Linear programming feedrate optimization - Adaptive path sampling and feedrate override

Petr Petráček (P.Petracek@rcmt.cvut.cz)
Czech Technical University in Prague: Ceske Vysoke Uceni Technicke v Praze
Bořivoj Vlk
Czech Technical University in Prague: Ceske Vysoke Uceni Technicke v Praze
Jiří Švéda
Czech Technical University in Prague: Ceske Vysoke Uceni Technicke v Praze

Research Article

Keywords: Linear programming, Feedrate optimization, Feedrate override, NURBS curve, Curve sampling

Posted Date: July 8th, 2021

DOI: https://doi.org/10.21203/rs.3.rs-644788/v1

The International Journal of Advanced Manufacturing Technology manuscript No.
(will be inserted by the editor)

Linear programming feedrate optimization

Adaptive path sampling and feedrate override

Petr Petráček · Bořivoj Vlk · Jiří Švéda

Received: date / Accepted: date

Abstract This paper focuses on two aspects of feedrate optimization via linear programming methods. Namely, the effect of curve sampling on time optimality of the resultant feedrate profile and a method of feedrate profile adaptation in response to a feedrate override command. A comparison of three distinct curve sampling approaches (uniform in parameter, uniform in arc length and curvature adaptive) is performed on a series of standard tool path curves. Results show that the curvature-adaptive sampling approach leads to substantial machining time reduction for tool path curves displaying high degree of curvature variation. Secondly, a method by which a new feedrate profile can be calculated in response to a feedrate override command is developed. The method formulates a new set of boundary conditions on the control point sequence of the feedrate curve in such a way that the resulting profile is guaranteed to coincide with the currently active profile up to the moment of override command, while minimizing the arc length necessary for transition to the newly commanded feedrate.

Keywords Linear programming · Feedrate optimization · Feedrate override · NURBS curve · Curve sampling

Petr Petráček (ORCID:0000-0002-3955-4698) (corresponding author)
Research Center of Manufacturing Technology (RCMT), Department of Production Machines and Equipment, Faculty of Mechanical Engineering, Czech Technical University in Prague
E-mail: P.Petracek@rcmt.cvut.cz

Bořivoj Vlk (ORCID:0000-0002-1570-6597)
Research Center of Manufacturing Technology (RCMT), Department of Production Machines and Equipment, Faculty of Mechanical Engineering, Czech Technical University in Prague
E-mail: B.Vlk@rcmt.cvut.cz

Jiří Švéda (ORCID:0000-0003-0271-2899)
Research Center of Manufacturing Technology (RCMT), Department of Production Machines and Equipment, Faculty of Mechanical Engineering, Czech Technical University in Prague
E-mail: J.Sveda@rcmt.cvut.cz

1 Introduction

Interpolator is a component of the computer numerical control (CNC) system, which governs axial servo drives by providing position commands. Vectors of axis positions are being sent to servo drives at specified frequency. In a precision and high-speed machining environment, interpolator is required to optimize axial movement, so that maximum accuracy and shortest machining time are achieved. Considerable effort has been put by researchers into developing algorithms capable of interpolating tool paths which are determined by parametric curves such as NURBS. NURBS curves (and surfaces) are a staple in computer assisted design due to their compact representation and efficient evaluation. The necessity of generating feedrate profiles on NURBS tool paths has led to the development of a variety of distinctive interpolation methods.

To create efficient feedrate profiles for NURBS curve tool paths, it is necessary to consider the relationship between the curve's parameter and its arc length. This relationship does not have a closed form solution in general and disregarding it can lead to undesirable feedrate oscillations. This problem was typically solved using Taylor series expansion (see, e.g. [16]). This, of course, introduces errors caused by omitting higher order terms. As a possible solution to this problem, the so-called Pythagorean hodograph curves have been proposed as an alternative to NURBS curve representa-

tions (for a comprehensive study of Pythagorean hodograph curves see [15]). Pythagorean hodograph curves enjoy the nice property of polynomial dependence of arc length on curve parameter, making them particularly useful for CNC interpolations. However, due to the widespread use of NURBS curves in CAD systems, Pythagorean hodograph curves still have not found wide application in common practice.

In order to solve the problem of parameter-to-arc length relationship in NURBS curves, it is necessary to devise approximation methods. In [10], the authors develop a recursive approximation algorithm to interpolate quintic spline tool paths with constant distance increment. The feedrate profile is constructed by varying the interpolation period, which is then reconstructed in the control loop period by fitting a fifth-degree polynomial. Another method that approximates the relationship between the spline parameter and its arc length is presented in [11]. The main idea is to use a seventh-degree polynomial (feed correction polynomial) to approximate this relationship. The coefficients of this polynomial are estimated using least squares. This initial approximation is then further refined using the Newton-Raphson iterative method. This method has been further modified in [17], where the parameter-to-arc length relationship is first approximated for a series of discrete points using the adaptive Simpson rule and this discrete representation is then used to fit a series of feedrate polynomials depending on a maximal allowed deviation. Due to the adaptive nature of this algorithm, it can approximate the parameter-to-arc length relationship with arbitrary precision. This particular algorithm was used for the purposes of this paper.

The most common approaches to feedrate optimization found in current literature are: composing the feedrate profile by connecting S-shaped feedrate profiles with piecewise constant jerk (constructed analytically or by using FIR filters), brute force optimization methods and utilization of linear programming (LP). The last mentioned approach combines computational efficiency with the ability to construct nearly time-optimal feedrate profiles. First presented in [14], this method of feedrate optimization is based on the use of tool path discretization followed by reformulation of the feedrate optimization problem where the feedrate function is represented by a B-spline curve whose control points serve as the free variables. Finding the control points poses a non-linear optimization problem, which requires the introduction of the so-called pseudojerk, that serves as an upper bound to the jerk inequality, thus allowing to linearize the optimization problem. This upper bound is precomputed using only axial velocity and acceleration limits. The proximity of this solution to true optimal feedrate profile is limited by the finite number of control points for feedrate profile curve and the pseudo-jerk relaxation. This method was further expanded on in [13], where a windowing based parallelization method was developed in order to further improve the computational efficiency of the original algorithm.

It is of note, that the methods of linear programming have also been applied to the feedrate optimization problem in an earlier article [1], although in a substantially different way. The feedrate profile on a spline toolpath was defined as a function of time and was obtained by minimizing a square integral of jerk via manipulation of time durations of selected path segments. This solution did not, however, consider the influence of the parameter-to-arc length relationship. For a similar approach using time parametrization, see [19].

Another computationally efficient method to construct the feedrate profile for NURBS tool paths is based on connecting piecewise jerk constant S-curve type feedrate transition segments. Feedrate can be modulated in this way to comply with kinematic limits which are evaluated at a fixed set points (see [12]). Similarly, in [17] feedrate profiles for spline tool paths are generated by connecting constant feedrate segments by S-shaped curve transition segments according to a heuristic algorithm. Axis jerk and acceleration limits are evaluated at knot locations. For a given portion of the spline trajectory, which is bounded by two knots, the maximum feedrate is calculated based on these constraints. The algorithm performs a search for maximum feedrate that is both achievable and respects constraints in all portions of the spline trajectory.
In [3], an axis movement smoothing algorithm for 5-axis milling is developed, which utilizes a heuristic algorithm to conform to contour tolerance limits. In [26], the authors describe a look-ahead algorithm, including analytical expressions, for interpolating linear segments. The article [25] presents a corner smoothing technique for five-axis machining using micro splines inserted between consecutive linear blocks with synchronized position and tool orientation.

An alternative approach to generating feedrate profiles relies on the application of sequences of FIR filters. The authors of [6] show that applying a sequence of moving average filters to the feedrate impulse produces a feedrate function that is equivalent to the analytically expressed S-shaped feedrate profile. Their work provides a relationship between feedrate profile and its frequency domain, which can be utilized to suppress vibrations on given resonant frequencies by adjusting filter parameters. In [2], analytically generated acceleration limited feedrate profile is combined with the FIR approach to generate feedrate profile for arbitrary ve-

locity and acceleration conditions. Finally, [24] extends this method by including circular and linear toolpath blending capability with confined contour error. A transition between linear and circular segments can thus be performed at non-zero feedrates by controlling the convolution overlap time of two consecutive feed impulses. This method was recently extended to 5-axis machining in [23].

Iterative method, referred to as VPOp, can be applied to interpolate NURBS toolpaths even in 5-axis milling. This method achieves short (shortest among all the mentioned approaches) machining times while respecting axis acceleration and jerk constraints. However, its high computational demands severely limit its real world application. Articles [4] and [5] use the VPOp algorithm to interpolate directly into the parametric space of the surface of the workpiece CAD model. This original approach removes the issue of CAM tolerance and interpolation tolerance stacking up.

To summarize the above comparison, linear programming methods produce results which are close to optimal even on complex curves. Given a sufficient number of discretization points, the method is able to respect all the given kinematic limits while maintaining computational efficiency. The biggest disadvantage is the necessity of linearization of the jerk constraint, which results in loss of optimality of the solution. Nevertheless, when compared to other methods the conciseness of the mathematical formulation, ability to simultaneously incorporate both tangential and axial limits on kinematic variables and the possibility of parallelization all make linear programming methods a favorable choice.

This paper is organized as follows: Section 2 reviews the formulation of the linear programming optimization problem (Section 2.1) as well as knot vector construction and evaluation point selection given a general sampling of the toolpath curve (Section 2.2).

In Section 3, three types of curve sampling (equidistant in curve parameter, equidistant in arc length and adaptive) are presented and the effects of sampling approach on time optimality of the resultant feedrate are discussed based on a comparison for several test curves. The main original result of this section is that an application of a suitable adaptive sampling method can lead to a significant decrease of machining time.

