Interpolation method for robot trajectory planning

Martin Švejda, Tomáš Čechura
Department of Cybernetics
University of West Bohemia
Pilsen, Czech Republic
Email: msvejda@kky.zcu.cz, tomek89@ntis.zcu.cz

***Abstract*—This paper deals with the problem of interpolation of generated end-effector trajectories for application in robotics. For complex shaped trajectories there are often many high resolution coincident points generated from CAD/CAM systems or trajectory generators. The preprocessing (refining) is necessary for proceeding these points to the manipulator control system. Two interpolation methods are discussed: Line interpolation with polynomial blending and new proposed method Cubic spline interpolation with recalculated feed rate. Second introduced method describes how to interpolate the coincident points and compute the feed rate of the interpolation in order to reduce undesirable peaks in acceleration and ensure demanded position, velocity and acceleration profile along the trajectory. The new interpolation algorithm was implemented to the trajectory generator for complex pipe weld inspection robot. Experimental measurements made on control system of real prototype are finally presented for evaluation of proposed method performance.**

I. Introduction

The main problem in robotics regarding the trajectory planing of the manipulator end-effector is correct computation of the demanded setpoints (position and orientation) in task space and their real-time interpretation in the control system of the manipulator. There can be found many methods for dealing with this problem. Note, that many algorithms for trajectory planning of manipulator are assumed for direct computation of joint coordinates of manipulator depending on the given initial and terminal point along the trajectory. This is typical example for the joint trajectory planning of standard industrial manipulator (namely for pick and place applications). In this case, the trajectory planning algorithm computes initial and terminal position of the joints from initial and terminal position of the end-effector given in the task space and the interpolation is performed in the joint space as time synchronous line interpolation between these joints position with respect to given constraints (maximal velocity, acceleration, etc. of the joint actuators). Unfortunately, this approach is much more suitable for simple pick and place applications because the shape of the trajectory between initial and terminal points in the task space is given by nonlinear kinematic transformation forming hardly predictable behaviour of the end-effector in the task space.

The presented paper deals with the problem of the end-effector trajectory planning where the precise following of the desired trajectory in the task space plays important role concerning robots for arc welding and machining, non-destructive testing (NDT) and inspecting, etc. The main aim of the following research is develop the trajectory planning algorithm for NDT manipulator which is supposed to inspect the welds of the pipes of complex geometries. New presented algorithm is convenient for interpolation of the position coincident points from weld trajectory generators because of the following important properties: natural interpolation through given position coincident points with respect to continuity of velocity and acceleration, possibility to interpolate large amount of the coincident points with small mutual distance (precise points of given resolution from trajectory generators), required velocity and acceleration profile. The paper is divided as follows: Firstly the standard line interpolation with polynomial blending algorithm is used for trajectory planning of the end-effector coordinates in the task space. Secondly considering some drawbacks of the algorithm the new cubic spline interpolation algorithm with recalculated feedrate is proposed. Finally new algorithm is used for trajectory planning of the NDT manipulator for pipe welds.

II. Discussion on convenient trajectory planning algorithms

Complex overview of the robot trajectory planning can be found in [1], [2], [3] but mainly for joint space planning. In addition, there are only the algorithms where the time for given position coincident points have to be known in advance. But this approach is not convenient for our purposes because the knowledge of the interpolated trajectory properties such velocity and acceleration profile are much more significant. The exact time stamps of the coincident points are only the consequence of required trajectory properties and it is difficult to determine them directly from generated position data.

A. Line interpolation with polynomial blending

We consider only algorithms for trajectory planning in the task space because the motion of the end-effector is supposed to be fully controlled between given task space setpoints. There is possibility to modify algorithm which is originally used for joint space trajectory planning and interpolate individual coordinates in the task space with the line segments and suitable polynomial blending segments in the vicinity of the coincident points. It is clear that this approach is only approximation because the coincident points are not intersected exactly. On the other hand the line segments ensures native shape of the trajectory (from point to point) and the blending segment can be parametrized appropriately. Firstly we suppose one 1-dimensional coordinate p and m desired coincident point p^k and the constraints on maximal velocity $\dot{p}^k_{max}$ and acceleration $\ddot{p}^k_{max}$ in each segment. Line interpolation with polynomial blending is depicted in Fig. 1. The starting and the terminal

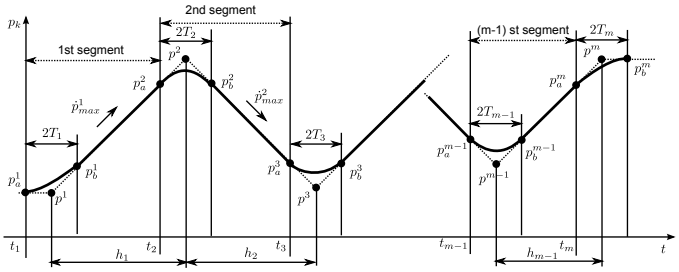

Fig. 1. Scheme of line interpolation with polynomial blending

point of each polynomial blending segment is given as:

$$p_a^k = p^k - T_k \dot{p}_{max}^k, \quad p_b^k = p^k - T_k \dot{p}_{max}^k \tag{1}$$

where $t_1 = 0$, $t_k = T_1 - T_k + \sum_{i=1}^{k-1} h_i$, $k = 2 \dots m$ and h_i is time between p^k and p^{k+1} for given constant velocity $\dot{p}_{max}^k$.

