Interpolation Algorithm for High-Speed Processing of Complex Curvilinear Trajectories

Kseniia Zimenko, Maxim Afanasev, Yuri Andreev,
Anastasiya Krylova, Sergey Shorokhov, Yuri Fedosov, Mikhail Kolesnikov
ITMO University, St. Petersburg, Russia
zksenia@yahoo.com, amax@niuitmo.ru, ysandreev@itmo.ru, {ananasn94, stratumxspb}@gmail.com,
yf01@yandex.ru, km@hexaxis.ru

***Abstract*—The rising popularity of complex product design leads to frequent use of curvilinear toolpaths during machining. Along with growing accuracy requirements for processing the complication of part geometry leads to the importance of developing efficient interpolation methods for processing along complex toolpaths. In this case, to obtain an online interpolation algorithm for complex trajectories the combination methods, such as the use of parametric curve representation for complex paths, approximation into equal linear segments during interpolation and application of curve on trajectory corners, is considered. The paper contains the description and analysis of the developed interpolation algorithm for complex trajectories. The proposed curve approximation into equal segments allows to achieve constant speed and minimal contour error. The comparison between the conventional and proposed segmentation methods have been obtained via simulation and analyzed in the present paper. The developed algorithm allows to significantly reduce the segment length and velocity deviations. The application of an acceleration/deceleration control with S-shape feedrate profile for curve interpolation is also described in the paper. The proposed method allows the calculation of an optimal parameter increment with low computational load, and thus can be applied to real-time machining.**

I. Introduction

The growing popularity and availability of personalized manufacturing, when the design of a product and its functions and features is made strictly according to customer requirements, can be noticed in the field of modern industrial production. More and more enterprises offer a complex and unique product design for their customers.

This tendency leads inevitably to the complication of devices geometry. So despite the fact that during processing a toolpath still mostly consists of linear segments and arcs, complex trajectories such as curves, spirals and involutes are used increasingly more often. Combined with the high required accuracy, the complication of trajectories leads to the problem of developing efficient algorithms for processing along complex toolpaths – curves.

In Computer Numeric Control (CNC) software the axial tool movement generation, based on data obtained from the control program code (G-codes), is performed by the interpolation module [1]. At this stage the toolpath is formed. The processing accuracy depends on how much the initial path obtained from G-codes coincides with the toolpath obtained during interpolation. Therefore, the aim of this project is to develop an efficient interpolation algorithm for complex trajectories that will produce minimal contour error and stable velocities, which thus will ensure high precision processing and high quality of a machined surface.

Most of CNC machines presented on the modern market are equipped with the necessary software, allowing it to process complex trajectories at high tool speeds with minimal machining error. However, these solutions are proprietary and are revealed neither for third-party developers nor for the public. In light of the above, one of the aims of the present project is to create an efficient open-source solution.

Most of the open-source CNC algorithms available imply curve segmentation into unequal path units which leads to inconsistent velocity and, hereby, to the deterioration of a part surface obtained [2]-[4]. Apart from the mentioned above most of the proposed methods are not applicable for high-speed machining due to limitations that implies a machining error which is in direct proportion of a tool speed.

The way to perform segmentation and calculate a curvilinear path so that obtained equal segments ensure the constant feedrate value is proposed in the present paper. Parametric representation of curves is used in the solution. The described interpolation method is combined with the acceleration/deceleration control that generates an S-shaped speed profile and thus provides smooth changes of velocity during processing.

Together with curves, arcs interpolation is also considered in the paper with application of the same approach. The developed interpolation method for trajectories at corners will also be described. This paper presents a description of the methodology, implementation and simulation results.

Among the criteria for the development is maximum simplicity and universality of calculations. Firstly, all proposed calculations are suitable for application in a real-time interpolator and do not have a heavy computational load. Secondly, as opposed to most of the interpolators that are based on a specific type of a curve, the present research works with parametric representation in general, so that the developed method could be applicable to any type of curvilinear trajectory.

The paper describes the algorithm application for machines that are stiff dynamic systems. The influence of dynamic constraints is compensated by the negligible mass of a tool (laser head, extruder) compared to a machine itself.

