Accepted for publication in IEEE Transactions on Industrial Electronics — 2012

Novel on-line speed profile generation for industrial machine tool based on flexible neuro-fuzzy approximation

L. Rutkowski, *Fellow, IEEE*, A. Przybył, K. Cpałka, *Member, IEEE*

***Abstract*—Reference trajectory generation is one of the most important task in the control of machine tools. Such a trajectory must guarantee a smooth kinematics profile to avoid exciting the natural frequencies of the mechanical structure or servo control system. Moreover, the trajectory must be generated on-line to enable some feedrate adaptation mechanism working. The paper presents the on-line smooth speed profile generator used in trajectory interpolation in milling machines. Smooth kinematics profile is obtained by imposing limit on the jerk - which is the first derivative of acceleration. This generator is based on the neuro-fuzzy system and is able to adapt on-line the current feedrate to changing external conditions. Such an approach improves the machining quality, reduces the tools wear and shortens total machining time. The proposed trajectory generation algorithm has been successfully tested and can be implemented on a multi axis milling machine.**

***Index Terms*—control systems, fuzzy neural networks, intelligent control**

I. Introduction

IN the Computer Numerical Controlled (CNC) machine (Fig. 1) the high feedrate of the tool, required by high speed machining (HSM) technology, cannot be achieved at every working point because of the mechanical and electrical limitations of the machine. For example every electrical motor, used as servo-drive, has limited output power, so it can produce a limited component of centrifugal force along the toolpath. This results in a limited attainable feedrate, which depends on the current curvature of the geometrical path (Fig. 2) [1]. Moreover, in the CNC system, the feedrate and acceleration cannot be changed abruptly, because of the possibility of exciting the natural modes of the mechanical structure or servo control system. A non smooth trajectory results in a fast wear of a mechanical components of the machine.

Manuscript received December 1, 2010; revised March 18, 2011. Accepted for publication June 15, 2011.

L.Rutkowski is with the Department of Computer Engineering, Częstochowa University of Technology, Częstochowa, Poland and Information Technology Institute, Academy of Management, Łódź, Poland (e-mail: leszek.rutkowski@kik.pcz.pl).

A.Przybył and K.Cpałka are with the Department of Computer Engineering, Częstochowa University of Technology, Częstochowa, Poland.

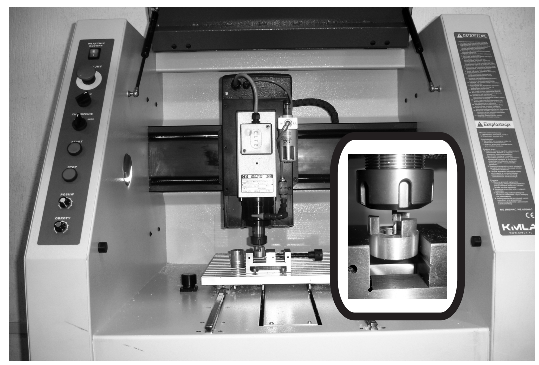

Fig. 1. Three axis milling machine used for testing the proposed algorithm.

Any control system of a CNC machine should control the servo drives in such a way, that the feedrate is as close as possible to the demanded value, and simultaneously the defined speed limits are not violated. Moreover, the generated trajectory should be smooth, to avoid exciting the natural frequencies of the machine. The smooth trajectory can be obtained by imposing limits on the first and second time derivatives of feedrate, resulting in trapezoidal acceleration profiles (Fig. 3.a). In Fig. 3.b a trapezoidal speed profile is shown. It is very popular and widely used because of its simplicity. Unfortunately, it does not guarantee a high quality of the machining because of discontinuities in acceleration reference values. In the other case, if a smooth speed profile is used (Fig. 3.a), the acceleration profile has no discontinuity and its trapezoidal form results from jerk limit. The seven segments of that speed profile have maximal, minimal or zero values of the jerk. The trajectory presented in Fig. 3.a describes a simple move from stop to stop, but if a more complex move is used (i.e. continuous move without stops) then these segments may occur in different sequences and/or amounts.

The on-line speed profile generation methods are widely investigated in literature [2]-[6]. Unfortunately, none of the reported methods were able to adjust on-line the generated speed profile to the changing external conditions with simultaneous limitation of the value of the jerk. However, it should be emphasized that feedrate adaptation mechanisms, used in

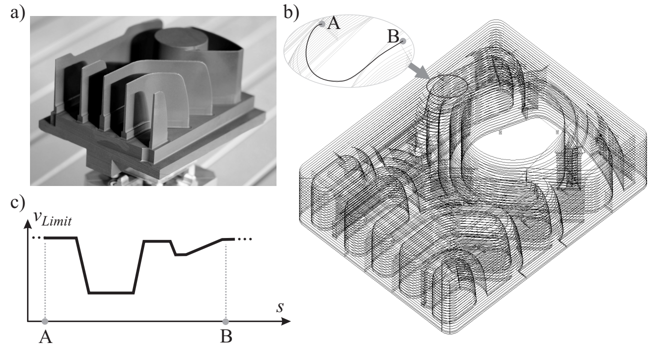

Fig. 2. Illustration of the exemplary model machined in the CNC system: (a) graphite electrode milled on the CNC machine, (b) the geometrical path of the tool designed in the CAM system, (c) the velocity limit resulting from the local curvature of the path in the indicated exemplary fragment.

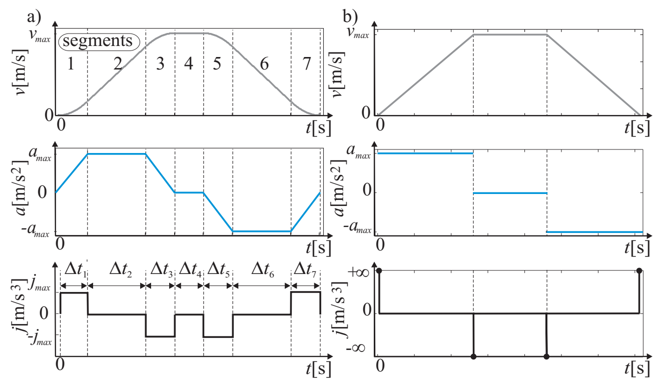

Fig. 3. Speed, acceleration and jerk profiles: (a)with and (b)without jerk limit. Segments 1, 7 - $jerk = j_{max}$; segments 3, 5 - $jerk = -j_{max}$; segments 2, 4, 6 - $jerk = 0$.

TABLE I
MAIN FEATURES OF THE ALGORITHMS FOR JERK-LIMITED SPEED PROFILE GENERATION

Algorithm	On-line operation	Jerk limitation	Computational complexity
J.-Y. Dieulot et al. [2]	no	yes	high
Y. Sun et al. [5]	yes	no	medium
H.-T. Yau et al. [8]	yes	no	low
H.-T. Yau et al. [9]	yes	yes	high
M.-S. Tsai et al. [10]	yes	yes	high
Sheng-Jung Tseng et al. [11]	no	yes	medium
R. A. Osario-Rios et al. [12]	no	yes	high
C. G. Lo Bianco et al. [13]	yes	no	low
Ch. Zheng et al. [14]	yes	no	low
our method	yes	yes	medium

high precision machines, strongly require such a feature.

In this work we are focused on the feedrate profile generation along to a toolpath. The feedrate and corresponding displacement are then converted to coordinates of all machine's axes by using the inverse kinematics. It can be always realized even if complex interpolation methods, like e.g. NURBS (Non-Uniform Rational B-Splines) based [3], [7], are applied. Such an approach enables the multi-axis interpolation task to be reduced to the one-dimensional problem. In order to simplify the terminology, in the sequel "speed" has the same meaning as the "feedrate", and "speed profile generation" and "trajectory generation" has the same meaning.