Section 4 is dedicated to an original method of feedrate override for linear programming feedrate optimization. The method is divided into two distinct cases: override to a higher commanded feedrate (Section 4.2) and override to a lower commanded feedrate (Section 4.3). This method describes how, and under what conditions a new feedrate profile implementing the override command can be obtained given curve parameters and the currently active feedrate profile. The method is suitable for application purposes due to low memory requirements and relative ease of implementation.

The article is concluded with several closing remarks in Section 5 and two appendices: Appendix 7 providing definitions of the test path curves used in Sections 3 & 4 and Appendix 8 in which a definition of an auxiliary function is given.

2 Feedrate optimization - a linear programming problem

This chapter recalls the formulation of feedrate optimization via linear programming (Section 2.1) while briefly discussing knot vector construction and the selection of evaluation points (Section 2.2).

2.1 Problem formulation

Assume that the tool path is defined as a NURBS curve with at least C^2 continuity (in practice this typically means a curve of order three of five)

$$\boldsymbol{r}(u) = [x(u), y(u), z(u)], \quad u \in [0, 1]$$

In order to formulate the relations between velocity, acceleration and jerk along the curve, it is beneficial to introduce the arc length parametrization so that one can also express $\boldsymbol{r}$ as

$$\boldsymbol{r}(s) = [x(s), y(s), z(s)], \quad s \in [0, L]$$

where L represents the total length of the curve. Let $\boldsymbol{v}_{max}, \boldsymbol{a}_{max}, \boldsymbol{j}_{max}$ denote the maximal axial limits of velocity, acceleration and jerk, respectively. Furthermore, let $\bar{v}_{max}$, $\bar{a}_{max}$, $\bar{j}_{max}$ denote the maximal limits of tangential velocity, tangential acceleration and tangential jerk, respectively. The axial velocity, acceleration and jerk can be expressed as:

$$\boldsymbol{v} = \frac{d\boldsymbol{r}}{dt} = \boldsymbol{r}'\dot{s}$$
$$\boldsymbol{a} = \frac{d^2\boldsymbol{r}}{dt^2} = \boldsymbol{r}''\dot{s}^2 + \boldsymbol{r}'\ddot{s}$$
$$\boldsymbol{j} = \frac{d^3\boldsymbol{r}}{dt^3} = \boldsymbol{r}'''\dot{s}^2 + 3\boldsymbol{r}''\dot{s}\ddot{s} + \boldsymbol{r}'\dddot{s}$$

where $\boldsymbol{r}$, $\boldsymbol{r}'$ and $\boldsymbol{r}''$ denote the derivatives of $\boldsymbol{r}$ with respect to s, while $\dot{s}$, $\ddot{s}$ and $\dddot{s}$ denote the tangential velocity (feedrate), tangential acceleration and tangential jerk, respectively. The feedrate optimization problem,

i.e. derivation of time-optimal feedrate profile, can then be formulated as:

$$\text{maximize} \int_0^L \dot{s}\, ds$$

such that

$$\begin{aligned}
\boldsymbol{v}_{max} &\geq |\boldsymbol{r}'\dot{s}| \\
\boldsymbol{a}_{max} &\geq |\boldsymbol{r}''\dot{s}^2 + \boldsymbol{r}'\ddot{s}| \\
\boldsymbol{j}_{max} &\geq |\boldsymbol{r}'''\dot{s}^2 + 3\boldsymbol{r}''\dot{s}\ddot{s} + \boldsymbol{r}'\dddot{s}| \\
\bar{v}_{max} &\geq |\dot{s}| \\
\bar{a}_{max} &\geq |\ddot{s}| \\
\bar{j}_{max} &\geq |\dddot{s}|
\end{aligned}$$

In order to linearize the above problem, the authors of [14] (see also [13]) apply the following substitution:

$$\dot{s}^2 = q(s) = \sum_{i=1}^{K} N_{i,2}(s) \cdot a_i = \boldsymbol{N}(s) \cdot \boldsymbol{a},$$

i.e. the square of feedrate is expressed as a cubic B-spline where $\boldsymbol{a} = [a_1, \ldots, a_K]$ is the vector of control points and $[N_{1,2}, \ldots, N_{K,2}]$ are the basis functions.

Using the above substitution the optimization problem can be reformulated as

$$\text{maximize} \int_0^L q(s)\, ds \tag{1}$$

such that

$$(\boldsymbol{v}_{max})^2 \geq |\boldsymbol{r}'|^2\, q \tag{2a}$$

$$\boldsymbol{a}_{max} \geq \left|\boldsymbol{r}''q + \frac{1}{2}\boldsymbol{r}'q'\right| \tag{2b}$$

$$\boldsymbol{j}_{max} \geq \left|\boldsymbol{r}'''q + \frac{3}{2}\boldsymbol{r}''q' + \frac{1}{2}\boldsymbol{r}'q''\right|\sqrt{q} \tag{2c}$$

$$(\bar{v}_{max})^2 \geq q \tag{2d}$$

$$\bar{a}_{max} \geq \left|\frac{1}{2}q'\right| \tag{2e}$$

$$\bar{j}_{max} \geq \left|\frac{1}{2}q'\right|\sqrt{q} \tag{2f}$$

Except for the square root of q appearing in the inequalities (2c) and (2f), the above optimization problem can be posed as a linear optimization problem in which the control points a_i represent the free variables. To overcome the nonlinearity, the optimization problem (1) is first solved without the tangential and axis jerk constraints, leading to the LP formulation:

$$\text{maximize } \boldsymbol{c}^T\widehat{\boldsymbol{a}} \text{ subject to: } \begin{matrix}\widehat{\boldsymbol{M}}\widehat{\boldsymbol{a}} \leq \widehat{\boldsymbol{b}} \\ \widehat{\boldsymbol{a}} \geq 0,\end{matrix}$$

where the vector $\boldsymbol{c}^T$ is a uniformly distributed weighting vector. The matrix $\widehat{\boldsymbol{M}}$ is a constant matrix, whose terms are obtained via evaluation of the constraint equations (2a), (2b), (2d) and (2e) at a set of evaluation points. In this way a solution $\widehat{q}$ (so-called pseudojerk) is obtained which realizes a larger feedrate than any other feasible solution. In the next step, the following constraints are substituted for the original jerk constraints (2c) and (2f), respectively.

$$\boldsymbol{j}_{max} \geq \left|\boldsymbol{r}'''q + \frac{3}{2}\boldsymbol{r}''q' + \frac{1}{2}\boldsymbol{r}'q''\right|\sqrt{\hat{q}} \tag{3a}$$

$$\bar{j}_{max} \geq \left|\frac{1}{2}q'\right|\sqrt{\hat{q}} \tag{3b}$$

Thus, the approximate solution of the original optimization problem (1) can be formulated as :

$$\text{maximize } \boldsymbol{c}^T\boldsymbol{a} \text{ subject to: } \begin{matrix}\boldsymbol{M}\boldsymbol{a} \leq \boldsymbol{b} \\ \boldsymbol{a} \geq 0.\end{matrix}$$

2.2 Linear programming - evaluation point and knot vector construction

To the authors' best knowledge, none of the previous works describe how exactly should the knot vector and evaluation points of the feedrate curve be constructed given a sampling of the toolpath $\boldsymbol{r}$. The process is thus briefly described below for the reader's convenience. Given a sampling in the arc-length parameter s

$$\Gamma = \{s_1 = 0, s_2, \ldots, s_{K-1}, s_K = L\}, \quad K \in \mathbb{N}$$

the knot vector of the squared-feedrate function q of order d is defined as

$$U = \{\underbrace{0, \ldots, 0}_{\times d}, s_2, \ldots, s_{K-1}, \underbrace{L, \ldots, L}_{\times d}\}. \tag{4}$$

Next, a sequence of evaluation points needs to be determined such that for every basis function $N_{i,d}$, there exists at least one point in this sequence lying in its support. A suitable choice is the sequence of *Greville points* (*Greville abscissae*) defined as

$$G = \{g_1, \ldots, g_{N-d}\}, \quad N \in \mathbb{N}$$

where N is the number of control points of q and d is the order of q (defined as degree of $q+1$) and

$$g_i = \frac{1}{d-1}\left(u_{i+1} + \cdots + u_{i+d-1}\right), i \in \{1, \ldots, (N-d)\}. \tag{5}$$

The Greville point g_i generally lies near the parameter value corresponding to the maximum of the basis function $N_{i,d}$ [21, p. 512]. The main computational advantage of this particular selection of evaluation points

is that the matrices $\widehat{\boldsymbol{M}}$ and $\boldsymbol{M}$ (which comprise the evaluations of the respective constraints at points of G) become sparse band matrices with band equal to d, thus increasing stability and efficiency of linear programming optimization methods.

3 Curve sampling methods

The simplest way to sample the toolpath curve is to sample the curve parameter interval uniformly, i.e.

$$\Gamma_M^u = \left\{0, \frac{1}{M-1}, \ldots, \frac{M-2}{M-1}, 1\right\}, \quad M \in \mathbb{N}.$$

This sampling technique is denoted as PAR in the following.

Somehow surprisingly, very little attention has been dedicated to the question of sampling of the underlying toolpath curve and its effect on time optimality of the resulting feedrate profile. To the author's best knowledge, only sampling uniform in the curve parameter (PAR) has been considered in literature dealing with feedrate optimization via the LP approach (see e.g. [14], [7], [9], [22] and [27]). It is of note, however, that in [14] the authors remark that a nonuniform subdivision of the tool path that would take into consideration its local shape could result in a better performance of the algorithm.

