{\tau_u(u)})^2, \tau \in \{x_1, x_2, ..., x_n\} \\ &|\tfrac{1}{2}\tfrac{d(q\tau r^2)}{d\tau}| \le a_{\tau\max}, \tau \in \{x_1, x_2, ..., x_n\} \\ &|(\tau''' q + (\tau' + \tfrac{\tau''}{2})q' + \tfrac{\tau'}{2}q'')\sqrt{d}| \le j_{\tau\max}, \tau \in \{x_1, x_2, ..., x_n\} \end{aligned} \tag{8}$$

Using these transformations and discretization of the search space (segment [0,1] $u_1, u_2, ..., u_n$) the problem 2.2.3. is reduced to:

$$\max_q (q_1, q_2, ..., q_n), \tag{9}$$

and the constraints from the formula (8) are reduced to:

$$\begin{aligned} &0 \le q_i \le (\tfrac{v_{\tau\max}}{\tau_u(u_i)})^2, |q_{i+1}\tau_{i+1}'^2 - q_i\tau_i'^2| \le 2A_\tau |\tau_{i+1} - \tau_i| \\ &|\sqrt{q_i^*}\alpha_{\tau i}q_{i-1} + \sqrt{q_i^*}\beta_{\tau i}q_i + \sqrt{q_i^*}\gamma_{\tau i}q_{i+1}| \le J_\tau(\tfrac{3}{2} - \tfrac{q_i}{2q_i^*}), i = 1, 2, ..., N-1 \end{aligned} \tag{10}$$

$$\begin{aligned} &|\tfrac{N^2\tau_2'\sqrt{q_1^*}q_2}{8\tau_1'}| \le J_\tau(\tfrac{3}{2} - \tfrac{q_1}{2q_1^*}); \\ &|\tfrac{N^2\tau_N'^2 - 2\sqrt{q_{N-1}^*}q_{N-1}}{8\tau_{N-1}'}| \le J_\tau(\tfrac{3}{2} - \tfrac{q_{N-1}}{2q_{N-1}^*}) \end{aligned} \tag{11}$$

where:

$$\alpha_{\tau i} = \frac{\tau_i'}{2\Delta u^2} - \frac{\tau_i'}{2\Delta u} - \frac{\tau_i''}{4\Delta u}, \quad \beta_{\tau i} = \tau'''i - \frac{\tau_i'}{\Delta u^2}, \quad \gamma_{\tau i} = \frac{\tau_i'}{2\Delta u^2} + \frac{\tau_i'}{2\Delta u} + \frac{\tau''i}{4\Delta u}$$

Now, the algorithm consists in:

- Founding solution $q_1^*, q_2^*, ..., q_{N-1}^*$ of the problem (9) subject to (10) - only velocity and acceleration constraints.
- Founding optimal solution of the problem (9) subject to (10) and (11) using $q_1^*, q_2^*, ..., q_{N-1}^*$.
- Determining velocity profile *v=v(u)*

4. RESULTS

Feed profile obtained from each of the algorithms for filament winding process on a tube.

Next few results will be given that compare algorithm computational time versus winding time obtained from the algorithms.

According to the previous theoretical analysis of the algorithms we expect largest computational time when non-linear programming method is used, since it amounts to

non-linear optimization, then the heuristic method which amounts to look-ahead search using trial and error method, and less computational time we expect in linear programming method.

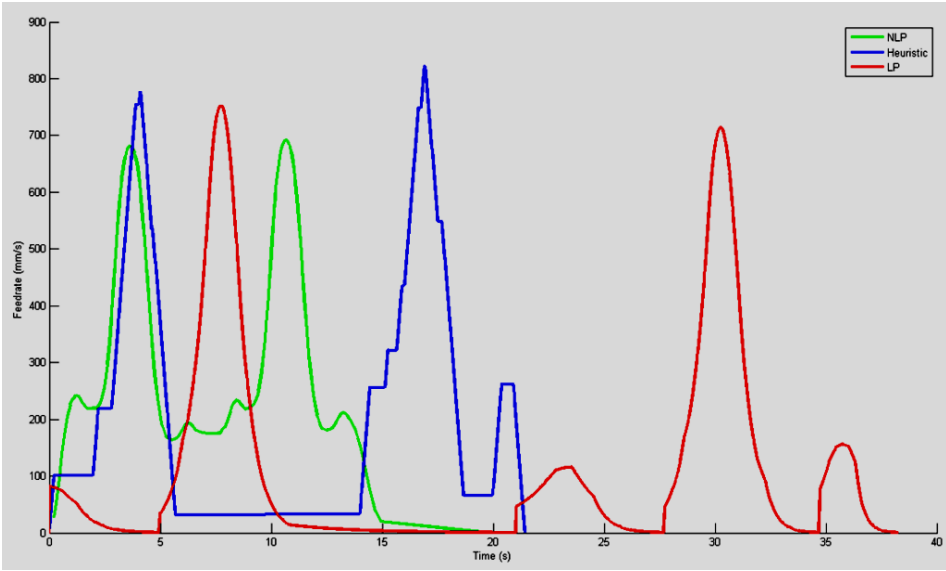


Fig. 7 Feedrate profiles obtained from tree different velocity planning methods

Example 1: Filament winding on a tube (length L=1450 mm) by machine with two drives