The trajectory generation task is realized by an interpolator in a real time, and then calculated reference values are supplied to the servo drives. The real time speed profile generation methods are widely investigated in literature. For example in [5] authors consider the real-time parametric interpolator, which is able to generate continuous move along a parametric curve (e.g. linear, circular, NURBS). They take into account a machine dynamics and restrictions imposed on the feedrate along a toolpath to limit a defined error between desired and obtained path of the tool. Unfortunately, in this paper the jerk limitation is not considered at all. Similarly in [8] the authors presented the real-time fast interpolation method which uses a look-ahead function to produce continuous move of the tool. However, the trapezoidal speed profile, considered by these authors, does not guarantee that jerk is limited. In [9] and [10] the authors extended the NURBS interpolation method, taking into account the jerk limitation. They proposed iterative method for generating the trajectory. Unfortunately, their results are presented only for a quite simple geometrical path, consisting of a few splines. In a practical case much more complicated geometrical paths should be used, especially in mold milling. In such a case it is likely that their method could be too time consuming. In [2] there was a proposal to use high order polynomial acceleration profile. It resulted in a more smooth move but it required much more complicated numerical calculations. The real necessity to use such an acceleration profile instead of trapezoidal one was not proved. In [11] a method for jerk-limited trajectory planning was proposed in which the parametric interpolator is composed of a look-ahead stage and a real-time sampling stage. In that method the real-time sampling stage operates on the basis of data calculated in the first (non real-time) stage. In a result, actually it is not the real-time algorithm because it is not able to on-line modify the generated speed profile. The authors in [12] focused their attention on difficulties with the on-line generation of polynomial-based trajectories, due to high computational load demand- both hardware resources and processing time. They proposed a hardware implementation of profile generation with jerk limitation, based on the Field Programmable Gate Array (FPGA) without using any multiplier. There was no discussion concerning continuous move, because only quite simple moves from stop to stop were considered. Some authors, for example [13] and [14], proposed another approach, in which the trajectory is not generated with acceleration and jerk limitations, but there is some post-filtering method used to limit them. Such an approach causes a substantial position tracking error and it is not preferred in the reference trajectory generation used in high precision CNC machines.

None of the presented methods (Table I) were able to adjust the generated speed profile to the changing external conditions, e.g. spindle load change, in an efficient manner. As we indicated, some feedrate adaptation mechanisms, used in high precision machines, require such a feature. Moreover, if a very complicated CAM model is machining, it is possible that the internal memory of the interpolator has not enough capacity to hold the whole path. In such a case the work must be divided into separate parts, what is unfavorable. The solution is to treat the limited memory of the interpolator as a dynamic buffer, which can be filling up while the machine is working. The incoming new data should be taken into account in the on-line trajectory generation method, because of a necessity to generate the continuous work without unnecessary stops.

In the paper the on-line speed profile generation method will be developed by making use of a flexible neuro-fuzzy system proposed and studied in [15]-[18]. In our investigations it is very important to achieve a very high accuracy of an approximation at a certain stage of on-line speed profile generation and the flexible neuro-fuzzy approximator satisfied our requirements. It should be noted that neuro-fuzzy structures combine the advantages of neural networks and classical fuzzy systems and are frequently applied to solve various problems of control, process modeling and fault diagnosis [4], [19]-[26]. We will develop a new method for efficient generation of a smooth velocity profile for CNC machines. The unique feature of our method, which is distinguished from other solutions, is the ability to quickly adjust the generated trajectory to changing speed limits. In our approach it is possible to modify the demanded value of the feed rate of the tool during machine operation. This feature is very important for operating CNC machines because of the need to protect the cutter from the brake and spindle from the overload in high speed machining. To our best knowledge the approach presented in this paper is the only method providing the efficient on-line smooth speed profile generator in milling machines. The idea of a new method for the on-line trajectory generation is described in Section II. In Section III.A we describe the flexible Takagi-Sugeno neuro-fuzzy system used for the speed profile generation, whereas Section III.B presents simulation results. Conclusions are given in Section IV.

II. A NEW ALGORITHM FOR ON-LINE TRAJECTORY GENERATION

Our method is based on the proposed in this paper original concept of test trajectories (Fig. 4) which are generated in fixed time periods T_G.

A. Main idea

The interpolation of a displacement, speed and acceleration as a function of time t_L, with the jerk limitation, is based on the well known motion equations

$$a\left(t_L\right) = a_0 + j_0 \cdot t_L, \tag{1}$$

$$v\left(t_L\right) = v_0 + a_0 \cdot t_L + j_0 \cdot \frac{t_L^2}{2}, \tag{2}$$

and

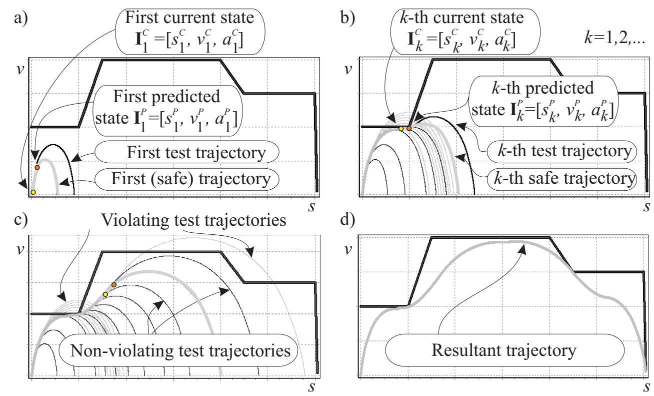

Fig. 4. Method for the on-line generation of the jerk limited trajectory (thick gray curve) taking into account the feedrate limitation (thick black curve). Thin gray and black curves represent test trajectories, violating and not violating velocity limitation, respectively, along corresponding distance.

$$s\left(t_L\right) = s_0 + v_0 \cdot t_L + a_0 \cdot \frac{t_L^2}{2} + j_0 \cdot \frac{t_L^3}{6}, \tag{3}$$

where j_0 is an applied value of the jerk, and (4) fully defines the interpolator state

$$\mathbf{s} = [s, v, a], \tag{4}$$

where subscript zero denotes values at a moment of a relative time $t_L = 0$. Based on equations (1)-(3), the on-line speed profile generator is designed. The detailed flowchart of our algorithm is presented in Fig. 5 and Fig. 9.

The interpolation is always based on a basis of a known current interpolator state

$$\mathbf{I}_k^C = \left[s_k^C, v_k^C, a_k^C\right] \tag{5}$$

and an initially known safe trajectory (step A3 in Fig. 5) which is shown in Fig. 4.a. The trajectory is defined by a set of starting values

$$\mathbf{T}_k^S = \left[\begin{array}{c} \left\{\Delta t_{k,1}^S, s_{k,1}^S, v_{k,1}^S, a_{k,1}^S\right\}, \ldots, \\ \left\{\Delta t_{k,7}^S, s_{k,7}^S, v_{k,7}^S, a_{k,7}^S\right\} \end{array} \right], \tag{6}$$

i.e.: displacement s, velocity v, acceleration a, jerk j and value of the time period Δt for seven successive segments as it is shown in Fig. 6. The safe trajectory guides the CNC machine from the starting point to the stop, guarantying the velocity, acceleration and jerk limitations.