The paper is organized as follows. Section II is dedicated to the overview of related researches. In Section III the main problem concerning complex trajectory interpolation,

segmentation of a curve, is discussed and the proposed solution is introduced. The opportunity to use the developed interpolation algorithm for arc processing and corner smoothing technique are also considered in the paper. Particular properties that circular processing imposes on the proposed algorithm application, are described in Section IV. Section V is given for highlighting the specifications for curves generation at corners. Section VI includes the description of the acceleration/deceleration control developed and the example of its application for curve interpolation. The analysis of the algorithm performance, including possible drawbacks and directions for further improvement and research, is given in Section VII.

II. Related Work

Two main method groups used in interpolation are:

1) Reference-Pulse Interpolators.
2) Reference-Word Interpolators [5].

In the first group mentioned, a computer generates reference pulses as an external interrupt signal. The produced pulses are relegated directly to the machine drive. It can achieve high accuracy but as the velocity on each axis depends on external interrupt signal frequency, high speed machining cannot be obtained and a high performance Central Processing Unit (CPU) is required [5].

The Reference-Word Interpolation implies calculating the next reference point based on the equal sample time T also called interpolation time. The trajectory is divided into small segments that are processed in one T in accordance with the required feedrate. This method is more widely used in modern CNC with servo motors and does not impose limitations on a processing speed [6]. It will be considered in the present paper.

The main idea of interpolation lies in dividing a path into small units, individual for each axis that are processed during the sample time T.

While this method of processing does not cause machining errors along the linear path, its application for complex trajectories brings certain problems including the optimal segmentation of a curve and the contour error value. Main attention in this paper is dedicated to developing an effective interpolation algorithm for complex trajectories that solves this problems.

The step of generating a toolpath and velocity profile based on a given trajectory is a key stage of CNC performance. Therefore, the question of developing an effective motion planning algorithms in both CNC machining and robot manipulators has always received a lot of attention among researches [7]-[9].

Parametric representation of curves finds application in a large number of CNC software algorithms for curve trajectories [9]-[12]. However, the proposed solutions are mostly applicable to specific type of a curve. For example, Daoshan Du et al. [10] described the adaptive parametric curve interpolator which results in high accuracy, designed specifically for NURBS curves. Lijuan Chen et al. in [11] took into consideration the shape preserving interpolation with the use of cubic NURBS curves. Kaan Erkorkmaz and Yusuf Altintas [12] proposed a trajectory generation algorithm that produces continuous position profile and reduced jerk value during processing, based on a quintic spline curve. The present research, on the other hand, does not work with a specific type of curves but aims to provide universality in terms of applying different types of curves. The developed algorithm should be efficient for any curvilinear representation including Bezier, B-spline, NURBS etc.

Most of papers include solutions for parametric path generation during interpolation based on the uniform increment of the curve parameter. Burak Sencer et al. [13] propose the algorithm for corner smoothing with Bezier curve, however the calculating is performed by gradual addition of the curve parameter. Bingran Li et al. [14] described a corner smoothing technique based on the allowable contour error, the solution does not guarantee the equal size of an obtained curve. Other solutions require complex calculations of a curve parameter that are only suitable for offline interpolations. The proposed solution provides the curvilinear approximation into equal segments and the calculations are suitable for the real-time generation.

Lu L. et al. [15] proposes the algorithm of feedrate scheduling for parametric curves processing. However, the method implies its application in an offline mode. One of the aims of the present project is the development of a solution for real time performance. H. Li et al. [16] presents a method for acceleration/deceleration control that allows smooth changes of velocity and creates an S-shape feedrate profile. Although the proposed algorithm could be used in a real time interpolation procedure it is only applied for processing of linear trajectories.

To summarize, after analysis of related researches the task remains develop an interpolation algorithm for curvilinear trajectories that provides uniform path segments and thus stable velocity during processing that does not impose limitations on the type of curve used and is applicable for real-time CNC algorithms.

III. Curve segmentation

The most common way to represent a curve in a CNC is the parametric form, when it is described by the Eq. 1.

$$x = x(u); y = y(u) \tag{1}$$

where u is a parameter that, in most cases, changes in range between 0 and 1. Most CAD/CAM systems represent curves using these equations.