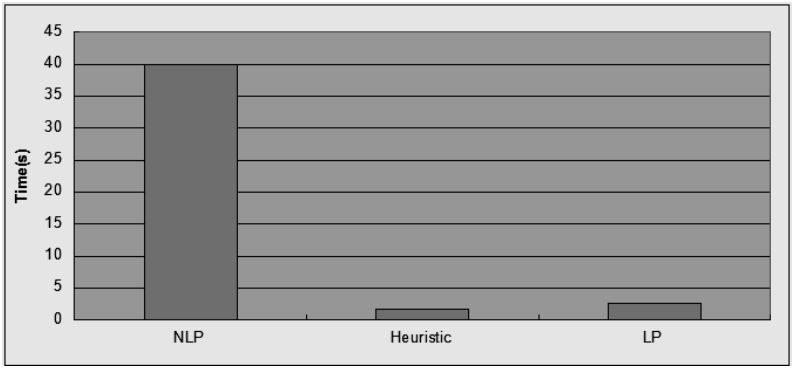


Fig. 8 Computational time for the tree different velocity planning methods

On the example 2 there can be seen that the computational time using the heuristic method is less than the time for algorithm evaluation in the linear programming method. That is because we can't predict computational time in these direct search methods, sometimes the optimal solution can be obtained with very little searching of the objective search space.

Example 2: Filament winding on a bigger tube (length *L*=9570 mm) by machine with two drives

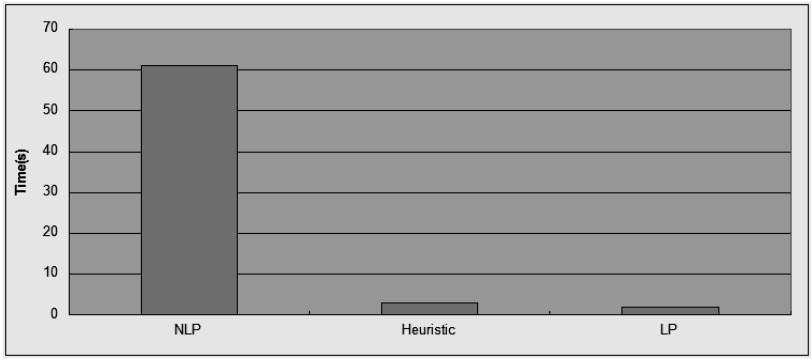


Fig. 9 Computational time for the tree different velocity planning methods

According to the winding time from each of the algorithms we expect that the winding time obtained from non-linear programming method will always be smaller than other two methods and the biggest winding time will be obtained from linear programming method. The heuristic method will give time between these two.

Example 1: Filament winding on a tube (length L=1450 mm) by machine with two drives

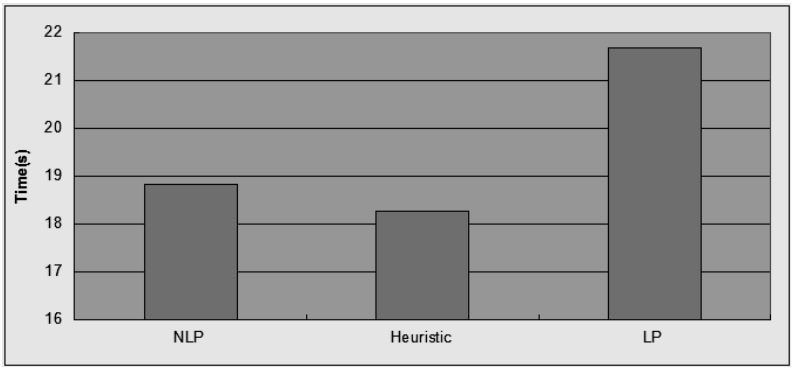


Fig. 10 Winding time obtained from the tree different velocity planning methods

Example 2: Filament winding on a bigger tube (length *L*=9570 mm) by machine with two drives

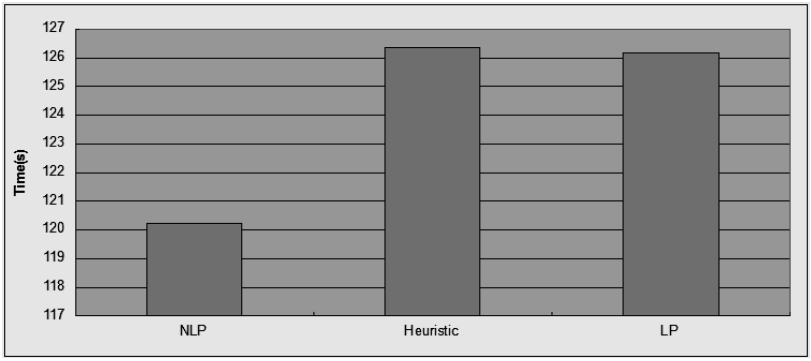


Fig. 11 Winding time obtained from the tree different velocity planning methods

5. CONCLUSION

In this paper is presented a detailed process for speed control that lead as to a coordinated axis motion that is accurate, smooth and time-optimal within the limits imposed by drives dynamics.