We can easily predict (step A4 in Fig. 5) a future state of the interpolator (in a time distance T_G from the current moment) along the safe trajectory

$$\mathbf{I}_k^P = \left[s_k^P, v_k^P, a_k^P\right]. \tag{7}$$

Treating this predicted state as a starting point we can generate one test trajectory (step A7 in Fig. 5), defined by the following set of parameters

$$\mathbf{T}_k^T = \left[\begin{array}{c} \left\{\Delta t_{k,1}^T, s_{k,1}^T, v_{k,1}^T, a_{k,1}^T\right\}, \ldots, \\ \left\{\Delta t_{k,7}^T, s_{k,7}^T, v_{k,7}^T, a_{k,7}^T\right\} \end{array} \right]. \tag{8}$$

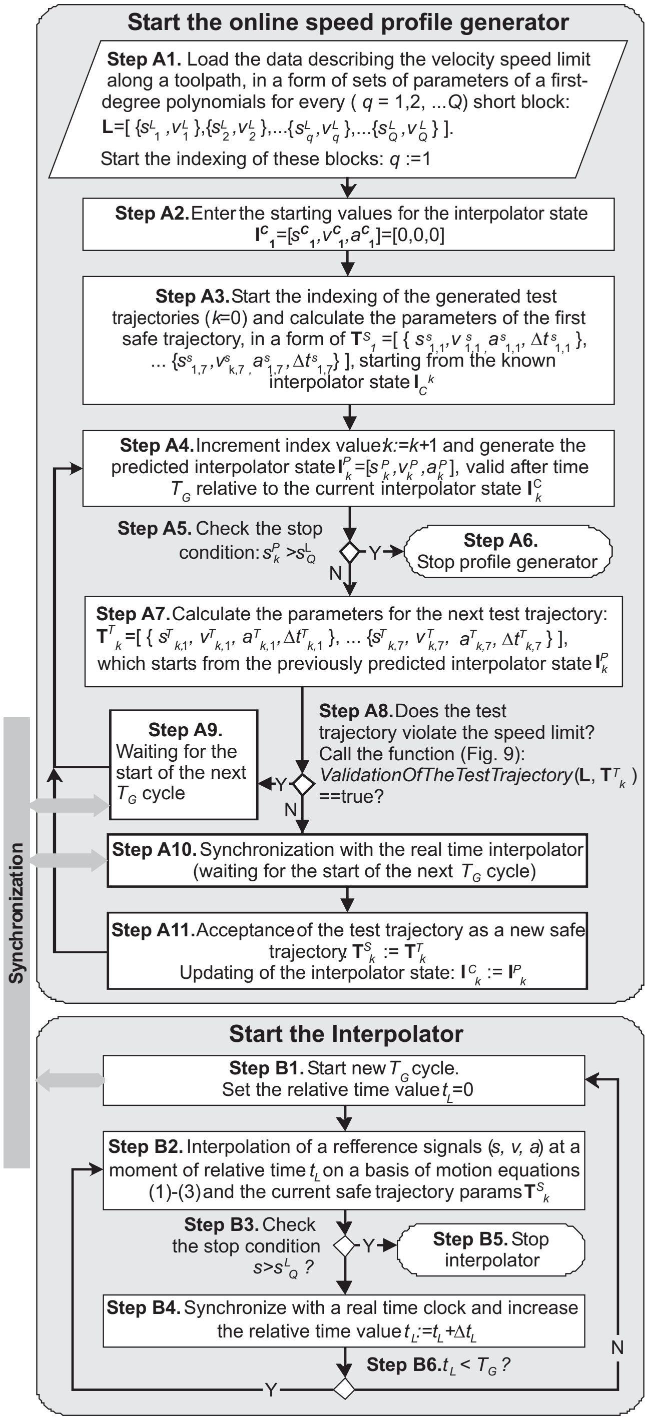

Fig. 5. Algorithm for generating test trajectories.

The test trajectory has the task to speed up the move a little bit - comparing with the move resulted from the safe trajectory (Fig. 7). This speed up can be is easily done by applying a non zero time period $(\Delta t_1^T, \Delta t_2^T, \Delta t_3^T)$ values in a seven segments speed profile. In our case sum of these three parameters was chosen experimentally and is equal to the value of parameter T_G. After this time interval the move should be immediately slowing down to the stop, in a way defined by seven segments trajectory, using segments marked as 5, 6 and 7 (Fig. 3.a). The calculations required to determine parameters

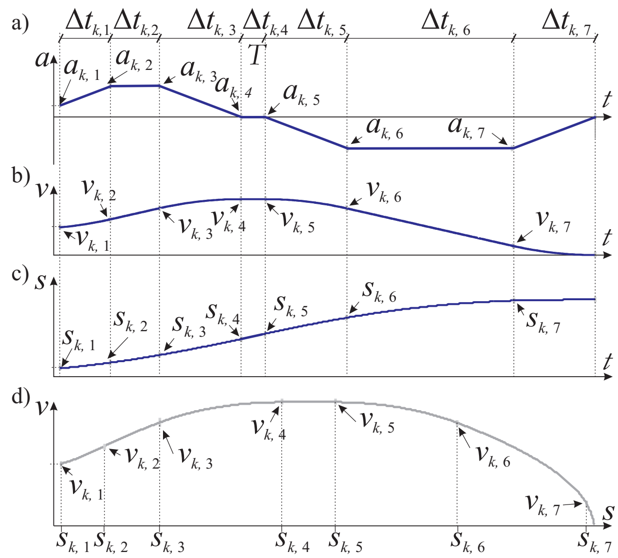

Fig. 6. Example of the seven segments trajectory, defined by a set of starting values: a)acceleration as a function of time, b)velocity as a function of time, c)displacement as a function of time, d)velocity as a function of displacement.

of seven segments trajectory are widely presented in literature [1], [27]-[29] and will not be presented here.

The generated test trajectory has to be validated (step A8 in Fig. 5 and Fig. 9), to determine if it violates or not the velocity limit along the toolpath (thick black curve in Fig. 4). The speed limit depends on the local curvature of the geometrical path designed by a CAM system [1] and this dependency can be approximated by any piecewise function, for example by the zero or higher order polynomial [28]. In this paper we use the first order polynomial described by the following sets of reference knots (Fig. 2c and Fig. 7)

$$\mathbf{L} = \begin{bmatrix} \{s_1^L, v_1^L\}, \{s_2^L, v_2^L\}, \ldots, \\ \{s_q^L, v_q^L\}, \ldots, \{s_Q^L, v_Q^L\} \end{bmatrix}, \tag{9}$$

which gives satisfactory compromise between accuracy and computational complexity. The number of blocks Q of such a piecewise curve depends on the complexity and length of the geometrical path. In our work we assume that the piecewise curve was determined in advance by a separate algorithm [8] that will not be discussed in this paper.

If a validation algorithm determines that the test trajectory violates the speed limit, this trajectory will be discarded (thin gray curve in Fig. 4) what is shown in the block diagram as step A9. Otherwise, this trajectory will be the new safe trajectory, valid after T_G time period (steps A10 and A11 in Fig. 5). This procedure is repeated in successive time periods and consecutive test trajectories are generated, each starting from the new working point (Fig. 4.b, Fig. 4.c). The final smooth speed profile (Fig. 4.d) is formed by a merger of short subsequent fragments of the safe trajectory.

Simultaneously, in a real time, with generating and validating the test trajectories, a motion controller is working (steps B1-B6). It performs the interpolation of a displacement, speed and acceleration along a tool path in the fixed time steps Δt_L. At this point the proper (adequate) inverse kinematics is also used to generate the reference values for all machine's servo

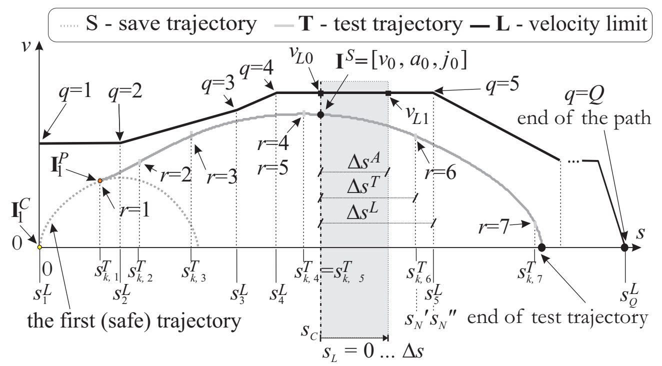

Fig. 7. Example of a test trajectory as a function of distance and intersections of its segments with speed constraints blocks. Grey area shows the currently analyzed sectors in the iterative validation algorithm of the test trajectory.

drives. This method is commonly known [29], [30] and will not be presented in this paper.