Coordinates of each axis are calculated directly using different equations, which facilitates the transition from two-dimensional to three-dimensional representations and higher. For simplicity, all examples shown in the paper are presented in two dimensional (XY) plane.

When working with complex trajectories, traditional interpolation algorithms produce a preliminary approximation of a curve by linear sections. Each of these sections than in turn is further divided into smaller segments. It leads to an increased contour error. In this case, to avoid inaccuracies, the initial segmentation of a curve should produce length units as close as possible to the initial trajectory. However, this process requires significant computational resources and memory. To solve the problem, in this paper the algorithms that

perform direct calculations of a parametric curve form with single approximation are considered. This way the number of iterations is reduced and processing accuracy is maintained.

The curve segmentation is carried out by successively increasing the parameter u value and calculating the next point to which a tool should arrive per sample time T. To construct a curve in this manner adequately, it is necessary determine the optimal u increment or Δu. The Δu value has to comply with three requirements. Firstly, parameter u must be calculated independently of the geometric parameters of the curve. Secondly, the representation of $x(u)$ and $y(u)$ must be explicit to allow direct calculation of $x(u)$ and $y(u)$. And the last rule by mention, but one of the most important in processing is that each next increment of u should produce an equal in length segment Δl. Since the tool passes one segment in equal sample time T, this rule should ensure that a constant feedrate $V = \Delta l \ / \ T$ is obtained.

Most of the traditional interpolation algorithms meet the first two requirements, but the third condition is usually ignored and presents the notable difficulty. This rule holds only in the case of linear paths and arcs.

In this paper two main strategies to calculating the u increment for curve interpolation are analyzed.

A. Segmentation with equal Δu

The first approach for curve calculation suggests dividing it by incrementing the u parameter gradually to define each next point $[x(u_i), y(u_i)]$. The tool then moves straight from the point $[x(u_{i-1}), y(u_{i-1})]$ to $[x(u_i), y(u_i)]$. And segment Δl is then calculated using the Eq. 2:

$$\sqrt{(x_i - x_{i-1})^2 + (y_i - y_{i-1})^2} \tag{2}$$

It can be concluded that the smaller the increments of parameter u, the bigger number of segments will be generated. This leads to less divergence between to the initial and generated curve. The described segmentation method is used in most interpolation algorithms. However, despite its obvious calculation simplicity, this method has two main drawbacks that limit its use in a CNC system.

The first problem is related to a tool speed during processing. Equal increments Δu do not guarantee that each next segment of a curve Δl will have a equal length. However, as according to the processing standards, a tool passes each segment in an equal period of time (sample time T), therefore it will result in different processing speed during each segment, which results in irregular feedrate and, and as a consequence, to deterioration in the quality of the obtained surface.

The second disadvantage is the problem of finding the optimal parameter increment Δu. There is no specific equation or rule that could determine the exact needed value for each curve. There are convincing arguments in favor of both maximizing, and minimizing increment Δu [17].

On the one hand, Δu should be value as little as possible for the following reasons. Firstly, as was stated above it will ensure more accurate approximation of the original curve. Secondly, it will minimize the segmentation effect, which results in poor surface quality.

However, on the other hand this leads to a large number of curve segments that require significant computation and memory capacity. Apart from that, since the CNC system perceives each segment as a separate processing unit, this leads to a decrease in the average tool speed and consequently to an increase in machining time [17]. In this regard, the number of segments of the curve should be minimized (increase of the Δu value). Between maximization and minimization it is necessary to find the average value, which differs for each next curve. The possible equation for Δu should consider the curvature of a path, the feedrate and the sample time value and should be calculated for each T.

B. Segmentation with equal Δl

The main idea of the second algorithm for curve segmentation is to obtain equal segments of the curve Δl, but not equal Δu. At each interpolation step, the optimal Δu is calculated so as to achieve Δl segments equal in length. This will guarantee an even feedrate during processing. The method does not require much computation time and can be used in real-time interpolators.