Interpolation of the k-th line segment is obvious and is given as:

$$p(t) = (t - t_k - T_k)\dot{p}_{max}^k + p^k \text{ for } t \in \langle t_k + 2T_k, t_{k+1} \rangle \tag{2}$$
$$\dot{p}(t) = \dot{p}_{max}^k, \quad \ddot{p}(t) = 0$$

Substituting boundary condition in starting and terminal points of blending segments on position, p_a^k, p_b^k, see (1), velocity $\dot{p}_{max}^{k-1}$, $\dot{p}_{max}^k$ and zero acceleration to 5-th order polynomial function (6 conditions for 6 unknown parameters) we get only the 4-th order polynomial function for k-th blending segment as follows:

$$p(t) = p^k - \frac{1}{16T_k^3}(t - t_k)^3(t - t_k - 4T_k)(\dot{p}_{max}^k - \dot{p}_{max}^{k-1}) + \tag{3}$$
$$+ (t - t_k - T_k)\dot{p}_{max}^{k-1}$$
$$\dot{p}(t) = \dot{p}_{max}^{k-1} - \frac{1}{4T_k^3}(t - t_k)^2(t - t_k - 3T_k)(\dot{p}_{max}^k - \dot{p}_{max}^{k-1})$$
$$\ddot{p}(t) = -\frac{3}{4T_k^3}(t - t_k)(t - t_k - 2T_k)(\dot{p}_{max}^k - \dot{p}_{max}^{k-1})$$

for $t \in \langle t_k, t_k + 2T_k \rangle$.

Furthermore, we take into account the constraints on maximal acceleration $\ddot{p}_{max}^k$ (only in the blending segments, in the line segments the acceleration is zero). It can be easily shown from the necessary condition $\frac{d}{dt}\ddot{p}(t) = 0$ that the extreme of acceleration occurs for time $t = t_k + T_k$. Substituting the condition to the acceleration $\ddot{p}(t)$ the term for unknown time difference T_k is expressed as:

$$T_k = \frac{3}{4\ddot{p}_{max}^k}|\dot{p}_{max}^k - \dot{p}_{max}^{k-1}| \tag{4}$$

In order to use 1-dimensional line interpolation with polynomial blending algorithm for trajectory planning of manipulators the problem have to be extended to multidimensional case. We suppose that position and orientation of the end-effector are generally given with six dimensional coordinate $\boldsymbol{P} = [p^m]$ for $m = \{x, y, z, \alpha, \beta, \gamma\}$ where x, y, z are position coordinates and α, β, γ are orientation coordinates given by the Euler angles. The constraints are give together for translation movement $v_{max_{pos}}$, $a_{max_{pos}}$ (norm of velocity and acceleration) and for each orientation coordinate v_{max_α}, a_{max_α}, v_{max_β}, a_{max_β}, v_{max_γ}, a_{max_γ}. In order to synchronize the motion of task space coordinates $\boldsymbol{P}_i$ the maximal velocities and accelerations have to be recomputed. One of the possibilities is depicted in Fig. 2 where we suppose that time for movement through the segment will be the same for all coordinates.

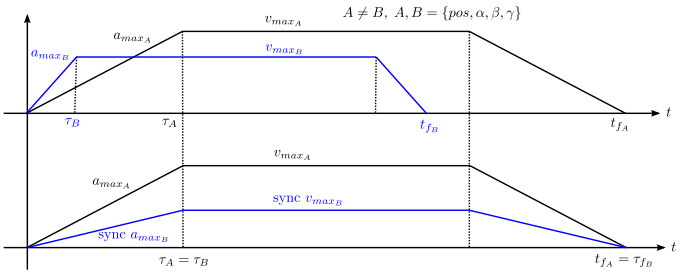

Fig. 2. Synchronisation of velocity and acceleration constraints in the segment.

Synchronized constraints for velocity and acceleration $v_{max_\star}^{\text{sync}}$, $a_{max_\star}^{\text{sync}}$, $\star = \{pos, \alpha, \beta, \gamma\}$ in each segments follow from comparison the final time t_f and from the condition on the same acceleration time τ for coordinates.

$$v_{max_\star}^{\text{sync}} = \lambda_\star v_{max_\star}, \; a_{max_\star}^{\text{sync}} = \vartheta_\star v_{max_\star} \tag{5}$$

where

$$\lambda_\star = \min\left[1, \frac{v_{max_\dagger}}{v_{max_\star}} \frac{\star_{max}}{\dagger_{max}}\right], \; \vartheta_\star = \min\left[1, \frac{a_{max_\dagger}}{a_{max_\star}} \frac{\star_{max}}{\dagger_{max}}\right] \tag{6}$$

for all $\dagger = \{pos, \alpha, \beta, \gamma\}$ and $\dagger_{max}$, $\star_{max}$ is traveled distance of given coordinate between segments.