An alternative to the uniform parameter sampling is the uniform arc length sampling (denoted as LEN)

$$\Gamma_M^s = \{s_1 = 0, \ldots, s_M = 1\},$$

such that

$$\text{arc length of } r|_{[s_i, s_{i+1}]} = \frac{L}{M}, \; i \in \{1, \ldots, (M-1)\}.$$

To apply this sampling technique the information about the parameter to arc length relation is required. As this relation cannot be computed analytically in general, it is necessary to find a sufficiently close approximation. To this end, the method described in [17] was used (note that this does not pose an extra requirement, as the arc length evaluation is also used in the formulation of the LP optimization problem).

Intuitively, one would expect that a sampling technique would produce a set of sampling points with density that is proportional to the curvature of the sampled curve, while maintaining some minimal point density in areas of zero curvature so that the degree of control over the squared feedrate function q correlates with the local geometric properties of the toolpath curve.

An adaptive algorithm with precisely those properties was proposed (along with a detailed description of its implementation) in [18, p. 1471-1476]. The inputs of this algorithm are: m, the desired number of sampling points, non-negative weights λ_s, λ_κ satisfying $\lambda_s + \lambda_\kappa = 1$ and a threshold parameter $\varepsilon_\kappa > 0$ denoting a minimum integrated curvature amount to be taken into account by the algorithm. The values of λ_s and λ_κ represent contributions of arc length and curvature, respectively, to the density of the set of sampling points. Thus, the combination $\lambda_s = 1$, $\lambda_\kappa = 0$ results in a uniform arc length sampling, while the combination $\lambda_s = 0$, $\lambda_\kappa = 1$ results in a sampling set with zero point density in areas of zero curvature.

In the following, this adaptive algorithm is referred to as ADA.

For a comparison of the outputs of the three sampling variants see Figures 1, 2, 3 and 4.

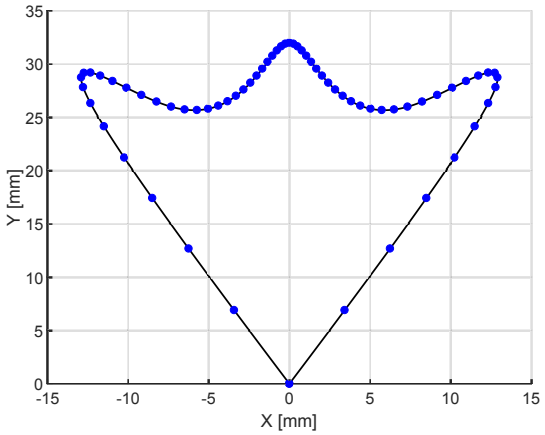

Fig. 1 PAR sampling with $m = 65$

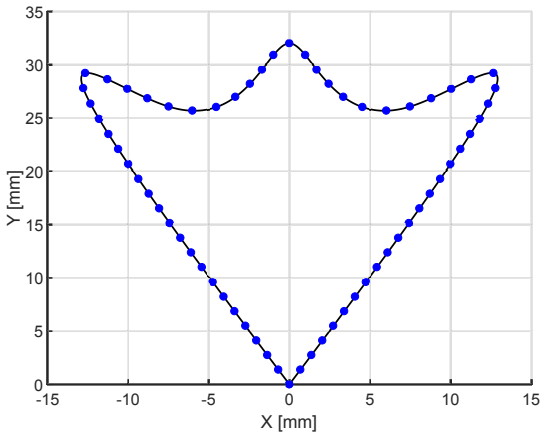

Fig. 2 LEN sampling with $m = 65$

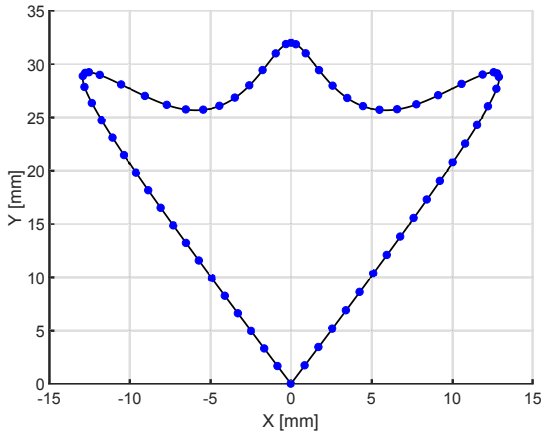

Fig. 3 ADA sampling with $m = 65$, $\lambda_s = 0.8$, $\lambda_\kappa = 0.2$

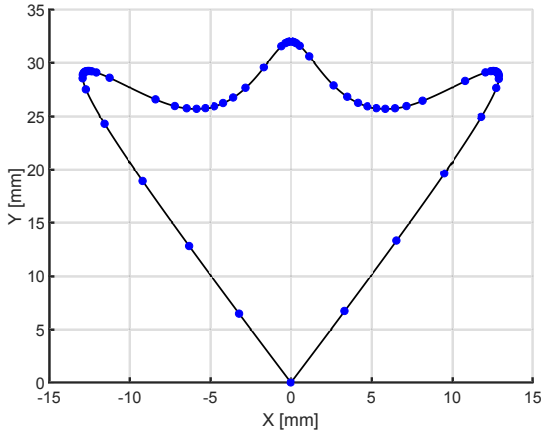

Fig. 4 ADA sampling with $m = 65$, $\lambda_s = 0.2$, $\lambda_\kappa = 0.8$

3.1 Sampling techniques - comparison of results

In this section, the effects of the PAR, LEN and ADA sampling techniques on the total machining time of the LP optimization output are presented and discussed. The algorithm was programmed in Matlab2017a software in combination with C++ code for the LP optimization and NURBS curve evaluation via the MEX interface. The *COIN-OR Linear programming solver* [8] has been used to solve the LP optimization task, while the C++ *openNURBS®* library [20] has been used to construct the NURBS curves and evaluate their derivatives. All computations were performed on a computer with an Intel® Core™ i7-7700K processor and Windows 10 operating system.

In order to compare the optimization results several testing curves have been used. These include the Trident curve, the Butterfly curve, the Pentacle curve and the Phobos curve (see Appendix A, Figures 14–17). With the exception of the Phobos curve, all of the testing curves have been previously used in articles concerning feedrate interpolation and optimization. All the curves are third degree continuous and display different behaviors regarding maximal and minimal values of curvature and its rate of variation. The Trident and Phobos curve comprise segments of zero curvature and segments of slowly varying curvature. The curvature of the Pentacle curve varies slowly, while the Butterfly curve displays both highest absolute values of curvature and highest rates of its variation. The Phobos curve represents a spline smoothed contour curve of a blade cross section designed in a commercial CAD system.

For each curve, the LP feedrate optimization has been performed with the kinematic limits configuration presented in Table 1. This configuration corresponds to one used on a AXA VCC 1200 machining center equipped with a MEFI CNC872 iTQ-E numerical control system (developed in part by the authors). Each curve was then sampled with increasingly larger values of sampling density (defined as number of points per mm of arc length). For every such sampling density the PAR, LEN and ADA sampling methods were used, the respective machining times were recorded and the relative percentage differences (rounded to the nearest percentage point) of the PAR and LEN methods with respect to the ADA method were calculated. The parameters of the ADA method were chosen as $\lambda_s = \lambda_\kappa = 0.5$ and $\varepsilon_\kappa = 1 \cdot 10^{-3}$. The results for individual curves are presented in Figures 5, 6, 7 and 8. A comparison of feedrate functions for the Butterfly and Trident curves is presented in Figures 9 and 10, respectively.

Table 1 Kinematic limits configuration

Parameter	Value	Units
$\bar{v}_{max}$	10	m/min
$\bar{a}_{max}$	1500	mm/s^2
$\bar{j}_{max}$	16000	mm/s^3
$\boldsymbol{v}_{max}$	[10, 10, 10]	m/min
$\boldsymbol{a}_{max}$	[1500, 1500, 1500]	mm/s^2
$\boldsymbol{j}_{max}$	[16000, 16000, 16000]	mm/s^3

The results presented in Figures 5-8 support the following conclusions: For curves with low curvature variation such as the Trident, Pentacle and Phobos curves, the ADA sampling technique leads to machining times that are comparable (±5%) with the PAR and LEN methods, while typically being a few percent faster. On the other hand, for curves with high curvature variation, such as the Butterfly curve, the ADA sampling technique leads to machining times that are faster (up to 12%) than the LEN and ADA methods. This behavior is due to the higher density of knot points of the squared feedrate function in areas of high curvature.