The equation for determining the increment at $i - th$ step is calculated based on the given feedrate value, sample time and curve characteristics. The velocity V of the tool moving along the curve can be determined by Eq. 3 with respect to the increment of u:

$$V(u) = \frac{ds}{dt} = \frac{ds}{du}\frac{du}{dt} \tag{3}$$

Then the increment Δu equals to:

$$\frac{du}{dt} = \frac{V}{ds/du} = \frac{V}{\sqrt{(x')^2 + (y')^2}}$$

The final equation for the next u value can be obtained with the use of Taylor's expansion. The first order approximation is possible since the interpolation time usually has small value, the assumption takes the form as shown (Eq. 4).

$$u_{i+1} = u_i + T\dot{u}_i = u_i + \frac{VT}{\sqrt{(x')^2 + (y')^2}} \tag{4}$$

So, to calculate the parameter increment Δu on the step $i + 1$, Eq. 5 can be applied.

$$\Delta u_{i+1} = \frac{VT}{\sqrt{(x')^2 + (y')^2}} \tag{5}$$

Since the algorithm allows obtaining a stable feedrate and solves the problem of determining the optimal increment Δu, the use of this calculation method for interpolating curvilinear trajectories seems appropriate.

To analyze the difference in obtained values after applying the mentioned segmentation methods the example of a curve is considered below.

C. Simulation system characteristics

All given examples have been implemented using the Python programming language version 3.6.5, charts are made with the Matplotlib plotting library version 2.2.2 and calculations are made using Numpy library version 1.14.2. CPU's clock speed is 2.20 GHz, Random Access Memory volume is 4 Gb, operation system is 64-bit Windows 10.

The tool trajectory is represented in the parametric form by the Eq. 6 and is depicted in Fig. 1.

$$\begin{aligned} x &= -90u^3 + 5u^2 + 85u \\ y &= 10\sin(u) \end{aligned} \tag{6}$$

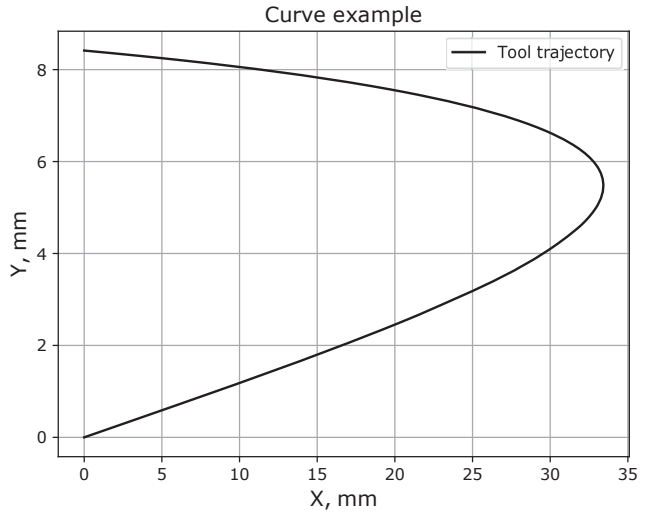

Fig. 1. Tool path example

The initial processing specifications for the simulation are given below.

```
start velocity V = 25 mm/sec;
sample time T = 0.01 sec.
```

In the case of conventional method the length of obtained segments goes from 0.35 to 0.64 mm. This significant change may cause variations in speed up to 98%. The contour error reaches 11.4 μm with the number of segments equal 100.

With the application of the proposed method the contour error stays within 8 μm, the discrepancy between the segment length does not exceed 3%. This results in a stable velocity during processing. The number of linear segments in this example equals 270. Bigger number of segments leads to greater correspondence of the initial given trajectory. It should be noted that in the case of applying traditional segmentation method with the same number of linear units (270), due to the segments length inequality, the contour error is higher than obtained by the proposed algorithm.

Summarizing the obtained results, the proposed approximation method allows to achieve minimal deviations in size of segments obtained during interpolation, which leads to stable velocity during processing.

As a part of the development for complex paths interpolation, the algorithm for arcs processing and corner rounding technique has been modified according to proposed segmentation method to satisfy the feedrate stability and are described in the following sections.