B. Method of validation of the test trajectory

In our system the validation algorithm of a test trajectory (Fig. 9) plays a key role. Analytical solution of such a task is very complicated, because velocity constraints are linear functions of a displacement given by

$$v_{Limit}\left(s_L\right) = v_{L0} + \left(v_{L1} - v_{L0}\right) \cdot \frac{s_L}{\Delta s}, \tag{10}$$

where

$$s_L = \langle 0 \ldots \Delta s \rangle , \tag{11}$$

and v_{L0}, v_{L1} are parameters of currently analyzed sector Δs, resulting from the velocity constraint curve (Fig. 7), t_L and s_L are time and position relative to the origin of the considered sector (gray area in Fig. 7), while generated speed and displacement profile are polynomial functions of time, given by (2) and (3).

Note that in Fig. 6.d as well as in Fig. 7 the velocity profiles are shown as a function of a displacement. Creating these figures was only possible on a basis of a performed iterative simulation.

It is clear that the validation of the test trajectory can not be done in one step. The velocity limit curve and the test trajectory are defined in blocks or segments, respectively. As a result the validation algorithm must be an iterative, with the number of the iterations resulting from the number of blocks of the limit curve and values of parameters of the test trajectory. A length of currently analyzed sector (Δs) in successive iterations results from an intersection of the trajectory segments and the speed constraints blocks (Fig. 7). Analyzed sectors must be iterated in such a way that they do not cross the boundaries resulting from the blocks of the velocity limit curve and the segments of the test trajectory.

Our method, which is based on a quadratic approximation of the test trajectory, requires to satisfy the following limitation; the value of Δs cannot be greater than parameter Δs^A, which is explained in the sequel in this section, otherwise the approximation accuracy will be poor. As a result the sector's length is determined as a minimal value of three values (Δs^A, Δs^T, Δs^L), as it is shown in Fig. 7, i.e.

$$\Delta s = min(\Delta s^A, \Delta s^T, \Delta s^L). \tag{12}$$

Appropriate iterations to determine Δs are shown in Fig. 9 as steps C2-C6 and steps C13-C18.

To validate the whole test trajectory, all successive sectors must be tested until the trajectory segments (8) end (step C19 in Fig. 9). If any of the tested sectors violates the speed limit

$$v\left(s_L\right) \le v_{Limit}\left(s_L\right), s_L \in < 0, \ldots, \Delta s >, \tag{13}$$

where

$$v\left(s_L\right) = v\left(s_L, v_0, a_0, j_0\right), \tag{14}$$

then the whole tested trajectory must be discarded (step C12 in Fig. 9)

In order to check analytically if the test trajectory at a given sector violates or not the speed limit, we should have it in a form of a function of the displacement. Unfortunately, there is no simple analytical projection, converting the test trajectory from a function of time (Fig. 6.b) to a function of distance (Fig. 6.d). It results from the fact that equation (14) is an implicit function. Therefore, we propose a neuro-fuzzy structure to build an efficient validation system checking if the test trajectory violates the velocity limit. More precisely, we develop the algorithm, depicted in Fig. 8, in which the neuro-fuzzy structure efficiently aids the quadratic approximation given by

$$v\left(s_L\right) \approx v^A\left(s_L\right) = C_0 + C_1 \cdot s_L + C_2 \cdot s_L^2, s_L \in < 0, \ldots, \Delta s > \tag{15}$$

of function (14). The neuro-fuzzy system is used in our concept to aid the classical quadratic approximation of the speed profile, but not to directly approximate this profile. Such an approximated function reduces mentioned earlier complex calculation, checking condition (13), to a simple task of solving a quadratic inequality, related to checking condition given by

$$v^A\left(s_L\right) \le v_{Limit}\left(s_L\right), s_L \in < 0, \ldots, \Delta s > . \tag{16}$$

The effective method of determining the coefficients of equation (15) with the help of neuro-fuzzy system will be presented in the following part of this section.

The quadratic approximation of function (14) is always possible with a defined maximum acceptable approximation error ($v_E < v_{Emax}$ in Fig. 10), if the approximation distance Δs does not exceeds Δs^A, which depends on the current curvature of the approximated function (14). The value of Δs^A, depends on the three parameters, i.e.

$$\Delta s^A = \Delta s^A\left(v_0, a_0, j_0\right), \tag{17}$$

which fully defines the interpolator state at the origin of the analyzed sector (Fig. 7).

Unfortunately, this dependency is not known in advance and can only be obtained by the trial and error method, based on

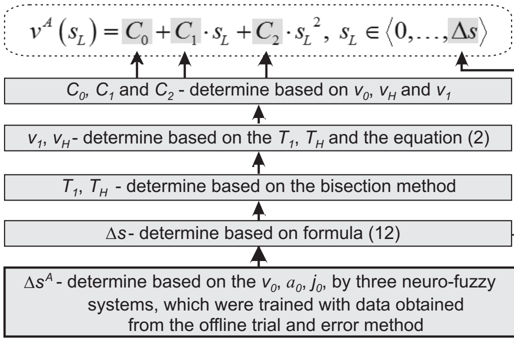

Fig. 8. Flowchart illustrating how the neuro-fuzzy structure efficiently aids the quadratic approximation of the test trajectory, as a function of the displacement in the subsequent segments.

many repeated iterative simulations, with an usage of motion equations (1)-(3). Because the trial and error method is very time consuming, it is not suitable to use in the validation system. Fortunately, we can use the neuro-fuzzy structure to approximate dependency (17) in an efficient manner.

Finally, if we know the value of Δs^A, we can approximate function (14) in the form of (15) making additional analytical calculations, resulting from the use of Dirichlet boundary conditions, i.e.:

$$C_0 = v_0, \tag{18}$$

$$C_1 = -\frac{3 \cdot v_0 + v_1 - 4 \cdot v_H}{\Delta s} \tag{19}$$

and

$$C_2 = \frac{2 \cdot v_0 + 2 \cdot v_1 - 4 \cdot v_H}{\Delta s^2}. \tag{20}$$

The proposed in this paper boundary conditions require the equality of the values of the approximated function (14) and their quadratic approximation (15) at the start point (v_0), half point (v_H) and end point (v_1) of the approximation distance Δs (Fig. 10). The velocity v_0 at the origin of the sector is already known (Fig. 7), but the value of v_H and value of v_1 are not known and should be determined here. At first the corresponding values of the relative times (ΔT_H) and (ΔT_1) must be determined. It can be easily done by a commonly known bisection method with the utilization of the motion equation (3) and the values of the interpolator state described by

$$\mathbf{I}^S = [s_0, v_0, a_0] \tag{21}$$

at the origin of analyzed sector. The move defined by the test trajectory is progressive (i.e. $v(t_L) \geq 0$) within considered range, so the bisection method with over a dozen simple iterations is sufficient to obtain satisfactory accuracy (step C8 in Fig. 9). In the bisection method the maximum value of ΔT_{max} for search algorithm is set to a minimal positive value of relative time, at which the velocity described by formula (2), reaches the value equal to zero. If the velocity does not reach the zero for any positive value of the relative time, then