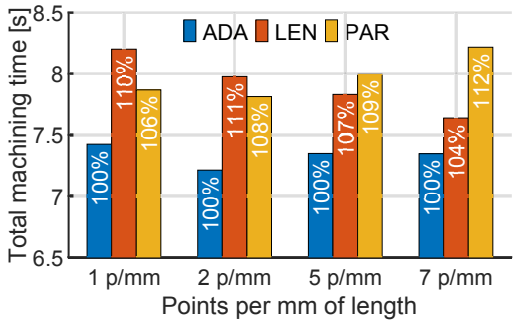

Fig. 5 Butterfly curve - sampling method comparison

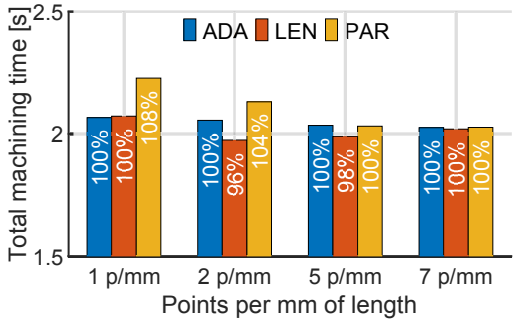

Fig. 6 Trident curve - sampling method comparison

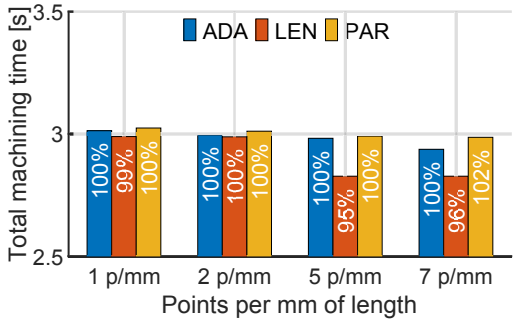

Fig. 7 Pentacle curve - sampling method comparison

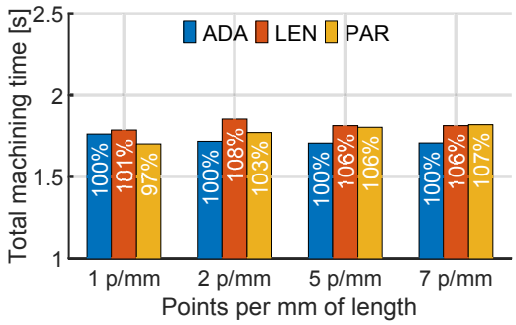

Fig. 8 Phobos curve - sampling method comparison

Thus, the feedrate profile can reflect the local changes of toolpath geometry more closely in these regions, leading to shorter machining times (while still respecting the kinematic limits).

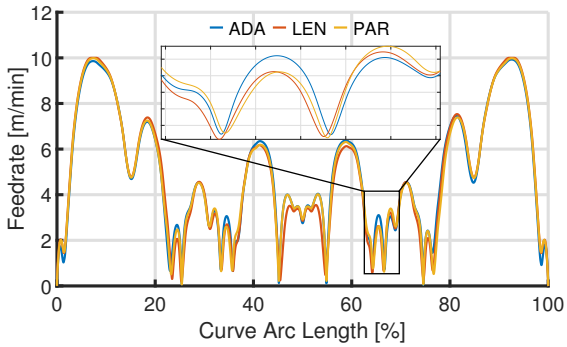

Fig. 9 Butterfly curve - feedrate function comparison at 1 p/mm sampling density

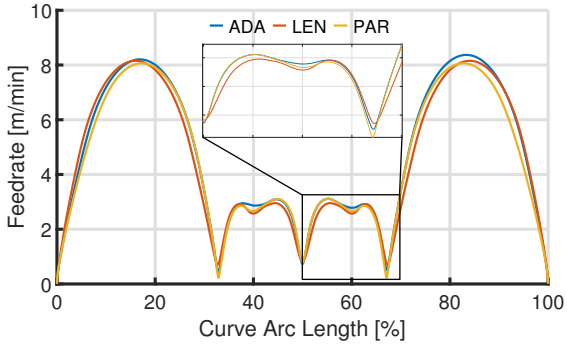

Fig. 10 Trident curve - feedrate function comparison at 1 p/mm sampling density

In conclusion, the ADA sampling technique in combination with the LP optimization approach described in Section 2.1 can lead to significantly shorter machining times when interpolating spline toolpaths with both high curvature variation and high maximal curvature. Such toolpaths are typically encountered in practice in machining of injection molds and in side milling (specifically in trimming operations).

4 Feedrate override

One of the standard commands in machining is the *feedrate override* command. With this command, the machine tool operator can change the value of commanded feedrate F_{cmd} to a different value expressed as a percentage of the originally programmed value.

The aim of this section is to present techniques by which the squared feedrate function that was obtained using methods described in Section 2 can be appropriately modified in response to an issued feedrate override command.

This chapter is divided into three section: notation is summarized in Section 4.1, method of override accel-

eration is presented in Section 4.2 and, finally, method of override deceleration is presented in Section 4.3.

4.1 Notation

Most of the notation in the following sections follows that introduced in Section 2. Parameters associated with pseudojerk are distinguished by the use of a hat symbol. The knot vector of the squared feedrate curve is denoted by q and the evaluation point vector is denoted by g. The number of control points is denoted by N, while the number of evaluation points is denoted by K (note that $K = (N - d)$, see (5)). For the sake of brevity the feedrate and acceleration at the i-th evaluation point are denoted by $F(i)$ and $a(i)$, respectively. The currently active commanded feedrate is denoted by F_{cur}, while the commanded feedrate imposed by feedrate override is denoted as F_{high} in case of override acceleration and F_{low} in case of override deceleration.

Subscripts are used to distinguish parameters pertaining to individual feedrate profiles. Specifically, parameters of the currently active feedrate profile are denoted by the subscript *cur* and parameters related to the feedrate profile realizing the override command are denoted by the subscript *ovr*. Parameters related to feedrate profiles globally limited by the override command feedrate in cases of override acceleration and deceleration are distinguished by subscripts *high* and *low*, respectively.

Specific control point indices defined in the subsequent section are denoted by I_{xyz}, while evaluation point indices are denoted by J_{xyz}. Finally, the phrase "up to" means "up to but not including" in the following text.

4.2 Override acceleration

The process of override transition to a new feedrate limit $F_{high} = \alpha_{ovr} \cdot F_{cur} > F_{cur}$ is performed in several steps: first, a feedrate profile with commanded velocity given by F_{high} is found; second, a feedrate profile that realizes the override command is computed using the information from both the currently used profile and the profile limited by F_{high}. The process is described in detail in the rest of this section.

Let $s \in [0, L]$ be the displacement where feedrate override command is received and let $n \in \{1, \dots, N-1\}$ such that $s_n \leq s < s_{n+1}$, where $s_i \in U$ are knots of the feedrate curve (4). Denote by d the order of the feedrate curve (in all discussed cases $d = 4$ is assumed). To construct the override profile, it is first necessary to establish the range of control points of the feedrate curve which need to be fixed in order to guarantee that the override profile coincides with the current profile on some neighborhood of s. This range can be identified as $\{a_{I_f}, \dots, a_{I_l}\}$, where

$$\begin{aligned} I_f &= \max\{1, n - (d-1)\}, \\ I_l &= \min\{N, n + (d-1)\}. \end{aligned} \tag{6}$$

The range of control points given by (6) defines the maximal range of basis functions of the feedrate profile whose supports intersect the interval $[s_n, s_{n+1})$ and which therefore influence the values of the feedrate profile on this neighborhood of s. In the case where $I_l \geq (N-1)$, the computation of the override profile is skipped, since the override command was received at the very end of the tool path curve. If $I_l < (N-1)$, it is further necessary to check whether the rest of the feedrate profile reaches current commanded feedrate F_{cur} and whether the override command was not issued during, or immediately before, the final deceleration phase of the current feedrate profile. To check this, consider the point g_{J_s} where

$$J_s = \min\{k : g_k \geq s_{I_l}\}. \tag{7}$$

The point g_{J_s} is the first evaluation point after the point of override for which the corresponding value of the feedrate profile is unaffected by the choice of $\{a_{I_f}, \dots, a_{I_l}\}$. If the feedrate at the end of the curve is fixed, then if the rest of the feedrate profile does not reach the current commanded feedrate, i.e.

$$\max\{q_{cur}(g_j),\ j \in \{J_s, \dots, K\}\} < F_{cur}^2,$$

the computation of the override profile can be skipped. Similarly, if the override command has been issued during (or immediately before) the final deceleration phase, i.e.

$$q_{cur}(g_{J_s}) > q_{cur}(g_{J_s+1}) > \cdots > q_{cur}(g_K), \tag{8}$$

the computation of the override profile is skipped. This follows from the time-optimality of the current feedrate profile (the deceleration starts as late as possible considering the acceleration and jerk limits). Obviously, the previous two arguments do not apply if the end feedrate is allowed to rise above F_{cur}.

As a second step, the feedrate profile with maximal feedrate given by F_{high} is constructed. The override profile (denoted by the subscript *ovr*) is then constructed in two steps. Firstly, the pseudojerk $\hat{q}_{ovr}$ of the override profile is constructed by solving the LP optimization problem:

$$\text{maximize } \boldsymbol{c}^T \widehat{\boldsymbol{a}}_{ovr} \text{ subject to: } \begin{matrix} \widehat{\boldsymbol{M}}\widehat{\boldsymbol{a}}_{ovr} \leq \widehat{\boldsymbol{b}}_{ovr} \\ \widehat{\boldsymbol{lb}}_{ovr} \leq \widehat{\boldsymbol{a}}_{ovr} \leq \widehat{\boldsymbol{ub}}_{ovr}, \end{matrix}$$

(9)

where

$$\widehat{\boldsymbol{lb}}_i = \begin{cases} F_{start}^2, & i = 1, \\ 0, & i \in \{2, ..., I_f - 1\}, \\ (\boldsymbol{a}_{cur})_i, & i \in \{I_f, ..., I_l\}, \\ 0, & i \in \{I_l + 1, ..., N - 1\}, \\ F_{end}^2, & i = N \end{cases}$$

and

$$\widehat{\boldsymbol{ub}}_i = \begin{cases} F_{start}^2, & i = 1, \\ (\boldsymbol{a}_{cur})_i, & i \in \{2, ..., I_l\}, \\ \alpha_i, & i \in \{I_l + 1, N - 1\}, \\ F_{end}^2, & i = N, \end{cases}$$

such that

$$\alpha_i = \min\left\{F_{high}^2, \max\left\{(\boldsymbol{a}_{cur})_i, (\boldsymbol{a}_{high})_i\right\}\right\}.$$

The feedrate limitation of the right-hand side $\widehat{\boldsymbol{b}}_{ovr}$ is given by the feedrate values of the current feedrate profile up to g_{J_s} and by F_{high} from g_{J_s} onward.