IV. Circular interpolation

Circular interpolation is performed using the proposed above algorithm for curves. The arc is represented in a parametric form determined by the Eq. 7.

$$\begin{aligned} x(u) &= a\cos(u), \\ y(u) &= a\sin(u), \\ 0 &\le u \le 2\pi \end{aligned} \tag{7}$$

where a is a real number. However, in this case the range of the parameter u differs from other types of curves. It depends on the quadrant, where the arc is in, and the direction of the tool movement. Applying the parametric form of an arc in calculation of Δu increment with Eq. 8 could easily show that Δu value depends only on the velocity V and sample time T.

$$u_{i+1} = u_i + \frac{VT}{\sqrt{(-asin(u))^2 + (acos(u))^2}} = u_i + \frac{VT}{a} \tag{8}$$

So on condition that tool speed remains the same, equal segments are obtained with equal u increments. Since the uniform increments of u results in equal segments and the feedrate maintains stable, it is reasonable to calculate the Δu value only once on the first step of arc interpolation process.

For example, the equation for the tool motion along the arc in the first quadrant when moving counterclockwise is described by the Eq. 9 and shown in Fig 2:

$$\begin{aligned} x(u) &= 10\cos(u), \\ y(u) &= 10\sin(u), \\ 0 &\le u \le \frac{\pi}{2} \end{aligned} \tag{9}$$

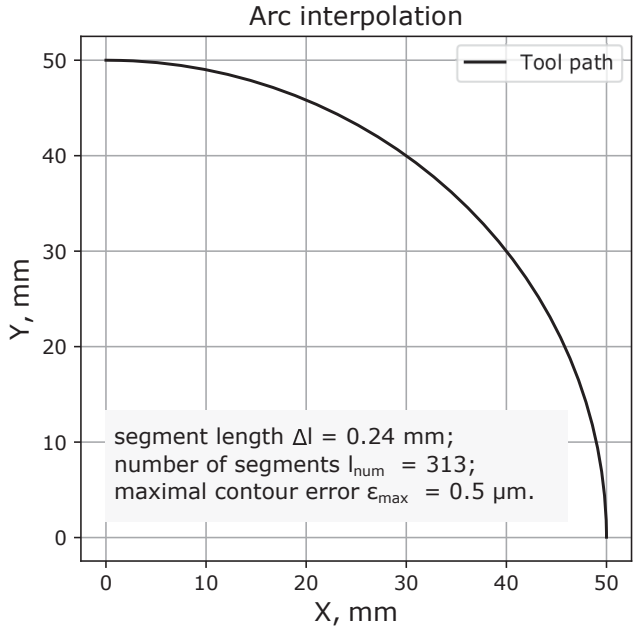

Fig. 2. Arc example

The input simulation parameters are set as follows:

```
feedrate V = 25 mm/sec;
sample time T = 0.01 sec.
```

The simulation of the defined arc results in the uniform sized segments 0.24 mm. The contour error lays within 0.5 μm. As has been mentioned above, the arc interpolation with equal increments Δu results in equal segments. However, the proposed algorithm is still preferable, as it defines the number of segments depending, besides velocity and sample time, on the size of an arc. For instance, in this case the segments quantity equals to 313. On the other hand, if an arc radius equaled to 10 mm instead of 50 mm, the number of segments decreases to 61 with the same contour error value.

V. Trajectory generation at corners

A corner rounding technique was also developed as a part of the proposed interpolation algorithm for complex trajectories. It suggests using parametric Bezier curves, which allows processing without complete tool stops at corners. The possibility to control the size and shape of rounding to achieve the required accuracy of processing and higher flexibility was considered. The calculation of a rounding is carried out based on 6 control points Bezier curve. The detailed rationale for the choice of the curve type and the number of control points for corner smoothing is given in the previous paper [18].

The curve is calculated based on the Eq. 10:

$$B(t) = \begin{bmatrix} B_{x(u)} \\ B_{y(u)} \end{bmatrix} = \left\{ \begin{array}{r} (1-u)^5 P_0 + \\ +5u(1-u)^4 P_1 + \\ +10u^2(1-u)^3 P_2 + \\ +10u^3(1-u)^2 P_3 + \\ +5u^4(1-u)P_4 + u^5 P_5 \end{array} \right\} and \tag{10}$$
$$P_0 = \begin{bmatrix} P_{0,x}(u) \\ P_{0,y}(u) \end{bmatrix}, P_1 = \begin{bmatrix} P_{1,x}(u) \\ P_{1,y}(u) \end{bmatrix}, ..., P_5 = \begin{bmatrix} P_{5,x}(u) \\ P_{5,y}(u) \end{bmatrix}$$