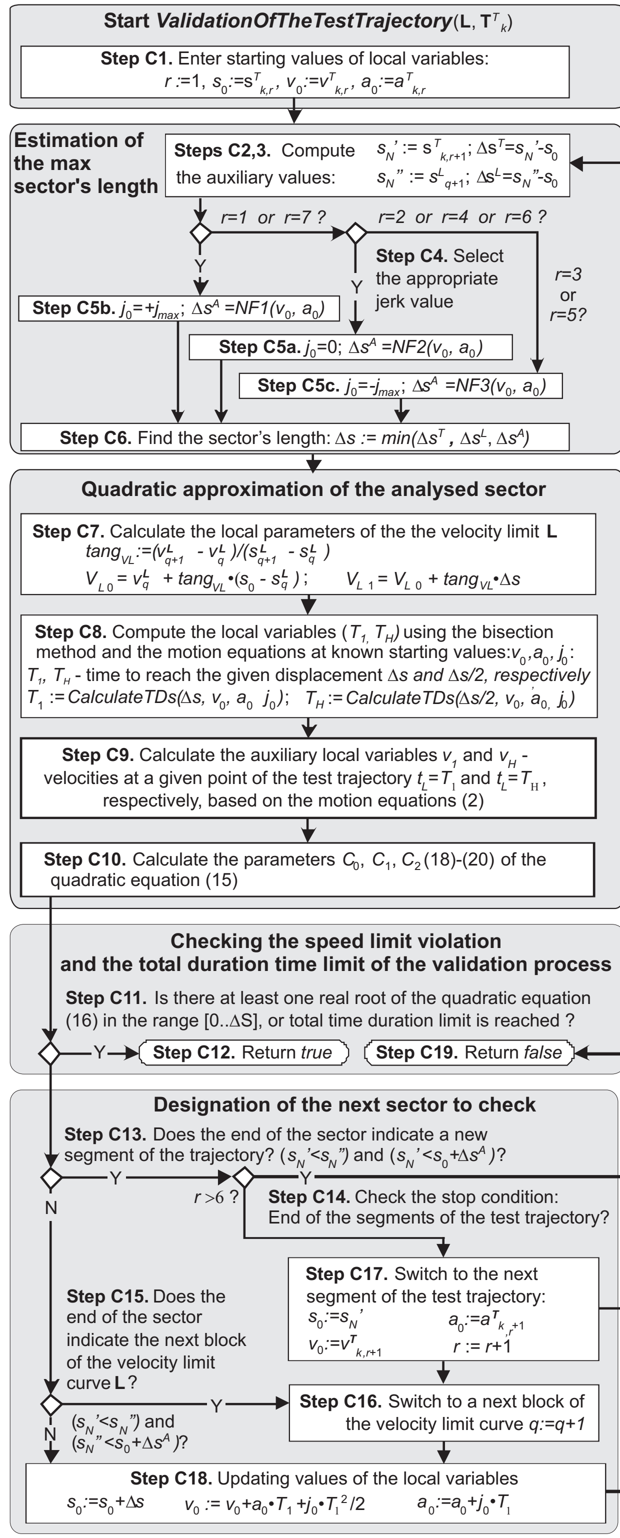

Fig. 9. Flowchart illustrating validation of the test trajectory.

value of ΔT_{max} is set to reasonable limit equal to one second. The minimal value for search algorithm is set to zero.

If the values of (ΔT_H) and (ΔT_1) are calculated, then the values of v_H and v_1 can be easily determined (step C9) on a basis of the motion equation (2). Finally at step C10 of the presented algorithm, parameters C_0, C_1 and C_2 of

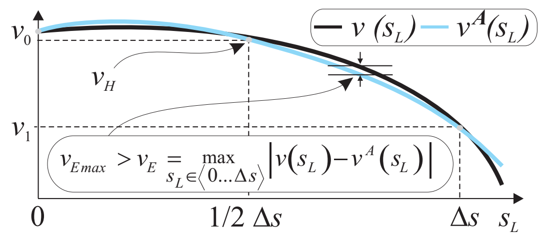

Fig. 10. Quadratic approximation of a velocity profile as a function of displacement.

the quadratic function (15) can be simply calculated using formulas (18)-(20). As a result we can use the quadratic inequality (16) instead of the complicated formula (13) which significantly simplifies the trajectory validation algorithm. In the next step (C11) we should solve the quadratic inequality (16) to check whether the analyzed sector of the test trajectory violates or not the speed limit. It can be easily done by checking if the adequate quadratic equation has at least one real root in the range $[0, \Delta_s]$. Because v_0 has always value less than v_{L0} (Fig. 7), the real root within a range $[0, \Delta_s]$ is the velocity violation point. As it was previously explained, in such a case validation of the test trajectory is completed (step C12) with a result equal to *true*. This procedure is also terminated with a result equal to *true* if the total duration time limit (T_G) of the validation procedure is reached. In another case the steps C13-C19 are performed to switch to the next sector in the iterative validation algorithm.

III. Flexible Takagi-Sugeno neuro-fuzzy system for the speed profile generation

A. Description of the system

The flexible Takagi-Sugeno neuro-fuzzy approximator is an important part of the proposed in the paper an original algorithm or the on-line speed profile generation. It allows to implement our approach in a typical real-time controller and to eliminate the lookup table method, which cannot be used because of the limited amount of the memory.

In the last decade different structures of neuro-fuzzy networks have been presented, often referred to in the world literature as neuro-fuzzy systems [6], [15], [17], [18]. As it was indicated in the Introduction, they combine the advantages of neural networks and classical fuzzy systems. In particular, the neuro-fuzzy networks are characterized - in contrast with neural networks - by a interpretable representation of knowledge represented by fuzzy rules. As generally known, the knowledge in neural networks is represented by the values of synaptic weights, and therefore is completely not interpretable, for instance, for a user of a medical expert system based on neural networks. Moreover, neuro-fuzzy networks can be trained, using the idea of error backpropagation method, which is the basis of learning of multilayer neural networks. The learning usually applies to membership function parameters of the IF and THEN parts of the fuzzy rules. It should be emphasized that neuro-fuzzy systems are universal approximators.

The advantages of neuro-fuzzy networks are the reason for their common application in classification, approximation and prediction problems. Most of neuro-fuzzy structures described in the world literature utilizes the Mamdani type inference or the Takagi-Sugeno schema. The Mamdani type inference consists in connecting the antecedents and the consequents of rules using a t-norm (most often the t-norm of the min type or of the product type). Then the aggregation of particular rules is made using a t-conorm. In case of the Takagi-Sugeno schema, the consequents of the rules are not fuzzy in nature, but are functions of the input variables. Less often the logical inference is applied, which consists in connecting the antecedents and the consequents of rules using a fuzzy implication that satisfies the conditions of definition of fuzzy implication. In case of an inference of logical type the aggregation of particular rules is made using a t-conorm [17], [18].

It is well know [18] that introducing additional parameters to be tuned in neuro fuzzy systems improves their performance and they are able to better represent the patterns encoded in the data. Therefore, in this paper, we incorporate flexibility concepts into the neuro-fuzzy system: certainty weights to the aggregation of rules and to the connectives of antecedents.

The flexible neuro-fuzzy system was used in our method because it is an excellent tool for solving approximation problems. However, alternatively other techniques (e.g. neural networks), can be incorporated into scheme depicted in Fig. 9, instead of flexible neuro-fuzzy systems. The novelty of our approach lies in developing the original algorithm for speed profile generation, by using the concept of the test trajectories, rather than in developing an approximator.

The algorithm in Fig. 9 uses the flexible neuro-fuzzy Takagi-Sugeno system. The neuro-fuzzy system determines the maximum value of the sector's length (steps C5a-C5c in Fig. 9), for which the quadratic approximation of the trajectory at k-th step is possible with an accuracy not worse than v_{Emax} (Fig. 10)). The neuro-fuzzy system determines the value of Δs^A (Fig. 12 and Fig. 8) for current values of v_0, a_0, and for given values of jerk ($j_0 = -j_{max}$, $j_0 = 0$, $j_0 = +j_{max}$).