The override profile q_{ovr} is then found by solving the LP problem:

$$\text{maximize } \boldsymbol{c}^T \boldsymbol{a}_{ovr} \text{ subject to: } \begin{matrix} \boldsymbol{M}\boldsymbol{a}_{ovr} \leq \boldsymbol{b}_{ovr} \\ \widehat{\boldsymbol{lb}}_{ovr} \leq \boldsymbol{a}_{ovr} \leq \widehat{\boldsymbol{ub}}_{ovr} \end{matrix}$$

where the velocity-acceleration part of $\boldsymbol{b}_{ovr}$ is equal to $\widehat{\boldsymbol{b}}_{ovr}$, while the jerk part is constructed using the values of $\widehat{q}_{cur}$ up to g_{J_s} and the values of $\widehat{q}_{ovr}$ from g_{J_s} onward.

For an example of override acceleration, see Figure 11.

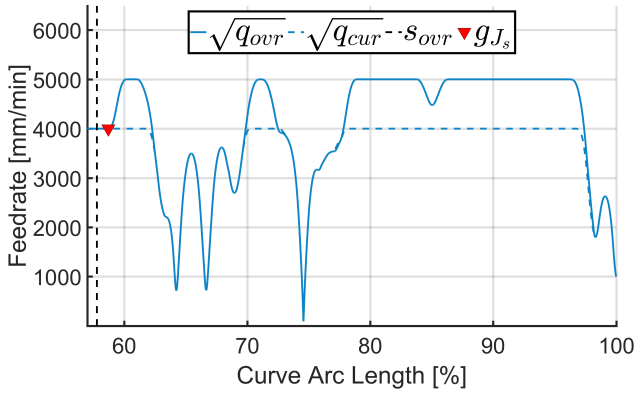

Fig. 11 Example of override acceleration - Butterfly curve, $F_{cur} = 4000$ mm/s, $F_{high} = 5000$ mm/s, $\alpha_{ovr} = 125\%$

4.3 Override deceleration

The process of override transition to a new feedrate limit $F_{low} = \alpha_{ovr} \cdot F_{cur} < F_{cur}$ requires a more involved approach than the case of override transition to a higher feedrate limit. The main idea is the following: first, as in Section 4.2, the first evaluation point g_{J_s} at which the override profile is allowed to deviate from the original profile is found. Then, for each following evaluation point g_{next}, the arc length necessary to transition from the current profile's feedrate at g_{J_s} to the feedrate value of the F_{low}-commanded feedrate profile at g_{next} is estimated. The first evaluation point for which this arc length estimate is not higher than the actual arc length between g_{J_s} and g_{next} is then used in the LP formulation of the override profile to define the range of control points to be bounded from above by the respective values of the F_{low}-commanded profile's control points. The rest of this section explains the algorithm in detail.

As in Section 4.2, the first order of business is to define the range of control points of the override profile which need to be fixed (6) and the index J_s of the first evaluation point at which the current profile and the override profile are allowed to deviate (7).

Secondly, as in Chapter 4.2, if the override command was issued during or immediately before the final deceleration phase (8), the override profile's construction should be skipped. This follows from the time optimality of the current feedrate profile (the deceleration is as fast as possible considering the acceleration and jerk limits).

If the override command was issued before the final deceleration phase, a feedrate profile q_{low} with commanded feedrate given by F_{low} is constructed. The final velocity of q_{low} is defined as

$$F_{end_{low}} = \min\left\{F_{end_{cur}}, F_{low}\right\}.$$

Next, the feedrate and tangential acceleration values of q_{cur} and q_{low} are evaluated at the point g_{J_s}. These values are denoted as $[F_{cur}(J_s), a_{cur}(J_s)]$ and $[F_{low}(J_s), a_{low}(J_s)]$, respectively. The exact form of the override algorithm then depends on whether
$F_{cur}(J_s) \leq F_{low}(J_s)$ (Section 4.3.1), or
$F_{cur}(J_s) > F_{low}(J_s)$ (Section 4.3.2).

4.3.1 Case I: $F_{cur}(J_s) \leq F_{low}(J_s)$

In this case, it is not necessary to compute any kind of estimate of the arc length necessary to transition from the current profile's feedrate to the feedrate of q_{low}. Instead, it is sufficient to take the appropriate range of

control points of the q_{low} profile as the upper boundary in construction of the override profile. To this end, define an index I_t as

$$I_t = I_l + d.$$

If $I_t \geq (N-1)$, the rest of the trajectory is not long enough to decelerate below the commanded feedrate. Otherwise, $\widehat{q}_{ovr}$ is constructed by solving the LP problem (9), where

$$\widehat{\boldsymbol{lb}}_i = \begin{cases} F^2_{start}, & i = 1, \\ 0, & i \in \{2, ..., I_f - 1\}, \\ (\boldsymbol{a}_{cur})_i, & i \in \{I_f, ..., I_l\}, \\ 0, & i \in \{I_l + 1, N - 1\}, \\ F^2_{end_{low}}, & i = N \end{cases}$$

and

$$\widehat{\boldsymbol{ub}}_i = \begin{cases} F^2_{start}, & i = 1, \\ (\boldsymbol{a}_{cur})_i, & i \in \{2, ..., I_l\}, \\ F^2_{cur}, & i \in \{I_l + 1, I_t - 1\}, \\ (\boldsymbol{a}_{low})_i, & i \in \{I_t, \ldots, N - 1\}, \\ F^2_{end_{low}}, & i = N. \end{cases}$$

The feedrate limitation of the right-hand side $\widehat{\boldsymbol{b}}_{ovr}$ is given by the feedrate values of the current feedrate profile up to g_{J_s} and by F_{new} from g_{J_s} onward. The override profile q_{ovr} is then found by solving the LP problem:

$$\text{maximize } \boldsymbol{c}^T \boldsymbol{a}_{ovr} \text{ subject to: } \begin{matrix} \boldsymbol{M}\boldsymbol{a}_{ovr} \leq \boldsymbol{b}_{ovr} \\ \widehat{\boldsymbol{lb}}_{ovr} \leq \boldsymbol{a}_{ovr} \leq \widehat{\boldsymbol{ub}}_{ovr} \end{matrix}$$

where the velocity-acceleration part of $\boldsymbol{b}_{ovr}$ is equal to $\widehat{\boldsymbol{b}}_{ovr}$, while the jerk part is constructed using the values of $\widehat{q}_{ovr}$.

For an example of case I override, see Figure 12.

4.3.2 Case II: $F_{cur}(J_s) > F_{low}(J_s)$

To formulate the upper and lower boundaries of the LP solution of the override profile, an upper estimate of the arc length necessary to realize the transition from $F_{cur}(J_s)$ to $F_{low}(k)$ (for some as yet unknown index k) is required.

First, define an index J_l as

$$J_l = \min\left(\{k > J_s : q_{cur}(g_k) \leq q_{low}(g_k)\} \cup \{K\}\right)$$

(i.e. the index of the first evaluation point after g_{J_s} for which q_{cur} is bounded from above by q_{low}. If no such point exists, the index is set as the index of the last evaluation point).

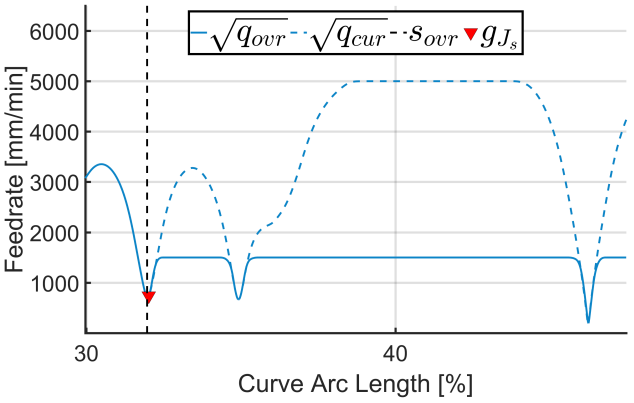

Fig. 12 Example of override deceleration, case I - Butterfly curve, $F_{cur} = 5000$ mm/s, $F_{low} = 1500$ mm/s, $\alpha_{ovr} = 30\%$

Algorithm 1: Transition arc length estimate

Input: $\{F_{cur}(k)\}_{J_s}^{K}$, $\{F_{low}(k)\}_{J_s}^{K}$, $\{a_{cur}(k)\}_{J_s}^{K}$, $\{a_{low}(k)\}_{J_s}^{K}$, $\{g_k\}_{J_s}^{K}$
Output: J_k `// index of first possible end point of feedrate transition`
Initialization: $J_k = J_l$;
for ($k = J_s + 1$; $k \leq J_l$; $k = k + 1$)
  if $F_{cur}(J_s) > F_{low}(k)$ **then**
    $s =$ `ArcLen`$(F_{cur}(J_s),\ F_{low}(k),\ a_{cur}(J_s), a_{max}, j_{max})$
  else
    $s =$ `ArcLen`$(F_{low}(k),\ F_{cur}(J_s),\ a_{low}(k), a_{max}, j_{max})$
  if $s < (g_k - g_{J_s})$ **then**
    $J_k = k$;
    return J_k;
return J_k;