where $P_0, P_1, \ldots, P_5$ are control points and u changes in range between 0 and 1. The algorithm for corner processing has been modified by applying the segmentation method with equal segments. Partial derivatives x' and y' are calculated as (see Eq. 11):

$$B'(t) = \begin{bmatrix} B_{x'(u)} \\ B_{x'(u)} \end{bmatrix} = \left\{ \begin{array}{r} 5(1-u)^4(P_1 - P_0) + \\ +20u(1-u)^3(P_2 - P_1) + \\ +30u^2(1-u)^2(P_3 - P_2) + \\ +20u^3(1-u)(P_4 - P_3) + \\ 5u^4(P_5 - P_4) \end{array} \right\} and$$
$$P_0 = \begin{bmatrix} P_{0,x}(u) \\ P_{0,y}(u) \end{bmatrix}, P_1 = \begin{bmatrix} P_{1,x}(u) \\ P_{1,y}(u) \end{bmatrix}, ..., P_5 = \begin{bmatrix} P_{5,x}(u) \\ P_{5,y}(u) \end{bmatrix} \tag{11}$$

Δu is calculated based on the proposed algorithm to achieve equal segment length, according to Eq. 12:

$$\Delta u = \frac{VT}{(B_{x'(u)})^2 + (B_{y'(u)})^2} \tag{12}$$

A trajectory segment with an acute corner has been generated to display the application of the proposed segmentation method for corner rounding (see Fig. 3). Two curves of different shapes, with ratio of n (the ratio of the distance between control points) 0.95 and 0.4, are interpolated using Eq. 11-12. The initial processing parameters are listed below:

```
Trajectory p = [
[0, 0],
[50, 50]
];
feedrate V = 25 mm/sec;
sample time T = 0.01 sec.
```

Results show that the size and number of segments is different depending on the length and curvature of trajectory. However, the contour error is low in both cases and stays within 5.5 μm and the segment length is uniform which allows to achieve stable feedrate and low machining error.

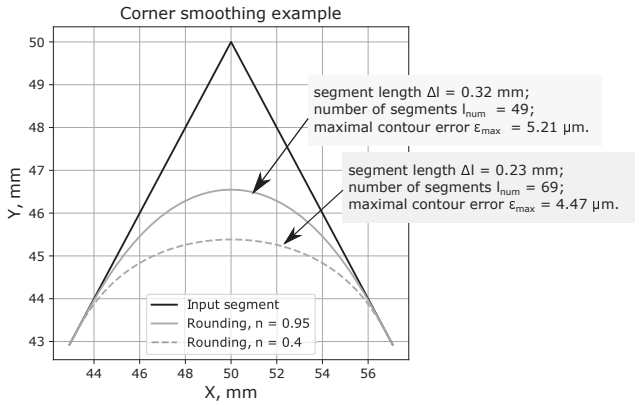

Fig. 3. The example of curves at corner

VI. Acceleration/deceleration control application

It should be noted that due to possible need for acceleration or deceleration of the tool during curve processing, segments size will not remain equal. In this case it is suggested to apply S-shape feedrate profile control to achieve smooth transition of velocity and for vibration and jerk reduction. The detailed description and rationale on the feedrate profile choice as well as the developed algorithm of acceleration/deceleration control has been given in the previous publication [18].

The S-shape feedrate scheduling is performed to minimize the velocity deviations and avoid the exceeding of the allowable acceleration A_{max} and deceleration D_{max}. The velocity on every sample time T is calculated based on the given feedrate F as the first derivative of acceleration A with $T \in [t_0, t_1]$. The following Eq. 11:

$$A = \frac{A_{max}}{2} \cdot \left(1 - \cos\left(\frac{2\pi A_{max}}{2F}\right) \cdot T\right) \tag{13}$$

and deceleration D with $T \in [t_2, t_3]$ as in Eq. 14:

$$D = -\frac{D_{max}}{2} \cdot \left(1 - \cos\left(\frac{2\pi D_{max}}{2F}\right) \cdot (T - t_2)\right) \tag{14}$$

For example, if the tool needs to accelerate from 20 to 25 mm/sec while processing the curve from the previous example

(see Fig. 1) with the allowable acceleration and deceleration A_{max} = D_{max} = 5 mm/sec, the feedrate profile will look as shown in Fig. 4. The obtained velocity values on every sample time could be used for interpolation procedure. The results show that segment length in this case will vary from 0.2 to 0.3 mm, and contour error will be within 7 μm.