Today high speed machines require very high speed for drive's movement in order to achieve good productivity, which can be harmful for the machine. To overcome this problem we need to impose some dynamical limitations of the drives. According to the imposed limits for the velocity, acceleration and jerk of all machine drives this paper presents tree different solutions for minimizing the winding time while making best use of the kinematical characteristics.

We give thoroughly explanation how to obtain the required input geometry for the velocity planning process when the starting geometry is presented as discrete sequence of positions and orientations of the machine tool or presented as a continuous curve like B-spline, NURBS and cubic spline. In this section, we dedicated special attention to parameterization because the speed depends on the type of parameterization.

Also we give detailed explanation for the implementation of the tree methods proposed as a solution for the velocity planning process. In this paper we compare these tree methods by their computational time and their winding time. From the conducted comparison we can conclude that all tree methods we have developed and implemented are valid. The non-linear programming method and the heuristics method give good results as winding time. When the speed control algorithm can be executed offline than is better to use non-linear programming method because the winding time obtained whit this method is generally smaller than the winding time obtained from the other two methods. But when the speed control algorithm have to be executed in real time than is better to use heuristic method because the winding time is not much bigger compared to one of the non-linear programming method, but the computational time is significantly smaller.

In our future research, we should improve the heuristic approach. Developing and implementation of some algorithm of the class "critical point methods" fits in our research plans as well.

References

[1] W. Fan, X-S. Gao, K. Zhang, *Time-Optimal Interpolation for Five-axis CNC Machining along Parametric Tool Path based on Linear Programming*, Mathematics Mechanization Research Preprints, Vol. 31, KLMM, Chinese Academy of Sciences, 2012, pp. 21-42.

[2] M. M-T. Heng, *Smooth and Time-Optimal Trajectory Generation for High Speed Machine Tools*, Master thesis - University of Waterloo, Waterloo, Ontario, Canada, 2008

[3] X. Beudaert, S. Lavernhe, C. Tournier, *5-axis local corner rounding of linear tool path discontinuities*, International Journal of Machine Tools & Manufacture 73 (2013), 2012, pp. 9 – 16

[4] J. E. Bobrow, S. Dubowsky, J. S. Gibson, *Time-Optimal Control of Robotic Manipulators Along Specified Paths*, The International Journal of Robotics Research, Vol. 4, No 3, Massachusetts Institute of Technology, 1985

[5] A. Siu, *Real time trajectory generation and interpolation*, Master thesis - University of British Colombia, Vancouver, 2011

[6] B. Sencer, Y. Altintas, E. Croft, *Feed optimization for five-axis CNC machine tools with drive constraints*, International Journal of Machine Tools & Manufacture 48, Elsevier, 2008, pp. 733–745

[7] S-H. Suh, S-K. Kang, D-H. Chung, I. Stroud, *Theory and Design of CNC Systems*, Springer, 2008

[8] K. Erkormaz, Y. Altintas, *Quintic Spline Interpolation With Minimal Feed Fluctuation*, Journal of Manufacturing Science and Engineering, Vol. 127, 2005, pp.339-349

[9] E. B. Msaddek, Z. Bouaziz, M. Baili, *Influence of interpolation type in high-speed machining (HSM)*, The International Journal of Advanced Manufacturing Technology, vol. 72 (N^{o} 1-4), 2014, pp.289-302

[10] J-Y. Lai, K-Y. Lin, S-J. Tseng, W-D. Ueng, *On the development of a parametric interpolator with confined chord error, feedrate, acceleration and jerk*, The International Journal of Advanced Manufacturing Technology, Volume 37, Issue 1-2, 2008, pp.104-121

[1)] Institute for Advanced Composites and Robotics, Prilep, Macedonia
E-mail address: igord@iacr.edu.mk
[2)] Institute for Advanced Composites and Robotics, Prilep, Macedonia
E-mail address: ssamak@iacr.edu.mk
[3)] Institute for Advanced Composites and Robotics, Prilep, Macedonia
E-mail address: dijanac@iacr.edu.mk
[4)] Institute for Advanced Composites and Robotics, Prilep, Macedonia
E-mail address: mirjanat@iacr.edu.mk
[5)] Institute for Advanced Composites and Robotics, Prilep, Macedonia
E-mail address: filipk@iacr.edu.mk