Next, to estimate the minimal arc length necessary for a transition from q_{cur} to q_{low}, the following algorithm is used: (for the definition of the ArcLen function, see Appendix 8, Algorithm 3 and Algorithm 4). Algorithm 1 searches for the closest evaluation point after which the feedrate transition can already be finalized. The search stops at the index J_l. As a next step consider the index J_e defined as

$$J_e = \min\left\{J_l, J_k\right\}.$$

If $J_e = K$, the remaining arc length of the path curve is not sufficient to realize the feedrate transition and the override command should be skipped (or postponed to the start of the following path segment). Otherwise, a control point index I_t is defined as

$$I_t = \min\{k : s_k \geq g(J_e)\} + (d - 1).$$

Index I_t denotes the first control point for which the support of the corresponding basis function lies past

the evaluation point $g(J_e)$. If I_t is undefined or $I_t \geq K$, the remaining arc length of the path curve is not sufficient to realize the feedrate transition and the override command should be skipped (or postponed to the start of the following path segment). Otherwise, the $\widehat{q}_{ovr}$ profile is constructed using the following LP-optimization problem:

$$\text{maximize } \boldsymbol{c}^T\widehat{\boldsymbol{a}}_{ovr} \text{ subject to: } \begin{matrix}\widehat{\boldsymbol{M}}\widehat{\boldsymbol{a}}_{ovr} \leq \widehat{\boldsymbol{b}}_{cur} \\ \widehat{\boldsymbol{lb}}_{ovr} \leq \widehat{\boldsymbol{a}}_{ovr} \leq \widehat{\boldsymbol{ub}}_{ovr},\end{matrix}$$

where the lower and upper boundaries of $\widehat{\boldsymbol{a}}_{ovr}$ are defined as

$$\widehat{\boldsymbol{lb}}_i = \begin{cases} F^2_{start}, & i = 1, \\ 0, & i \in \{2, ..., I_f - 1\}, \\ (\boldsymbol{a}_{cur})_i, & i \in \{I_f, ..., I_l\}, \\ 0, & i \in \{I_l + 1, N - 1\}, \\ F^2_{end_{low}}, & i = N \end{cases}$$

and

$$\widehat{\boldsymbol{ub}}_i = \begin{cases} F^2_{start}, & i = 1, \\ (\boldsymbol{a}_{cur})_i, & i \in \{2, ..., I_l\}, \\ F^2_{cur}, & i \in \{I_l + 1, I_t - 1\}, \\ (\boldsymbol{a}_{low})_i, & i \in \{I_t, \ldots, N - 1\}, \\ F^2_{end_{low}}, & i = N, \end{cases}$$

respectively. The feedrate solution q_{ovr} is then found by solving the LP problem:

$$\text{maximize } \boldsymbol{c}^T\boldsymbol{a}_{ovr} \text{ subject to: } \begin{matrix}\boldsymbol{M}\boldsymbol{a}_{ovr} \leq \boldsymbol{b}_{ovr} \\ \widehat{\boldsymbol{lb}}_{ovr} \leq \boldsymbol{a}_{ovr} \leq \widehat{\boldsymbol{ub}}_{ovr},\end{matrix}$$

where $\boldsymbol{b}_{ovr}$ is constructed from $\widehat{b}_{ovr}$ and $\widehat{q}_{ovr}$ in a standard fashion.

For an example of case II override, see Figure 13.

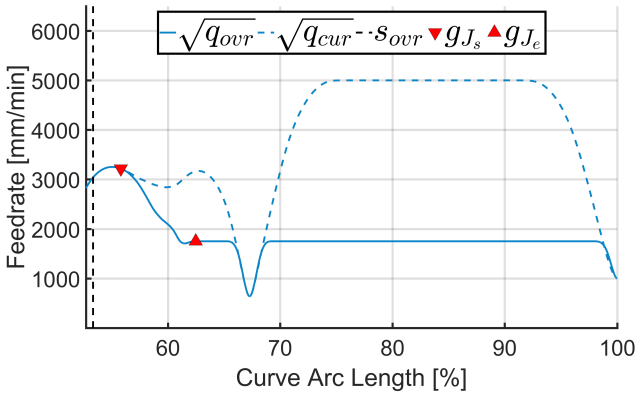

Fig. 13 Example of override deceleration, case II - Trident curve, $F_{cur} = 5000$ mm/s, $F_{low} = 1750$ mm/s, $\alpha_{ovr} = 35\%$

4.3.3 Case II: Convergence failure

The LP formulations associated with the construction of $\widehat{q}_{ovr}$ or q_{ovr} described in the previous section can occasionally fail to converge to a solution. This is due to the fact that the function ArcLen assumes a so-called "'bang-bang"' transition feedrate profile, which exhibits maximal absolute values of either acceleration or jerk. When applied to general spline toolpaths this behavior is not always possible, due to limitations posed by local geometry. This convergence failure thus sometimes occurs when the override command was issued during or immediately before a dip in feedrate caused by a significant change in curvature. In such cases the override profile can still be obtained by setting a new point of override as the nearest local extreme of the feedrate curve and repeating the override computation. Specifically, supposing that either of the LP problems

Algorithm 2: Nearest local extreme point of q_{ovr}

Input: J_s, $\{A_{cur}(k)\}_{J_s}^{K}$
Output: J_{ext} `// index of first point of local feedrate extreme after` g_{J_s}
if $(A_{J_s} < 0)$ **then**
  $J_{ext} = \min_{k \in \{J_s+1,\ldots,K\}} \{k : A_{cur}(k) \geq 0\}$
else
  $J_{ext} = \min_{k \in \{J_s+1,\ldots,K\}} \{k : A_{cur}(k) \leq 0\}$
return J_{ext}

failed to converge, the nearest local extreme point of q_{cur} defined as an evaluation point $g_{J_{ext}}$ can be found via Algorithm 2. The index J_{ext} is well defined since the override formulation is skipped whenever the override command is issued during the final deceleration phase.

Note that in the case of repeated override calculation, the feedrate profile q_{low} and its values at evaluation points have already been obtained and do not need to be recalculated.

5 Conclusions

This paper comprises two original results related to feedrate optimization via linear programming techniques: the effects of curve sampling methods on time optimality of the resulting feedrate profile (Section 3) and a technique by which a feedrate override profile can be implemented (Section 4).

Previous works have not dedicated much attention to the effects that toolpath sampling can have on the effectiveness of feedrate planning. This paper demonstrates that this effect can be substantial (when com-

bined with linear programming feedrate planning) and suggests a suitable sampling method. This method can be integrated into the feedrate planning process with low computational overhead, thus making it an interesting choice for practical applications. Note that while the adaptive method requires the approximation of the relation between the curve's natural parameter and its arc length as a prerequisite, this requirement is not really restrictive. Firstly, the approximation itself is required for the formulation of the feedrate LP optimization problem. Secondly, both the approximation and the sampling itself can (should) be performed in the preprocessing stage of feedrate planning and therefore do not add to the computational complexity of the online stage.

In order to successfully apply linear programming techniques to real world feedrate planning, a method capable of recalculating feedrate profile in reaction to a feedrate override command is essential. This topic, however, has not been dealt with in the past. Thus, a possible implementation of such a method is presented in this paper. While this method does require the computation of two additional feedrate profiles (one bounded by the newly issued commanded feedrate and second that realizes the desired feedrate transition), the majority of prerequisites for the computation of these profiles are shared with the original profile. Specifically: the knot vector associated with the feedrate profile, its evaluation points and the constant matrices of both the pseudojerk and feedrate curve optimization problems can be stored in memory the first time they are computed and reused in subsequent optimizations. Main part of the computational complexity is therefore due to the use of linear programming solvers as all the additional computations required are either index manipulations, or simple analytic formulas. The method is therefore suitable for application purposes due to its low memory requirements and simplicity of auxiliary calculations, as long as a suitably fast linear programming solver is available.

Acknowledgements The authors would like to acknowledge funding support from the Czech Ministry of Education, Youth and Sports under the project CZ.02.1.01/0.0/0.0/ 16_026/0008404 "'Machine Tools and Precision Engineering"' financed by the OP RDE (ERDF). The project is also co-financed by the European Union.

6 Declarations

6.1 Funding

The authors would like to acknowledge funding support from the Czech Ministry of Education, Youth and Sports under the project CZ.02.1.01/0.0/0.0/16_026 /0008404 "'Machine Tools and Precision Engineering"' financed by the OP RDE (ERDF). The project is also co-financed by the European Union.

6.2 Conflicts of interest

Financial interests: The authors have no conflicts of interest to declare that are relevant to the content of this article. **Non-financial interests:** None.

6.3 Availability of data and material

Not applicable.

6.4 Code availability

Code used for interpolation techniques referred to in this paper is available upon request from the corresponding author.

6.5 Author's contributions

Conceptualization:
[Petr Petráček, Bořivoj Vlk, Jiří Švéda];
Formal analysis: [Petr Petráček, Bořivoj Vlk];
Funding acquisition: [Jiří Švéda];
Investigation: [Petr Petráček, Bořivoj Vlk];
Methodology: [Petr Petráček];
Project administration: [Jiří Švéda];
Software: [Petr Petráček, Bořivoj Vlk];
Supervision: [Jiří Švéda];
Visualization:
[Petr Petráček, Bořivoj Vlk, Jiří Švéda];
Writing-original draft:
[Petr Petráček, Bořivoj Vlk];
Writing - review & editing:
[Petr Petráček, Bořivoj Vlk, Jiří Švéda]

6.6 Ethics approval

Not applicable.

6.7 Consent to participate

Not applicable.

6.8 Consent for publication

Not applicable.