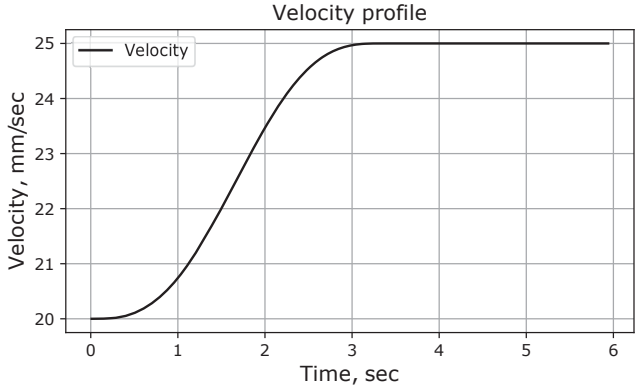

Fig. 4. The obtained feedrate profile

To summarize, the acceleration/deceleration algorithm with S-shape feedrate profile can be applied for complex trajectories, creates a smooth changes of velocity and provides a high quality surface at high speeds. Its application for curve interpolation does not affect the contour error that remains minimal.

VII. Analysis of the algorithm performance

The following aims have been achieved during the present research. The proposed interpolation algorithm allows to perform the segmentation of curvilinear paths resulting in segments of equal value by calculating the optimal increments of a parametric value u. It ensures the stable velocity value during processing. The obtained equations for calculations do not have heavy computational load and thus are applicable for the real-time interpolators. The proposed method is universal can be used for interpolation of any parametric curve, including NURBS, B-splines etc. The application of acceleration/deceleration algorithm prior to the interpolation results is possible to ensure smooth transitions of velocity.

To simplify the calculations it is reasonable to perform linear interpolation with the use of Reference-Word method and apply traditional representation method of linear trajectories instead of possible parametric representation, since the contour error does not occur in this type of path.

One of the main drawbacks of the proposed interpolation algorithm is the fact that the segment length is calculated independently of the parameters of a CNC system. Since every machine has different maximum accuracy, basic length-unit (BLU) size and other specifications the equation should be modified so that it would be possible to adjust the minimal segment length according to the actual hardware requirements.

Another problem which arises during the usage of the segmentation method is related to its dependency on the geometric parameters of a curve. It means that the number of segments are directly proportional to the size of a toolpath. This may cause low accuracy when working with small scale trajectories. The reduction of geometric characteristics impact on the process of segmentation may serve as a direction for further research.

The algorithm is implemented in the Python programming language (link to repository: `github.com/stwinter2014/Interpolation-algorithm-for-complex-trajectories`).

To sum up, the developed algorithm allows to obtain uniform sized curve segments during interpolation, which results in stable feedrate and ensures minimal contour error. The application of the feedrate control creates smooth changes of velocity on the acceleration and deceleration stages where needed. However, several improvements should be considered like the decrease of geometrical parameters (e.g. curve size) impact on segmentation and the opportunity to adjust the segmentation process according to CNC characteristics and required accuracy.

VIII. Conclusion

To summarize, an efficient interpolation algorithm for curvilinear trajectories has been developed. The use of parametric curves with the proposed segmentation method allows to calculate the optimal parameter increment resulting in equal curve interpolation units and thus maintain a stable velocity during processing. The use of the acceleration/deceleration control based on the S-shape feedrate profile creates gradual change of velocity during processing and could be applied for curve interpolation.

The performed simulation of the algorithm implementation showed its effectiveness and compliance with the initial requirements. The deviations of segment length do not exceed 3% and produce minimal contour error during simulations. The work is carried out as a continuation of a research published in Proceedings of the 23rd Conference of Open Innovations Association FRUCT in 2018 [18].

It is planned to integrate the present algorithm with the linear interpolation and feedrate control stages. A modification of the proposed segmentation method, so that the increment value could be restricted according to the CNC machine characteristics and accuracy requirements for any specific processing, is also taken into consideration. At the end of development, tests on a CNC machine will be carried out.