We will apply the two-input and single-output flexible Takagi-Sugeno neuro-fuzzy system mapping $\mathbf{X} \to \mathbf{Y}$, where $\mathbf{X} \subset \mathbf{R}^2$ and $\mathbf{Y} \subset \mathbf{R}$. The rule base is given by

$$R^{(r)} : \left\{ \begin{array}{c} \text{IF } \bar{x}_1 \text{ is } A_1^r \left(w_{1,r}^\tau\right) \\ \text{AND } \bar{x}_2 \text{ is } A_2^r \left(w_{2,r}^\tau\right) \\ \text{THEN} \\ f^{(r)}\left(\bar{\mathbf{x}}\right) = c_{0,r}^f + \sum_{i=1}^{2} c_{i,r}^f \bar{x}_i \end{array} \right\} \left(w_r^{def}\right), \quad (22)$$

where $r = 1, 2, \ldots, N$. The construction of the system is based on the following parameters and weights:

- parameters of membership functions $\mu_{A_i^k}\left(\bar{x}_i\right)$, $i = 1, 2$, $r = 1, 2, \ldots, N$,
- parameters $c_{0,r}^f$, $c_{i,r}^f$, $i = 1, 2$, $r = 1, 2, \ldots, N$, in linear models describing consequences,
- certainty weights $w_{i,r}^\tau \in [0, 1]$, $i = 1, 2$, $r = 1, 2, \ldots, N$, describing importance of antecedents in the rules,
- certainty weights $w_r^{def} \in R$, $r = 1, 2, \ldots, N$, describing importance of the rules.

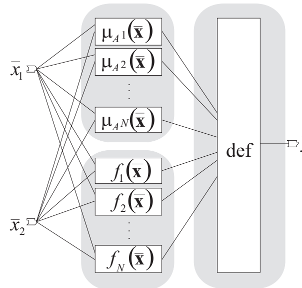

Fig. 11. Flexible Takagi-Sugeno neuro-fuzzy system used for the validation of the test trajectory.

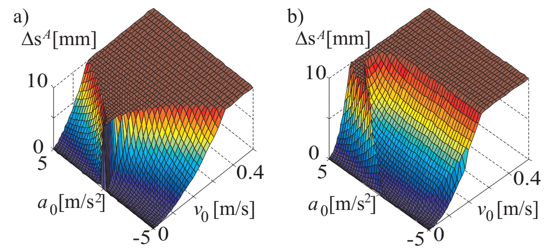

Fig. 12. Graphical representation of Δs^A obtained from simulations as a function of velocity and acceleration at given values of the jerk: a)$j_0 = 0$, b)$j_0 = 250$.

The aggregation in the Takagi-Sugeno model, described by the rule base (22), is in the form

$$\bar{y} = f(\bar{\mathbf{x}}) = \frac{\sum_{r=1}^{N} w_r^{def} \cdot f^{(r)}(\bar{\mathbf{x}}) \cdot \mu_{A^r}(\bar{\mathbf{x}})}{\sum_{r=1}^{N} \mu_{A^r}(\bar{\mathbf{x}})}, \tag{23}$$

where

$$\mu_{A^r}(\bar{\mathbf{x}}) = T^* \left\{ \mu_{A_1^r}(\bar{x}_1), \mu_{A_2^r}(\bar{x}_2); w_{1,r}^\tau, w_{2,r}^\tau \right\} \tag{24}$$

and T^* is a weighted t-norm [17], [18]. Weighted t-norm in the two-dimensional case is defined as follows

$$\begin{array}{c} T^* \left\{ \begin{array}{c} \mu_{A_1^r}(\bar{x}_1), \mu_{A_2^r}(\bar{x}_2); \\ w_{1,r}^\tau, w_{2,r}^\tau \end{array} \right\} = \\ T \left\{ \begin{array}{c} 1 + \left(\mu_{A_1^r}(\bar{x}_1) - 1\right) \cdot w_{1,r}^\tau, \\ 1 + \left(\mu_{A_2^r}(\bar{x}_2) - 1\right) \cdot w_{2,r}^\tau \end{array} \right\} \end{array}. \tag{25}$$

The weights $w_{1,r}^\tau$ and $w_{2,r}^\tau$ are certainties (credibilities) of both antecedents in (25). Observe that:

- If $w_{1,r}^\tau = w_{2,r}^\tau = 1$ then the weighted t-norm is reduced to the standard t-norm.
- If $w_{1,r}^\tau = 0$ then $T^* \left\{ \mu_{A_1^r}(\bar{x}_1), \mu_{A_2^r}(\bar{x}_2); 0, w_{2,r}^\tau \right\} = 1 + \left(\mu_{A_2^r}(\bar{x}_2) - 1\right) \cdot w_{2,r}^\tau$.

The general architecture of the flexible Takagi-Sugeno system is depicted in Fig. 11. As we can see, it is a multilayer network structure. To train it, the idea of the error backpropagation method may be applied [17], [18]. Let us define the learning sequence as $(\bar{\mathbf{x}}_1, \mathbf{d}_1), (\bar{\mathbf{x}}_2, \mathbf{d}_2), \ldots, (\bar{\mathbf{x}}_Z, \mathbf{d}_Z)$, where $\bar{\mathbf{x}}_z = [v_{0z}, a_{0z}]$, $\mathbf{d}_z = \left[\Delta s^A{}_z\right]$, $z = 1, 2, \ldots, Z$. Based on the learning sequence we determine all parameters and weights of fuzzy system (23).

B. Experimental results

The neuro-fuzzy structure (23) aids the validation algorithm used in the trajectory generation system. This system is used as an approximator of the highly nonlinear dependency (17). Because the jerk can have only three discrete values (Fig. 3.a), we can use three simple neuro-fuzzy system for these three separate cases instead of a complex one. In that case each of the three neuro-fuzzy systems approximate a highly nonlinear function (Fig. 12) for different values of the jerk (steps C5a, C5b and C5c in Fig. 9). We use neuro-fuzzy system NFS1 given by

$$\Delta s^A = NFS1(v_0, a_0), \tag{26}$$

when we analyze segments $r = 1, 7$, neuro-fuzzy system NFS2 given by

$$\Delta s^A = NFS2(v_0, a_0) \tag{27}$$

for segments $r = 2, 4, 6$ or neuro-fuzzy system NFS3 given by

$$\Delta s^A = NFS3(v_0, a_0), \tag{28}$$

for segments r=3, 5.

A minor disadvantage of such a simplification is the necessity to declare the value of j_{max} at the stage of designing the control system, in principle before training the neuro-fuzzy system. This is not a big drawback because the value of j_{max} is never changed during the entire use of the machine. The selection and fixing the value of the j_{max} as well as the initial tuning phase of the used neuro-fuzzy system must be done only once, at the stage of designing the system. It is possible to prepare several flexible neuro-fuzzy systems, each learned in advance (for different values of the j_{max} typically used in practice) and use them later without modification. The significant advantage of such an approach is the simplification of the neuro-fuzzy system and the whole algorithm is more efficient in a real time implementation.

Three independent flexible neuro-fuzzy Takagi-Sugeno systems were prepared for the verification of the test trajectory (steps C5a-C5c in Fig. 9). Each of these systems (26)-(28) determines the output Δs^A for another jerk j_0 from the set $\{-j_{max}, 0, +j_{max}\}$. Training data, which were used in the learning process, were generated by trial and error method. The idea of generate the training data was based on the assumption, that the outputs Δs^A, should have greater values, what results in decreasing number of steps validation algorithm of a test trajectory. Obviously, the condition $V_E < V_{Emax}$ should be satisfied (Fig. 10).