References

1. Altintas Y, Erkorkmaz K (2003) Feedrate optimization for spline interpolation in high speed machine tools. CIRP Annals 52(1):297–302, DOI https://doi.org/10.1016/S0007-8506(07)60588-5
2. Besset P, Béarée R (2017) FIR filter-based online jerk-constrained trajectory generation. Control Engineering Practice 66:169–180, DOI https://doi.org/10.1016/j.conengprac.2017.06.015
3. Beudaert X, Pechard PY, Tournier C (2011) 5-axis tool path smoothing based on drive constraints. International Journal of Machine Tools & Manufacture 51(12):958–965, DOI https://doi.org/10.1016/j.ijmachtools.2011.08.014
4. Beudaert X, Lavernhe S, Tournier C (2012) Feedrate interpolation with axis jerk constraints on 5-axis NURBS and G1 tool path. International Journal of Machine Tools & Manufacture 57:73–82, DOI https://doi.org/10.1016/j.ijmachtools.2012.02.005
5. Beudaert X, Lavernhe S, Tournier C (2014) Feedrate optimization in 5-axis machining based on direct trajectory interpolation on the surface using an open CNC. In: 11th International Conference on High Speed Machining, Prague, Czech Republic, pp paper n. 14042, 6 pages, URL `https://hal.archives-ouvertes.fr/hal-01064136`
6. Biagiotti L, Melchiorri C (2012) FIR filters for online trajectory planning with time- and frequency-domain specifications. Control Engineering Practice 20(12):1385–1399, DOI https://doi.org/10.1016/j.conengprac.2012.08.005
7. Chen J, Ren F, Sun Y (2016) Contouring accuracy improvement using an adaptive feedrate planning method for CNC machine tools. Procedia CIRP 56:299 – 305, DOI https://doi.org/10.1016/j.procir.2016.10.012, "The 9th International Conference on Digital Enterprise Technology – Intelligent Manufacturing in the Knowledge Economy Era"
8. COIN-OR Foundation Inc (2000–2021) COIN-OR linear programming project. https://github.com/coin-or/Clp
9. Dong W, Ding Y, Huang J, Zhu X, Ding H (2017) An efficient approach of time-optimal trajectory generation for the fully autonomous navigation of the quadrotor. Journal of Dynamic Systems, Measurement, and Control 139(6), DOI https://doi.org/10.1115/1.4035453
10. Erkorkmaz K, Altintas Y (2001) High speed CNC system design. Part I: jerk limited trajectory generation and quintic spline interpolation. International Journal of Machine Tools & Manufacture 41(9):1323–1345, DOI https://doi.org/10.1016/S0890-6955(01)00002-5
11. Erkorkmaz K, Altintas Y (2005) Quintic spline interpolation with minimal feed fluctuation. Journal of Manufacturing Science and Engineering-transactions of The Asme 127(2):339–349, DOI https://doi.org/10.1115/1.1830493
12. Erkorkmaz K, Heng M (2008) A heuristic feedrate optimization strategy for NURBS toolpaths. CIRP Annals-manufacturing Technology 57(1):407–410, DOI https://doi.org/10.1016/j.cirp.2008.03.039
13. Erkorkmaz K, Chen QGC, Zhao MY, Beudaert X, Gao XS (2017) Linear programming and windowing based feedrate optimization for spline toolpaths. CIRP Annals-manufacturing Technology 66(1):393–396, DOI https://doi.org/10.1016/j.cirp.2017.04.058
14. Fan W, Gao XS, Lee CH, Zhang K, Zhang Q (2013) Time-optimal interpolation for five-axis CNC machining along parametric tool path based on linear programming. The International Journal of Advanced Manufacturing Technology 69(5):1373–1388, DOI https://doi.org/10.1007/s00170-013-5083-x
15. Farouki RT (2008) Pythagorean–hodograph Curves. Springer-Verlag, Berlin, Heidelberg, DOI https://doi.org/10.1007/978-3-540-73398-0_17
16. Farouki RT, Tsai YF (2001) Exact Taylor series coefficients for variable-feedrate CNC curve interpolators. Computer-Aided Design 33(2):155–165, DOI https://doi.org/10.1016/S0010-4485(00)00085-3
17. Heng M, Erkorkmaz K (2010) Design of a NURBS interpolator with minimal feed fluctuation and continuous feed modulation capability. International Journal of Machine Tools & Manufacture 50(3):281–293, DOI https://doi.org/10.1016/j.ijmachtools.2009.11.005
18. Khamayseh A, Kuprat A (2002) Hybrid curve point distribution algorithms. SIAM Journal on Scientific Computing 23(5):1464–1484, DOI https://doi.org/10.1137/S1064827500367592
19. Kim H, Okwudire CE (2020) Simultaneous servo error pre-compensation and feedrate optimization with tolerance constraints using linear programming. The International Journal of Advanced Manufacturing Technology 109:1–13, DOI https://doi.org/10.1007/s00170-020-05651-w
20. McNeel, R & associates (1997–2021) openNURBS toolkit (version 6.0).

https://www.rhino3d.com/opennurbs
21. Piegl LA, Tiller W (1997) The NURBS Book, 2nd edn. Springer-Verlag, Berlin, Heidelberg
22. Sun Y, Chen M, Jia J, Lee YS, Guo D (2019) Jerk-limited feedrate scheduling and optimization for five-axis machining using new piecewise linear programming approach. Science China Technological Sciences 62:1067–1081, DOI https://doi.org/10.1007/s11431-018-9404-9
23. Tajima S, Sencer B (2019) Accurate real-time interpolation of 5-axis tool-paths with local corner smoothing. International Journal of Machine Tools & Manufacture 142:1–15, DOI https://doi.org/10.1016/j.ijmachtools.2019.04.005
24. Tajima S, Sencer B, Shamoto E (2018) Accurate interpolation of machining tool-paths based on FIR filtering. Precision Engineering-journal of The International Societies for Precision Engineering and Nanotechnology 52:332–344, DOI https://doi.org/10.1016/j.precisioneng.2018.01.016
25. Tulsyan S, Altintas Y (2015) Local toolpath smoothing for five-axis machine tools. International Journal of Machine Tools & Manufacture 96:15–26, DOI https://doi.org/10.1016/J.IJMACHTOOLS.2015.04.014
26. Wang L, Cao J (2012) A look-ahead and adaptive speed control algorithm for high-speed CNC equipment. The International Journal of Advanced Manufacturing Technology 63(5):705–717, DOI https://doi.org/10.1007/s00170-012-3924-7
27. Xin Z, Zhao H, Yang J, Ding H (2015) An adaptive feedrate scheduling method with multi-constraints for five-axis machine tools. In: Intelligent Robotics and Applications, pp 553–564, DOI https://doi.org/10.1007/978-3-319-22876-1_48

7 Appendix A

7.1 Pentacle curve

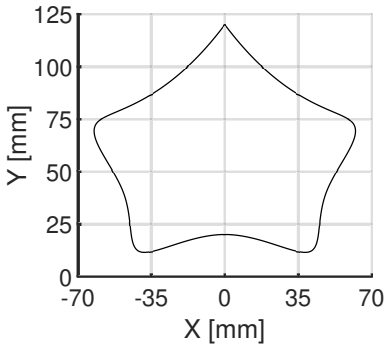

Fig. 14 Pentacle test curve

Table 2 Parameters of the Pentacle test curve

Parameters	Values
Control points	(0,120,0); (-30,80,0); (-80,80,0); (-40,40,0); (-50,0,0); (0,30,0); (50,0,0); (40,40,0); (80,80,0); (30,80,0); (0,120,0);
Knot vector	[0, 0, 0, 0, 0.125, 0.25, 0.375, 0.5, 0.625, 0.75, 0.875, 1, 1, 1, 1]
Weight vector	[1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1]
Order	4

7.2 Trident curve

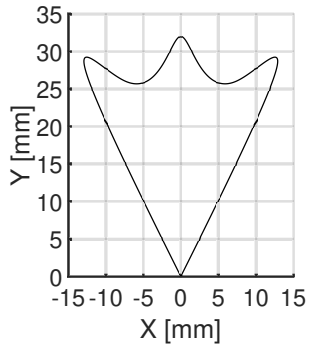

Fig. 15 Trident test curve

Table 3 Parameters of the Trident test curve

Parameters	Values
Control points	(0,0,0); (20,40,0); (4,16,0); (0,40,0); (-4,16,0); (-20,40,0); (0,0,0);
Knot vector	[0,0,0,0,0.25,0.5,0.75,1,1,1,1]
Weight vector	[1, 1, 1, 1, 1, 1, 1]
Order	4