Acknowledgment

This work was carried out under the project no. 619296 “Technologies of cyber-physical systems: management, computing, security” conducted at the Faculty of Control Systems and Robotics, ITMO University.

References

[1] P. Smid, *CNC Programming Handbook: A Comprehensive Guide to Practical CNC Programming*, ser. Industrial Press eBooks. Industrial Press, 2003.

[2] Smoothieware official website, Smoothieboards, Web: http://smoothieware.org/smoothieboard.

[3] Linux CNC official website, Lunix CNC documents, Web: http://linuxcnc.org/documents/.

[4] Grbl, GitHub Grbl repository, Web: https://github.com/grbl/grbl.

[5] S.-H. Suh, S.-K. Kang, D.-H. Chung and I. Stroud, *Theory and Design of CNC Systems, 1st ed*. Springer Publishing Company, Incorporated, 2008.

[6] A. Hughes, *Electric Motors and Drives: Fundamentals, Types and Applications*. Elsevier Science, 2005.

[7] L.-P. Luo, C. Yuan, R.-J. Yan, Q. Yuan, J. Wu, et al., "Trajectory planning for energy minimization of industry robotic manipulators using the Lagrange interpolation method", *International Journal of Advanced Manufacturing Technology*, vol. 16, issue 5, May 2015, pp. 911–917.

[8] A. Gasparetto and V. Zanotto, "A New Method for Smooth Trajectory Planning of Robot Manipulators", *Mechanism and Machine Theory*, vol. 42, no. 4, 2007, pp. 455-471.

[9] J. Wu, H. Zhou, X. Tang, J. Chen, "Fast NURBS interpolation based on the biarc guide curve", *The International Journal of Advanced Manufacturing Technology*, vol. 58, issue 5-8, Jan. 2012, pp. 597–605.

[10] D. Du, Y. Liu, C. Yan, C. Li, "An accurate adaptive parametric curve interpolator for nurbs curve interpolation", *International Conference on Information Computing and Applications*, vol. 32, issue 9-10, April 2007, pp. 999–1008.

[11] L. Chen, X. Zhang, M. Li, "Cubic nurbs interpolation curves and its convexity", *The International Journal of Advanced Manufacturing Technology*, 2010, pp. 488-495.

[12] Kaan Erkorkmaz and Yusuf Altintas, "High speed cnc system design. Part I: jerk limited trajectory generation and quintic spline interpolation", *International Journal of Machine Tools and Manufacture*, vol. 41, issue 9, July 2001, pp. 1323-1345.

[13] B. Sencer, K. Ishizaki, and E. Shamoto, "A curvature optimal sharp corner smoothing algorithm for high-speed feed motion generation of nc systems along linear tool paths", *International Journal of Advanced Manufacturing Technology*, vol. 76, no. 9-12, 2015, pp. 1977–1992.

[14] B. Li, H. Zhang, P. Ye, "Error constraint optimization for corner smoothing algorithms in high-speed CNC machine tools", *International Journal of Advanced Manufacturing Technology*, vol. 99, issue 1-4, Oct. 2018, pp. 635–646.

[15] L. Lu, L. Zhang, S. Ji, Y. Han, J. Zhao, "An offline predictive feedrate scheduling method for parametric interpolation considering the constraints in trajectory and drive systems", *International Journal of Advanced Manufacturing Technology*, vol. 83, no. 9-12, April 2016, pp. 2143–2157.

[16] H. Li, Z. Gong, W. Lin, and T. Lippa,"Motion profile planning for reduced jerk and vibration residuals", *SIMTech technical reports*, vol. 8, no. 1, 2007, pp. 32–37.

[17] Y. Koren, C. C. Lo, M. Shpitalni, "CNC interpolators: algorithms and analysis", *Manufacturing Sciem:e and Engineering*, vol. 64, 1993, pp. 83-92.

[18] K.V. Zimenko, M.Y. Afanasev, A.A. Krylova, S.A. Shorokhov, and Y.V. Fedosov "Motion Profile Control Algorithm and Corner Smoothing Technique for Trajectory Optimization of High-Precision Processing", *Proceedings of the 23rd Conference of Open Innovations Association FRUCT*, 2018, pp. 425-431.