We used neuro-fuzzy systems given by (23) characterized by the Gaussian fuzzy sets, 30 rules ($N = 30$) and algebraic t-norms. We employed the Fuzzy C-Mans algorithm to find initial values of membership functions parameters ($m = 2.0$, 1000 steps) [15], [17], [18]. We also initialized weights of antecedents $w_{i,r}^\tau = 1$, $i = 1, 2$, $r = 1, 2 \ldots, N$, and weights of the rules $w_r^{agr} = 1$, $r = 1, 2 \ldots, N$. The learning data length for each of three neuro-fuzzy systems (26)-(28) was

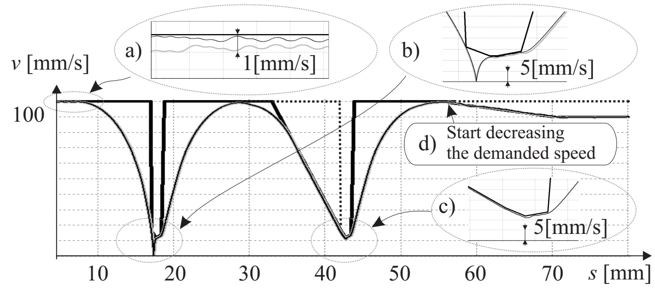

Fig. 13. The velocity limit (thick black), the smooth trajectory obtained with the three neuro-fuzzy systems (thin black), and the smooth trajectory obtained with the trial and errors method (gray).

$Z = 1682$. The system was learned by the backpropagation method ($\mu = 0.15$) with momentum ($\lambda = 0.10$) by 100000 epochs. The final average root mean square error (RMSE) was equal 0.0996 for three used neuro-fuzzy systems. Our approach allows to easily implement the presented algorithm in a microprocessor system used in the CNC machine.

The final trajectory obtained with the help of the three neuro-fuzzy systems (26)-(28) is shown in Fig. 13. The comparison with the trajectory obtained by the trial and errors method shows that there are some insignificant differences between them, resulting from the neuro-fuzzy systems approximation errors. Despite the slight differences between these two cases, the "neuro-fuzzy based" trajectory fully guarantees the required limits of jerk, acceleration and velocity, and in result it is suitable to use in the CNC system.

In Fig. 13 some areas are enlarged to better illustrate the specific features of the presented algorithm. In the first indicated area (Fig. 13.a) the small velocity fluctuations are visible. It results from the fact, that the final trajectory is formed by a merger of short fragments of the successive test trajectories. Generally this is a drawback of the presented algorithm. However, if the amplitudes of these fluctuations are small, then it does not influence negatively on the quality of the work. Their amplitude is proportional to the length of the connecting pieces, which depends in turn on the time period T_G used to generate and validate the subsequent test trajectories. Decreasing this time period causes the reduction of the amplitude, but it requires more computational power of the computer system. In our work we used T_G with experimentally chosen value equal to 2 milliseconds.

The next enlarged fragment (Fig. 13.b) shows that unfavorable slowdown occurs if the speed limit curve drops sharply. This drawback results from the lack of the global velocity optimization techniques. However, preprocessing of the velocity limit curve, i.e. eliminating the sharp drops, could be used to prevent that adverse slowdown (Fig. 13.c).

Despite of these minor drawbacks, a great advantage of our algorithm is that it is able to adjust the generated speed profile to the changing external conditions, e.g. spindle load change, in an efficient manner. As it was indicated in the Introduction, in our approach it is possible to modify the demanded value of the feed rate of the tool during machine operation. This is illustrated in simulation presented in Fig. 13.d in which the speed limit is decreased in order to protect the spindle from the overload. As we can see, the algorithm is able to on-line modify the generated speed profile.

IV. Conclusions

In this paper we presented a new algorithm for the on-line speed profile generation for industrial machine tool. The unique feature of our method is the ability to quickly adjust the generated trajectory to changing speed limits. It is possible to modify the requested value of the feed rate of the tool during machine operation. This feature is very important for operating CNC machines because of the need to protect the cutter from the brake and spindle from the overload in high speed machining. Our method, based on the neuro-fuzzy approach, allows the system to work properly and quickly, and to construct the trajectory generator operating on-line. It should be noted that neuro-fuzzy structures can be adopted for realization in hardware, e.g. in the CMOS technology [31].

Acknowledgment

This paper was prepared under project operated within the Foundation for Polish Science Team Programme co-financed by the EU European Regional Development Fund, Operational Program Innovative Economy 2007-2013, Polish-Singapore Research Project 2008-2011 and also supported by National Science Center NCN.

The authors would like to thank the reviewers for very helpful suggestions and comments in the revision process.

References

[1] M.-T. Lin, M.-S. Tsai, H.-T. Yau, Development of an dynamics-based NURBS interpolator with real-time look-ahead algorithm, *International Journal of Machine Tools & Manufacture*, vol. 47, pp.2246-2262, 2007.

[2] J.-Y. Dieulot, I. Thimoumi, F. Colas, R. Bare, Numerical aspects and performances of trajectory planning methods of flexible axes, *International Journal of Computers, Communications & Control*, vol. I, no. 4, pp. 35-44, 2006.

[3] W. T. Lei, M. P. Sung, L. Y. Lin, J. J. Huang, Fast real-time NURBS path interpolation for CNC machine tools, *International Journal of Machine Tools & Manufacture*, vol. 47, pp.1530-1541, 2007.

[4] Fuchun Sun, Li Li, Han-Xiong Li, Huaping Liu, Neuro-Fuzzy Dynamic-Inversion-Based Adaptive Control for Robotic Manipulators-Discrete Time Case, *IEEE Transactions on Industrial Electronics*, vol. 54, no. 3, 2007, pp. 1342-1351.

[5] Y. Sun, J. Wang, D. Guo, Guide curve based interpolation scheme of parametric curves for precision CNC machining, Parametric NURBS Curve Interpolators: A Review, *International Journal of Machine Tools & Manufacture*, vol. 46, pp. 235-242, 2006.

[6] T. Takagi and M. Sugeno. Fuzzy identification of systems and its application to modeling and control, *IEEE Trans. Syst., Man, Cybern.*, vol. 15, pp. 116 132, 1985.

[7] S. Mohan, S.-H. Kweon1, D.-M. Lee1 and S.-H. Yang, Parametric NURBS curve interpolators: A Review, *International Journal of Precision Engineering and Manufacturing*, vol. 9, no. 2, pp. 84-92, 2008.

[8] H.-T. Yau, J.-B. Wang, Fast Bezier interpolator with real-time lookahead function for high-accuracy machining, *International Journal of Machine Tools & Manufacture*,vol. 47, pp. 1518-1529, 2007.

[9] H.-T. Yau, J.-B. Wang, C.-Y. Hsu and C.-H- Yeh, PC-based controller with real-time look-ahead NURBS interpolator, *Computer-Aided Design & Applications*, vol. 4, 1-4, pp 331-340, 2007.

[10] M.-S. Tsai, H.-W. Nien, H.-T. Yau, Development of an integrated look-ahead dynamics-based NURBS interpolator for high precision machinery, *Computer-Aided Design*, vol. 40, pp. 554-566, 2008.

[11] Sheng-Jung Tseng, Kuan-Yuan Lin, Jiing-Yih Lai, and Wen-Der Ueng, A NURBS curve interpolator with jerk-limited trajectory planning, *Journal of the Chinese Institute of Engineers*, vol. 32, no. 2, pp. 215-228, 2009.