Synchronous velocity $v_{max_\star}^{\text{sync}}$ is used for calculating time h_i between coincident points, see (1), (now it is the same for all coordinates in the segment). Synchronous velocity $v_{max_\star}^{\text{sync}}$ and acceleration $a_{max_\star}^{\text{sync}}$ is used for calculating time difference T_k in (4). Note that synchronous acceleration $a_{max_\star}^{\text{sync}}$ is used only for the first and the last blending segments due to keeping the same time for accelerating and decelerating in these segments for all coordinates. Otherwise the maximal accelerations $a_{max_\star}$ is used for remaining internal blending segments. The result of the trajectory planning algorithm is shown in Fig. 3 for position and in Fig. 4 for orientation.

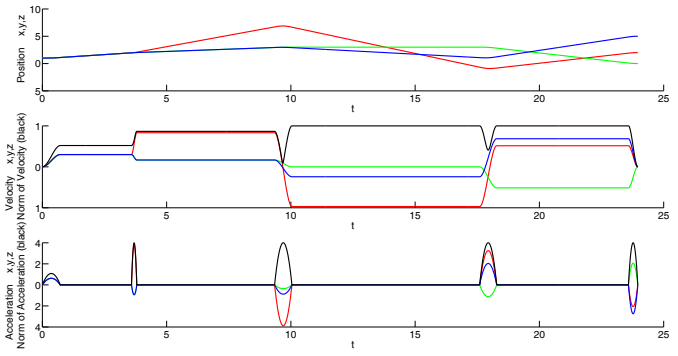

Fig. 3. Position, velocity and acceleration of translation motion (x-red, y-green, z-blue) and their norm (black). Constraints are $v_{max_{pos}} = 1$, $a_{max_{pos}} = 4$

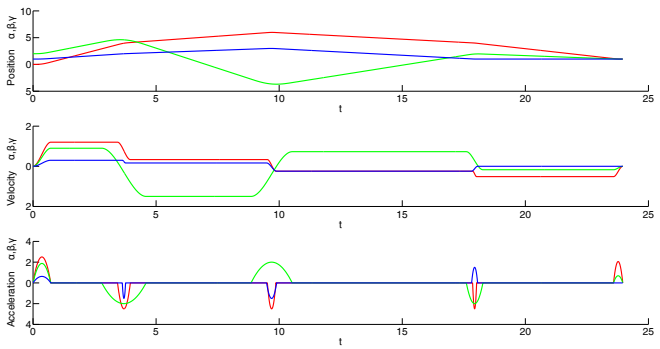

Fig. 4. Position, velocity and acceleration of rotation motion (α-red, β-green, γ-blue). Constraints are $v_{max_\alpha} = 1.2$, $a_{max_\alpha} = 2.5$, $v_{max_\beta} = 1.5$, $a_{max_\beta} = 2$, $v_{max_\gamma} = 2$, $a_{max_\gamma} = 1.5$

B. Cubic spline interpolation with recalculated feed rate

Unfortunately, it can be seen that line interpolation with polynomial blending is not very suitable for interpolation coincident point which are too close to each other. There are two main inconveniences. Firstly, the norm of the translation velocity (or velocity of Euler angles) is not constant along the blending segments, see Fig. 3 and there are many applications where it can be fundamental problem (e.g. arc welding, precise NDT testing, etc.). Secondly, there are peaks in the acceleration when motion is passing through blending segments, see Fig. 3, 4. This fact is getting much more important mainly when there are many close coincident points for interpolation because of possibility to excite undesirable vibrations of the manipulator. Therefore, the line interpolation with the polynomial blending is much more convenient for pick and place applications. Therefore the new algorithm based on the spline interpolation with appropriate feed rate computation was derived in order to interpolate coincident task space points.