7.3 Phobos curve

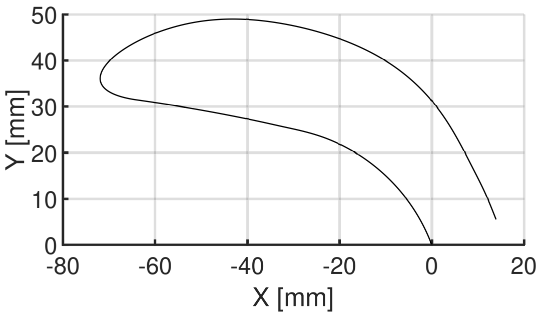

Fig. 16 Phobos test curve

Table 4 Parameters of the Phobos test curve

Parameters	Values
Control points	(-0.45492,6.2779,0); (-0.49971,6.3975,0); (-0.68117,6.8746,0); (-1.1578,8.0587,0); (-1.9497,9.8028,0); (-3.1389,12.063,0); (-4.6134,14.456,0); (-6.4198,16.933,0); (-8.5788,19.458,0); (-10.765,21.609,0); (-13.107,23.585,0); (-15.783,25.554,0); (-19.296,27.648,0); (-23.963,29.76,0); (-28.406,30.986,0); (-32.899,31.942,0); (-36.881,32.85,0); (-41.877,33.899,0); (-46.387,34.777,0); (-50.908,35.597,0); (-54.933,36.285,0); (-58.463,36.843,0); (-61.491,37.3,0); (-63.513,37.589,0); (-65.535,37.878,0); (-67.16,38.216,0); (-68.634,38.645,0); (-69.929,39.192,0); (-70.923,39.776,0); (-71.887,40.635,0); (-72.523,41.952,0); (-72.264,43.567,0); (-71.383,45.129,0); (-70.166,46.458,0); (-68.418,47.949,0); (-66.298,49.384,0); (-62.707,51.334,0); (-57.94,53.224,0); (-51.982,54.7,0); (-45.867,55.358,0); (-39.731,55.245,0); (-32.626,54.381,0); (-24.639,52.567,0); (-15.976,49.444,0); (-7.9015,44.909,0); (-1.79,39.401,0); (3.8241,32.068,0); (7.4947,24.751,0); (11.252,17.481,0); (12.751,13.68,0); (13.5,11.779,0);
Knot vector	[0,0,0,0,0.0019531,0.0078125,0.019531, 0.03125,0.046875,0.0625,0.078125, 0.097656,0.10938,0.125,0.14844, 0.17188,0.20313,0.21875,0.24219, 0.26563,0.29688,0.3125,0.33594, 0.35938,0.36719,0.38281,0.39063, 0.39844,0.4082,0.41406,0.41992, 0.42578,0.43359,0.44141,0.44922, 0.46094,0.46875,0.48438,0.5,0.53125, 0.5625,0.59375,0.625,0.65625,0.70313, 0.75,0.79688,0.84375,0.875,0.9375, 0.96875,1,1,1,1]
Weight vector	[1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1, 1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1, 1,1,1,1,1,1,1,1,1,1]
Order	4

7.4 Butterfly curve

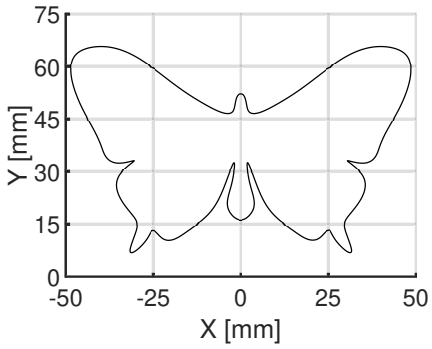

Fig. 17 Butterfly test curve

Table 5 Parameters of the Butterfly test curve

Parameters	Values
Control points	(0,52.139,0); (1.014,52.139,0); (1.589,49.615,0); (2.287,44.971,0); (15.082,51.358,0); (23.293,58.573,0); (36.033,67.081,0); (51.48,63.801,0); (45.907,47.326,0); (40.074,39.913,0); (37.876,30.485,0); (57.894,67.514,0); (37.489,28.509,0); (34.951,20.393,0); (143.63,77.23,0); (99.384,14.49,0); (26.452,9.267,0); (27.875,15.989,0); (21.581,8.522,0); (15.69,12.55,0); (9.678,16.865,0); (5.5,22.122,0); (1.187,36.359,0); (2.432,24.995,0); (5.272,19.828,0); (0,14.94,0); (-5.273,19.828,0); (-2.433,24.994,0); (-1.188,36.359,0); (-5.501,22.122,0); (-9.679,16.865,0); (-15.691,12.551,0); (-21.582,8.521,0); (-25.341,14.535,0); (-26.453,9.267,0); (-99.387,14.49,0); (-143.63,77.235,0); (-34.954,20.391,0); (-37.396,28.512,0); (-57.912,67.5,0); (-37.891,30.496,0); (-40.294,39.803,0); (-45.825,47.408,0); (-51.493,63.794,0); (-36.028,67.084,0); (-23.296,58.572,0); (-15.082,51.358,0); (-2.289,44.971,0); (-1.589,49.614,0); (-1.015,52.139,0); (-0.001,52.139,0);
Knot vector	[0,0,0,0,0.0083,0.015,0.0361,0.0855, 0.1293,0.1509,0.1931,0.2273,0.2435, 0.2561,0.2692,0.2889,0.317,0.3316, 0.3482,0.3553,0.3649,0.3837,0.4005, 0.4269,0.451,0.466,0.4891,0.5, 0.5109,0.534,0.5489,0.5731,0.5994, 0.6163,0.6351,0.6447,0.6518,0.6683, 0.683,0.7111,0.7307,0.7439,0.7565, 0.7729,0.8069,0.8491,0.8707,0.9145, 0.9639,0.985,0.9917,1,1,1,1]
Weight vector	[1,1,1,1,1,1,1,1,1,1,1,2,1,1,5,3,1,1.1,1,1, 1,1,1,1,1,1,1,1,1,1,1,1,1,1,3,5,1,1,2,1, 1,1,1,1,1,1,1,1,1,1]
Order	4

8 APPENDIX B

This section contains the definition of the ArcLen function used in Section 4.3.2, Algorithm 1. The ArcLen function calculates an estimate of the the arc length necessary to decelerate from starting feedrate F_s to end feedrate F_e with zero acceleration and limits on tangential acceleration and jerk given by a_{max} and j_{max}, respectively. The arc length estimate is obtained via a so-called "'bang-bang"' feedrate profile, i.e. a profile that maximizes the absolute value of either jerk or acceleration at every point. In case the maximization of acceleration would lead to reaching the end feedrate F_e with nonzero acceleration, a new maximal deceleration value a_{mid} is calculated, so that the end feedrate can be reached with exactly zero acceleration with a_{mid} replacing a_{max} as a new acceleration limit (see code blocks starting at lines A and B in Algorithms 3 and 4 below).

Algorithm 3: Feedrate transition profile: part 1

Input:
F_s // Feedrate at start $[mm/s]$
F_e // Feedrate at the end $[mm/s]$
a_s // Tangential acceleration at start $[mm/s^2]$
a_{max} // Tangential acceleration limit $[mm/s^2]$
j_{max} // Tangential jerk limit $[mm/s^3]$
Output:
s // Arclength neccessary for transition from F_s to F_e $[mm]$
Function ArcLen(F_s, F_e, a_s, a_{max}, j_{max}):
  if $a_s \geq 0$ **then**
    // Phase 0: Decelerate from a_s to zero acceleration
    $T_0 = \frac{a_s}{j_{max}}$
    $F0 = F_s + T_0 a_s - \frac{1}{2} T_0^2 j_{max}$
    $s_0 = T_0 F_s + \frac{1}{2} T_0^2 \cdot a_s - \frac{T_0^3}{6} j_{max}$
    $F_s = F0$
    // Starting from F_s with zero acceleration, decelerate until $a = -a_{max}$
    $T_1 = \frac{a_{max}}{j_{max}}$
    $F1_e = F1 - \frac{1}{2} T_1^2 j_{max}$
    // Starting from F_e with zero acceleration, accelerate until $a = a_{max}$
    $F3_s = F_e + \frac{1}{2} T_1^2 j_{max}$
    if $F1_e > F3_s$ **then**
      // Phase with $a = -a_{max}$, from feedrate $F1_e$ to $F3_s$
      $T_2 = \frac{F1_e - F3_s}{a_{max}}$
      $s_2 = T_2 F1_e - \frac{1}{2} T_2^2 a_{max}$
      // Add lengths of all phases
      $s_3 = T_1 F_e + \frac{T_1^3}{6} j_{max}$
      $s_1 = T_1 F_s - \frac{T_1^3}{6} j_{max}$
      $s = s_0 + s_1 + s_2 + s_3$
    else
A   // Calculate a_{mid} such that F_e is reached with zero acceleration
      $a_{mid} = -\sqrt{(F_s - F_e) j_{max}}$
      $T_1 = -\frac{a_{mid}}{j_{max}}$
      $F1_e = F_s - \frac{1}{2} T_1^2 j_{max}$
      $s_1 = T_1 (F_s + F1_e) + \frac{1}{2} T_1^2 a_{mid}$
      $s = s_0 + s_1$
  else
    // Starting from F_s, decelerate until $a = -a_{max}$
    $T_1 = \frac{(a_{max} + a_s)}{j_{max}}$
    $F1_e = F_s + T_1 a_s - \frac{1}{2} T_1^2 j_{max}$
    // Starting from F_e with zero acceleration, accelerate until $a = a_{max}$
    $T_3 = \frac{a_{max}}{j_{max}}$
    $F3_s = F_e + \frac{1}{2} T_3^2 j_{max}$

Algorithm 4: Feedrate transition profile: part 2

if $F1_e \geq F3_s$ **then**
  // Phase with $a = -a_{max}$, from feedrate $F1_e$ to $F3_s$
  $T_2 = \frac{F1_e - F3_s}{a_{max}}$
  $s_2 = T_2 F3_s - \frac{1}{2} T_2^2 a_{max}$
  $s_1 = T_1 F_s + \frac{1}{2} T_1^2 a_s + \frac{T_1^3}{6} j_{max}$
  $s_3 = T_3 F3_s + \frac{T_3^3}{6} j_{max}$
  $s = s_1 + s_2 + s_3$
else
B // Calculate a_{mid} such that F_e is reached with zero acceleration
  $a_{mid} = \max\left\{-a_{max}, -\sqrt{\frac{1}{2} a_s^2 + (F_s - F_e) j_{max}}\right\}$
  $T_1 = \frac{a_s - a_{mid}}{j_{max}}$
  $F1_e = F_s + T_1 a_s - \frac{1}{2} T_1^2 j_{max}$
  $s1_e = T_1 F_s + \frac{1}{2} T_1^2 a_s - \frac{T_1^3}{6} j_{max}$
  $s = s1_e + T_1 F1_e + \frac{1}{2} T_1^2 a_{mid} + \frac{T_1^3}{6} j_{max}$
return s