[12] R. A. Osario-Rios, R. d. J. Romero-Trosncoso, G. Herera-Ruiz, R. Casteneda-Miranda, FPGA implementation of higher degree polynomial acceleration profiles for peak jerk reduction in servomotors, *Robotics and Computer-Integrated Manufacturing*, vol. 25, pp. 379-392, 2009.
[13] C. G. Lo Bianco, R. Zanasi, Smooth profile generation for a tile printing machine, *IEEE Transactions on Industrial Electronics*, vol. 50, No. 3, pp. 471-477, 2003.
[14] Ch. Zheng, Y. Su, P. C. Müller, Simple on-line smooth trajectory generation for industrial systems, *Mechatronics*, vol. 19, pp.571-576, 2009.
[15] K. Cpałka, A New Method for design and reduction of neuro-fuzzy classification systems, *IEEE Transactions on Neural Networks*, vol. 20, No. 4, 2009, pp. 701-714.
[16] L. Rutkowski, *Flexible Neuro-Fuzzy Systems*. Kluwer Academic Publishers, 2004.
[17] L. Rutkowski and K. Cpałka, Designing and learning of adjustable quasi triangular norms with applications to neuro-fuzzy systems, *IEEE Trans. Fuzzy Syst.*, vol. 13, no. 1, pp. 140-151, 2005.
[18] L. Rutkowski and K. Cpałka, Flexible neuro-fuzzy systems, *IEEE Trans. Neural Networks*, vol. 14, no. 3, pp. 554-574, 2003.
[19] B.K. Bose, N.R. Patel, K. Rajashekara, A neuro-fuzzy-based on-line efficiency optimization control of a stator flux-oriented direct vector-controlled induction motor drive, *IEEE Transactions on Industrial Electronics*, vol. 44, no. 2, 1997, pp. 270-273.
[20] A. Chatterjee, R. Chatterjee, F. Matsuno, T. Endo, Augmented Stable Fuzzy Control for Flexible Robotic Arm Using LMI Approach and Neuro-Fuzzy State Space Modeling, *IEEE Transactions on Industrial Electronics*, vol. 55, no. 3, 2008, pp. 1256-1270.
[21] D. Fuessel, R. Isermann, Hierarchical motor diagnosis utilizing structural knowledge and a self-learning neuro-fuzzy scheme, *IEEE Transactions on Industrial Electronics*, vol. 47, no. 5, 2000, pp. 1070-1077.
[22] P.Z. Grabowski, M.P. Kazmierkowski, B.K. Bose, F. Blaabjerg, A simple direct-torque neuro-fuzzy control of PWM-inverter-fed induction motor drive, *IEEE Transactions on Industrial Electronics*, vol. 47, no. 4, 2000, pp. 863-870.
[23] P. Melin, O. Castillo, Intelligent control of complex electrochemical systems with a neuro-fuzzy-genetic approach, *IEEE Transactions on Industrial Electronics*, vol. 48, no. 5, 2001, pp. 951-955.
[24] T. Orlowska-Kowalska, M. Dybkowski, K. Szabat, Adaptive Sliding-Mode Neuro-Fuzzy Control of the Two-Mass Induction Motor Drive Without Mechanical Sensors, *IEEE Transactions on Industrial Electronics*, vol. 57, no. 2, 2010, pp. 553-564.
[25] T. Orlowska-Kowalska, K. Szabat, Control of the Drive System With Stiff and Elastic Couplings Using Adaptive Neuro-Fuzzy Approach, *IEEE Transactions on Industrial Electronics*, vol. 54, no. 1, 2007, pp. 228-240.
[26] C.K. Kwong, K.Y. Chan, H. Wong, Takagi-Sugeno neural fuzzy modeling approach to fluid dispensing for electronic packaging, *Expert Systems with Applications*, vol. 34, issue 3, 2008, pp. 2111-2119.
[27] K. Erkorkmaz, Y. Altintas, High speed CNC system design. Part I: jerk limited trajectory generation and quintic spline interpolation, *International Journal of Machine Tools & Manufacture*, vol. 41, pp.1323-1345, 2001.
[28] N.Y. Ki, A new velocity profile generation for high efficiency CNC machining application, *City University of Hong Kong*, Master Thesis, pp.1-83, 2008.
[29] S. Macfarlane, On-Line smooth trajectory planning for manipulators, *The University of British Columbia*, Master Thesis, pp.1-146, 2001.
[30] D. W. Pessen, *Industrial automation. Circuit Design and Components*, John Willey & Sons, Inc., pp.478-483, 1989.
[31] B.M. Wilamowski, R.C. Jaeger, M.O. Kaynak, Neuro-fuzzy architecture for CMOS implementation, *IEEE Transactions on Industrial Electronics*, vol. 46, no. 6, pp. 1132-1136, 1999.

Leszek Rutkowski (F'05) received the M.Sc. and Ph.D. degrees in 1977 and 1980, respectively, both from the Technical University of Wroclaw, Poland. Since 1980, he has been with the Technical University of Czestochowa where he is currently a Professor and Chairman of the Computer Engineering Department. From 1987 to 1990 he held a visiting position in the School of Electrical and Computer Engineering at Oklahoma State University. His research interests include neural networks, fuzzy systems, computational intelligence, pattern classification and expert systems. In May and July 2004 he presented in the IEEE Transaction on Neural Networks a new class of probabilistic neural networks and generalized regression neural networks working in a time-varying environment. He published over 170 technical papers including 20 in various series of IEEE Transactions. He is the author of the books Computational Intelligence published by Springer (2008), New Soft Computing Techniques For System Modeling, Pattern Classification and Image Processing published by Springer (2004), Flexible Neuro-Fuzzy Systems published by Kluwer Academic Publishers (2004), Methods and Techniques of Artificial Intelligence (2005, in Polish), Adaptive Filters and Adaptive Signal Processing (1994, in Polish), and co-author of two others (1997 and 2000, in Polish) Neural Networks, Genetic Algorithms and Fuzzy Systems and Neural Networks for Image Compression. Dr. Leszek Rutkowski is President and Founder of the Polish Neural Networks Society. He organized and served as a General Chair of the International Conferences on Artificial Intelligence and Soft Computing held in 1996, 1997, 1999, 2000, 2002, 2004, 2006, 2008 and 2010. Dr. Leszek Rutkowski is past Associate Editor of the IEEE Transactions on Neural Networks (1998-2005) and IEEE Systems Journal (2007-2010). He is Editor-in-Chief of Journal of Artificial Intelligence and Soft Computing Research and he is on the editorial board of the International Journal of Applied Mathematics and Computer Science (1996-present) and International Journal of Biometric (2008-present). Dr. Leszek Rutkowski was awarded by the IEEE Fellow Membership Grade for contributions to neurocomputing and flexible fuzzy systems. He is a recipient of the IEEE Transactions on Neural Networks 2005 Outstanding Paper Award. Dr. Leszek Rutkowski served in the IEEE Computational Intelligence Society as the Chair of the Distinguished Lecturer Program (2008-2009) and the Chair of the Standards Committee. He is the Founding Chair of the Polish Chapter of the IEEE Computational Intelligence Society which won 2008 Outstanding Chapter Award. In 2004 he was elected as a member of the Polish Academy of Sciences.

Andrzej Przybył received his Ph.D. in automatics and robotics from the Poznan University of Technology in 2003. He is an assistant professor in the Department of Computer Engineering at Częstochowa University of Technology. He is working on developing new control methods used in mechatronics systems. His research interests center around motion control systems, real-time Ethernet, FPGA devices and soft computing algorithms for electrical drives. Dr. Andrzej Przybył designed various microprocessors, digital signal processors and FPGA based embedded systems. He has published about 20 technical papers.

Krzysztof Cpałka (M'11) was born in Częstochowa, Poland, in 1972. He received the M.Sc. degree in electrical engineering in 1997 and the Ph.D. degree (Honors) in 2002 in computer engineering, both from the Częstochowa University of Technology, Poland.

Since 2010, he has been an associate professor in the Department of Computer Engineering at Częstochowa University of Technology. His research interests include fuzzy systems, neural networks, evolutionary algorithms and artificial intelligence. He published over 60 technical papers in journals and conference proceedings. Krzysztof Cpałka is a recipient of the 2005 IEEE Transactions on Neural Networks Outstanding Paper Award.