1) Interpolation coincident points by the cubic spline: We suppose the interpolation of i-th segment of coordinates p_i^m, $m = \{x, y, z, \alpha, \beta, \gamma\}$ by the cubic polynomial with parameter φ, see Fig. 5. Position and directional derivatives are given in the matrix form as:

$$\begin{aligned} p_i^m(\varphi) &= \begin{bmatrix}\varphi^3 & \varphi^2 & \varphi & 1\end{bmatrix} \cdot A_i^m \\ \frac{\partial p_i^m(\varphi)}{\partial \varphi} &= \begin{bmatrix}3\varphi^2 & 2\varphi & 1 & 0\end{bmatrix} \cdot A_i^m \\ \frac{\partial^2 p_i^m(\varphi)}{\partial \varphi^2} &= \begin{bmatrix}6\varphi & 2 & 0 & 0\end{bmatrix} \cdot A_i^m \end{aligned} \tag{7}$$

where $A_i^m = \begin{bmatrix}a_{i3}^m & a_{i2}^m & a_{i1}^m & a_{i0}^m\end{bmatrix}^T$ are polynomial parameters for each coordinate and segment. It is clear that there are more possibilities for choosing the polynomial parameters. It was shown experimentally that the best choice with respect to natural behaviour of interpolation curve is only the continuity of position, velocity and acceleration in the coincident points (it generates minimal variance in the acceleration along the trajectory). Therefore the boundary condition of each segment is given for N coincident points P_i^m as:

Continuity of position:

$$\begin{bmatrix}0 & 0 & 0 & 1\end{bmatrix} \cdot A_i^m = P_i^m, \ \begin{bmatrix}1 & 1 & 1 & 1\end{bmatrix} \cdot A_i^m = P_{i+1}^m \tag{8}$$

where $i = 1 \dots N-1$ results in $2N-2$ equations.

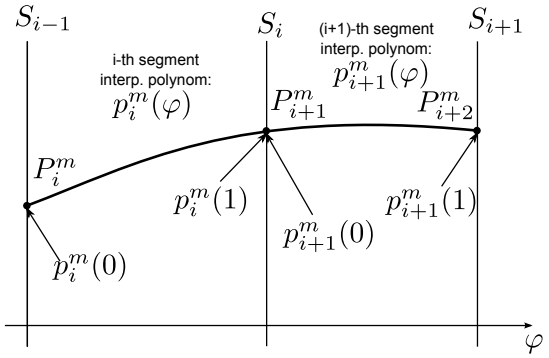

Fig. 5. Cubic spline interpolation

Continuity of velocity (w.r.t. φ):

$$\begin{bmatrix}3 & 2 & 1 & 0\end{bmatrix} \cdot A_i^m = \begin{bmatrix}0 & 0 & 1 & 0\end{bmatrix} \cdot A_{i+1}^m \tag{9}$$

where $i = N-2$ results in $N-2$ equations.

Continuity of acceleration (w.r.t. φ):

$$\begin{bmatrix}6 & 2 & 0 & 0\end{bmatrix} \cdot A_i^m = \begin{bmatrix}0 & 2 & 0 & 0\end{bmatrix} \cdot A_{i+1}^m \tag{10}$$

where $i = 1 \dots N-2$ results in $N-2$ equations.

Zero acceleration (w.r.t. φ) in the first and last segments:

$$\begin{bmatrix}0 & 2 & 0 & 0\end{bmatrix} \cdot A_1^m = \mathbf{0}, \ \begin{bmatrix}6 & 2 & 0 & 0\end{bmatrix} \cdot A_{N-1}^m = \mathbf{0}$$

results in 2 equation.

In total, we get $4N-4$ equations (8, 9, 10, 11) for $4(N-1)$ unknown parameters $\boldsymbol{A}_i^m$, $i = 1 \dots N-1$ for each coordinate $m = \{x, y, z, \alpha, \beta, \gamma\}$. It results in solving linear system of equation with sparse matrices:

$$\begin{bmatrix} \boldsymbol{\varphi}_0^3 & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \boldsymbol{\varphi}_0^3 & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \boldsymbol{\varphi}_0^3 \\ \hline \boldsymbol{\varphi}_1^3 & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \boldsymbol{\varphi}_1^3 & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \boldsymbol{\varphi}_1^3 \\ \hline \boldsymbol{\varphi}_1^2 & -\boldsymbol{\varphi}_0^2 & \mathbf{0} \\ \mathbf{0} & \boldsymbol{\varphi}_1^2 & -\boldsymbol{\varphi}_0^2 \\ \hline \boldsymbol{\varphi}_1^1 & -\boldsymbol{\varphi}_0^1 & \mathbf{0} \\ \mathbf{0} & \boldsymbol{\varphi}_1^1 & -\boldsymbol{\varphi}_0^1 \\ \hline \boldsymbol{\varphi}_0^1 & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \boldsymbol{\varphi}_0^1 & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \boldsymbol{\varphi}_1^1 \end{bmatrix} \cdot \begin{bmatrix} A_1^m \\ A_2^m \\ A_3^m \end{bmatrix} = \begin{bmatrix} P_1^m \\ P_2^m \\ P_3^m \\ \hline P_2^m \\ P_3^m \\ P_4^m \\ \hline \mathbf{0} \\ \mathbf{0} \\ \hline \mathbf{0} \\ \mathbf{0} \\ \hline \mathbf{0} \\ \mathbf{0} \end{bmatrix} \tag{11}$$

where

$$\begin{aligned} \boldsymbol{\varphi}_0^3 &= \begin{bmatrix}0 & 0 & 0 & 1\end{bmatrix}, & \boldsymbol{\varphi}_1^3 &= \begin{bmatrix}1 & 1 & 1 & 1\end{bmatrix} \\ \boldsymbol{\varphi}_0^2 &= \begin{bmatrix}0 & 0 & 1 & 0\end{bmatrix}, & \boldsymbol{\varphi}_1^2 &= \begin{bmatrix}3 & 2 & 1 & 0\end{bmatrix} \\ \boldsymbol{\varphi}_0^1 &= \begin{bmatrix}0 & 2 & 0 & 0\end{bmatrix}, & \boldsymbol{\varphi}_1^1 &= \begin{bmatrix}6 & 2 & 0 & 0\end{bmatrix} \end{aligned}$$

Note that it is necessary to solve the system of equations with large sparse matrices for the cubic spline interpolation curve of each coordinate (there exist effective methods for solving, e.g. `sparse` in Matlab). Interpolation curves for the coordinates with the same coincident points as we used above are shown in Fig. 6.

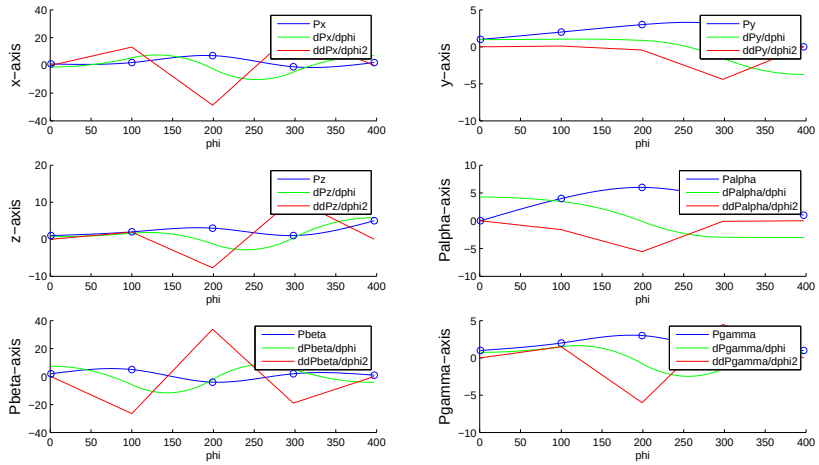

Fig. 6. Cubic interpolation curves $p^m(\varphi)$ and corresponding directional derivatives as a function of parameter φ.

2) Feed rate computation: The well-known problem for parametrization of the curve is computation of the parameter φ in such a way the arc-length, velocity and acceleration along the curve comply desired position, velocity and acceleration profile (typically trapezoidal velocity profile given by the maximal velocity and acceleration constraints v_{max}, a_{max}). The arc-length $s(\varphi(t))$ of the parametrization $\mathbf{\Phi}(\varphi(t))$ (only the translation) is given as:

$$s(\varphi(t)) = \int_0^{\varphi(t)} \| \frac{\partial \mathbf{\Phi}(\varphi(t))}{\partial \varphi(t)} \| \, d\varphi(t), \quad \mathbf{\Phi}(\varphi(t)) = \begin{bmatrix} p_i^x(\varphi(t)) \\ p_i^y(\varphi(t)) \\ p_i^z(\varphi(t)) \end{bmatrix} \tag{12}$$

The transformation between velocity and acceleration of arc-length $s(t)$ and parameter $\varphi(t)$ is given by the linear relationship (for the given position). It can be derived by time derivative of (12) as follows:

$$\dot{\varphi}(t) = \frac{1}{\| \frac{\partial \mathbf{\Phi}(t)}{\partial \varphi(t)} \|} v(t) \tag{13}$$

$$\ddot{\varphi}(t) = \frac{1}{\| \frac{\partial \mathbf{\Phi}(t)}{\partial \varphi(t)} \|} \cdot \left[a(t) - \frac{\left(\frac{\partial \mathbf{\Phi}(t)}{\partial \varphi(t)} \right)^T \cdot \frac{\partial^2 \mathbf{\Phi}(t)}{\partial \varphi^2(t)} \cdot (v(t))^2}{\| \frac{\partial \mathbf{\Phi}}{\partial \varphi(t)} \|^3} \right]$$

It can be easily shown that the integral (12) is not solvable in closed form for cubic polynomial parametrization $\mathbf{\Phi}(\varphi(t))$. Therefore the new numerical algorithm was derived:

Using adaptive Simpson's quadrature, see [4], (e.g. `quad` in Matlab) the integral (12) is numerically computed with given accuracy for each segment. It results in the arc-length sequence $S_1 \dots S_{N-1}$, where S_i represent the arc-length along the trajectory from the beginning of the trajectory to $i+1$-th coincident point.

The actual segment (active cubic polynomial) can be determined from desired position profile $s(t)$ and known sequence of arc-length $S_1 \dots S_{N-1}$. Suppose that the i-th segment is actual. The correct value of the parameter $\varphi(t) \in \{0, 1\}$ in the i-th segment corresponding to desired arc-length $s(t)$ is calculated using the following iteration:

$$\varphi_{k+1} = \varphi_k + \| \frac{\partial \mathbf{\Phi}_i(\varphi)}{\partial \varphi} \|^{-1} \cdot \left((s - S_{i-1}) - \underbrace{\int_0^{\varphi_k} \| \frac{\partial \mathbf{\Phi}_i(\varphi)}{\partial \varphi} \| \, d\varphi}_{\text{Adaptive Simpson's quadrature}} \right) \tag{14}$$

Note that there are only several iterations needed for sufficient accuracy for the coincident point close to each other.

For known $\varphi(t)$, the equations (13) are used to express the velocity $\dot{\varphi}(t)$ and the acceleration $\ddot{\varphi}(t)$ of the parameter from desired velocity $v(t)$ and the acceleration $a(t)$ profile.

Finally, the interpolated point $p_i^m(\varphi(t))$ in the segment is computed according to (7) and corresponding velocity $\dot{p}_i^m(\varphi(t))$ and the acceleration $\ddot{p}_i^m(\varphi(t))$ are given by time derivative of (7) for known $\varphi(t)$, $\dot{\varphi}(t)$, $\ddot{\varphi}(t)$ as follows:

$$\dot{p}_i^m(\varphi(t)) = \frac{\partial p_i^m(\varphi)}{\partial \varphi} \dot{\varphi}(t) = \begin{bmatrix} 3\varphi(t)^2 & 2\varphi(t) & 1 & 0 \end{bmatrix} \cdot A_i^m \cdot \dot{\varphi}(t) \tag{15}$$

$$\begin{aligned} \ddot{p}_i^m(\varphi(t)) &= \frac{\partial^2 p_i^m(\varphi)}{\partial \varphi^2} \dot{\varphi}^2(t) + \frac{\partial p_i^m(\varphi)}{\partial \varphi} \ddot{\varphi}(t) = \\ &= \begin{bmatrix} 6\varphi(t) & 2 & 0 & 0 \end{bmatrix} \cdot A_i^m \cdot \dot{\varphi}^2(t) + \\ &+ \begin{bmatrix} 3\varphi(t)^2 & 2\varphi(t) & 1 & 0 \end{bmatrix} \cdot A_i^m \cdot \ddot{\varphi}(t) \end{aligned}$$

The interpolated trajectory through given coincident points in time domain is depicted in Fig. 7. Note that peaks in acceleration are caused by the requirements of a constant velocity along the trajectory in spite of the coincident points significantly change. This effect will be reduced for points to be generated in finer resolution (e.g. for points derived from CAD systems or trajectory generators for welds in Section III). The feed rate function $s(t) \to \varphi(t)$ and its time derivative are depicted in Fig. 8 for position, velocity and acceleration profile chosen for the constraints $v_{max} = 1$, $a_{max} = 4$.

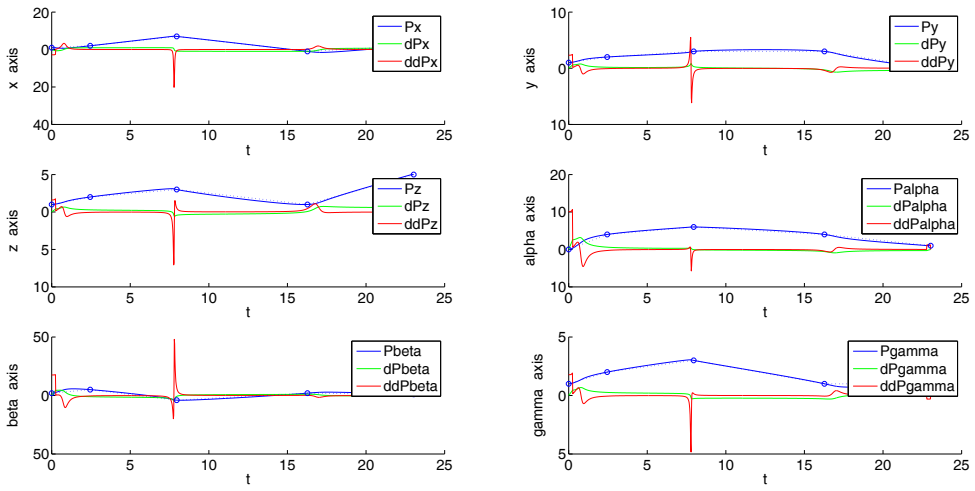

Fig. 7. Cubic interpolation curves $p^m(\varphi)$ and its time derivative as a function of time.

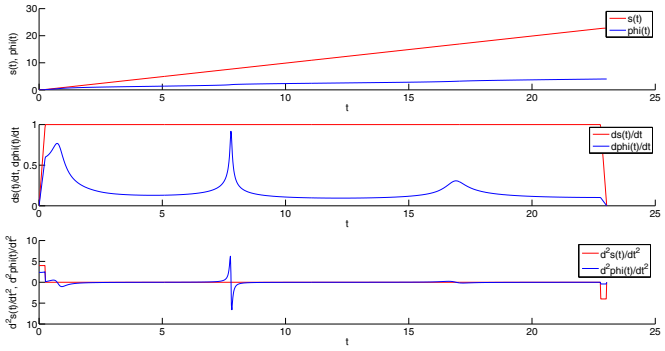

Fig. 8. feed rate for cubic interpolation curves. Desired position $s(t)$, velocity $v(t)$ and acceleration $a(t)$ profile (red), computed position, velocity and acceleration of parameter $\varphi(t)$ (blue).

III. Application in Non-Destructive Testing

The above stated interpolation methods were developed and tested on prototype application in ultrasonic NDT of complex pipe welds. Robotic manipulator SAVA has 5 DOFs [5] and is capable of testing branch, circumferential, longitudinal welds and both types of elbow welds. The prototype is controlled by real-time Control System REX [6].

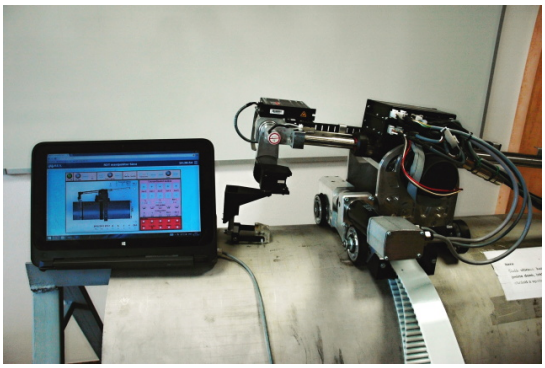

Fig. 9. Laboratory set up of robotic manipulator SAVA for NDT of complex welds

We have developed standalone application for generating NDT testing trajectories according to a currently inspected weld. It integrates trajectory generator which produces coincident points along the inspected weld trajectory with given resolution and the second proposed interpolation method II-B. Data output of the application is exported to the robot control system as a sequence of task space coordinates with specific equal time differences.

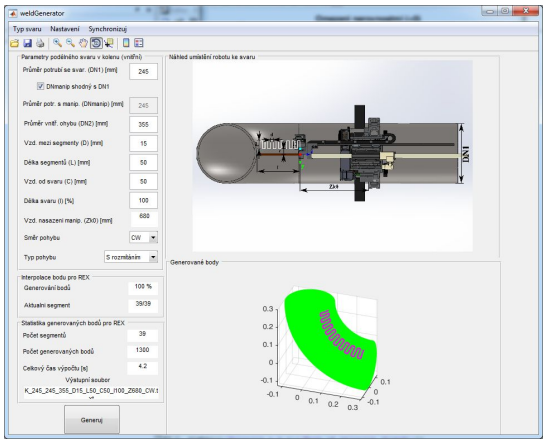

Fig. 10. Screen-shot of standalone application for generating inspection trajectories (interior elbow weld with meandering motion)

In order to process the sequence of task space coordinates by the manipulator we have implemented a new function block *RM_Feed*. The period of executing the *RM_Feed* block have to be the same as the period set in the standalone application. Simple idea of proposed function block is to process one set of task space coordinates in every execution. Other benefit of the stated block is a possibility to test any interpolation method because it does not add any undesirable blending or interpolation and blindly feeds task space coordinates directly to manipulator servo controllers. Results of the trajectory planning according to the new cubic spline interpolation algorithm for interior elbow weld with meandering motion is shown in Fig. 10. The motion of translation coordinates x, y, z and corresponding velocities and accelerations recorded

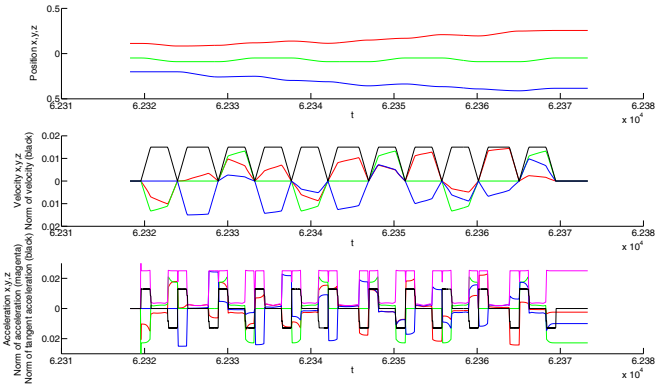

Fig. 11. Interpolated curve of translation coordinates (coincident points) by the new proposed algorithm

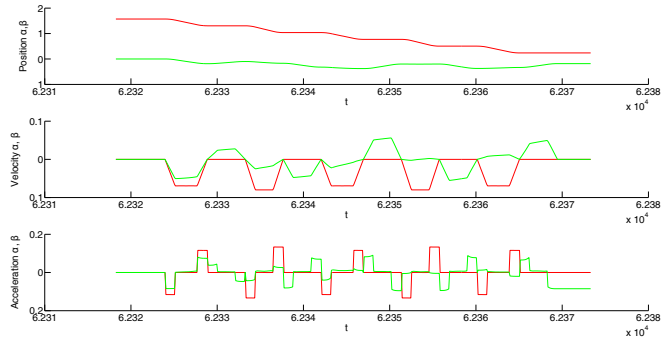

Fig. 12. Interpolated curve of orientation coordinates (coincident points) by the new proposed algorithm

during measurement on the real manipulator prototype are depicted in Fig. 11. The velocity and (tangent) acceleration profile are depicted in black colour and corresponds with demanded constraints $v_{max} = a_{max} = 0.015$. The motion of rotation coordinates α, β and corresponding velocities and accelerations are depicted in Fig. 12. It can be seen that there are no undesirable peaks in accelerations (it is caused by stopping motion in the sharp corners of the meandering motion).

IV. Conclusion

Two methods were presented for interpolation of the coincident points which are typically obtained from the data generator of complex trajectories in the task space. The first method, Line interpolation with polynomial blending, was introduced because the resulting trajectory consist only of line segments connecting coincident points and polynomial blending segments which approximate the trajectory in the vicinity of these points. Therefore, the shape of the trajectory can be fully predictable and the trajectory remains in convex area bounded by coincident points. The shape of polynomial blending segments is fully controlled by desired constraint on maximal acceleration. Unfortunately, there are some disadvantages which make this method poorly applicable in real applications of trajectory planning of manipulators concerning many mutually close coincident points. Firstly, the method gives only an approximation since it does not intersect the coincide points exactly. Secondly, it was shown that despite of acceleration along line segments are zero the acceleration along blending segments forms many undesired peaks which can lead to crucial problem considering excitation of residual vibrations in mechanical structures. In addition, the constant velocity is guaranteed only along line segments but not for blending segments (decrease of velocity). Therefore, the method is much more convenient e.g. for pick and place applications with only a few coincident points. The second new presented method was developed especially for purposes under consideration. The cubic spline polynomial was used for interpolation of coincide points. The graph of polynomial function between the coincident points plays important role considering the acceleration of the final trajectory with respect to excitation of vibrations. It was shown experimentally that the choice of minimal condition ensuring maximum smooth acceleration is formed only by the conditions on position, velocity and acceleration continuity in the initial and the final point of each cubic spline segment. In order to ensure the given position, velocity and acceleration profile (trapezoid velocity profile) along the interpolated trajectory the algorithm for feed rate computation was introduced. The numerical adaptive Simpson's quadrature is used to determine the arc-length of the interpolated trajectory. It leads to iterative algorithm for computation of position, velocity and acceleration of the cubic spline parameter satisfying demanded profile. Finally the second method was integrated to the standalone application for generating and interpolating task space trajectory for NDT of complex pipe welds. The resulting interpolated trajectory (generated in given sample times) was directly used as setpoints for control system of 5 DoF manipulator SAVA. Performance of the proposed interpolation method was demonstrated on the real prototype of the manipulator. It was verified that the generated trajectory fully complies with respect to meeting demands such as smooth of motion and the given position, velocity and acceleration profile. The extension of the interpolation algorithm by the jerk constraint is idea for further research.

Acknowledgment

This work was supported by the Technology Agency of the Czech Republic, Competence Centre CIDAM (Center for Intelligent Drives and Advanced Machine Control) project number TE02000103.

References

[1] L. Sciavicco and B. Siciliano, *Modelling and Control of Robot Manipulators*, ser. Advanced Textbooks in Control and Signal Processing, Springer, Ed. Springer London, 2000. [Online]. Available: http://books.google.fr/books?id=v9PLbcYd9aUC

[2] W. Khalil and E. Dombre, *Modeling, Identification and Control of Robots*, ser. Kogan Page Science paper edition, Butterworth-Heinemann, Ed. Elsevier Science, 2004. [Online]. Available: http://books.google.cz/books?id=nyrY0Pu5kl0C

[3] M. V. Mark W. Spong, Seth Hutchinson, *Robot Modeling and Control*. Wiley, 2005.

[4] W. Gander and W. Gautschi, "Adaptive quadrature-revisited," *BIT Numerical Mathematics*, vol. 40, no. 1, pp. 84–101, 2000.

[5] M. Švejda, "New robotic architecture for ndt applications," in *World Congress IFAC*, vol. 19, 2014, pp. 11 761–11 766.

[6] Rexcontrols.com, "Rex controls - advanced automation and control solutions," 2015. [Online]. Available: http://www.rexcontrols.com/