A Novel Robot Trajectory Planning Algorithm Based on NURBS Velocity Adaptive Interpolation

Qingxiao Zou, Weidong Guo and Fouaz Younès Hamimid

Abstract This paper proposed a novel trajectory planning algorithm for robot manipulator using Non-Uniform Rational B-Spline (NURBS) curve interpolation. Total trajectory planning process was divided into two stages: curve fitting stage (using NURBS) and trajectory interpolating stage. Velocity adaptive control methodology was introduced to adjust the steps of robot manipulator interpolation and further to reduce the chord error. Chord error and interpolation velocity of robot end-effector were monitored real-time and the consecutive interpolation point was pre-calculated to make a judgement on whether the chord error meet requirements, otherwise the interpolation velocity would be lowered to guarantee the accuracy was controlled under the error range. Finally, the author employed 4-DOF (Degree of Freedom) free floating space robot to verify the availability of the designed algorithm depending on several given demonstrating waypoints. The experimental result showed that the proposed NURBS trajectory planning algorithm was not only feasible for high-accuracy trajectory planning but also yielded a smooth trajectory with satisfactory kinematic properties for all robot joints.

Keywords Robot trajectory planning · Interpolation algorithm · NURBS curve
Velocity adaptive control

Q. Zou · W. Guo (✉) · F. Y. Hamimid
School of Mechanical Engineering and Automation, Beihang University,
Beijing 100191, China
e-mail: guowd@buaa.edu.cn

Q. Zou
e-mail: zouqingxiao@buaa.edu.cn

F. Y. Hamimid
e-mail: f.ou.ez@hotmail.com

J. Tan et al. (eds.), *Advances in Mechanical Design*,
Mechanisms and Machine Science 55, https://doi.org/10.1007/978-981-10-6553-8_78

1 Introduction

Over the past decades, industrial robots have been widely used in many fields, like aircraft industry and automobile manufacturing industry. Typical applications of industrial robots include welding, painting, assembly and material handling, etc. These kinds of tasks require high accuracy in positioning and trajectory planning. Trajectory planning is one of the most important issues of robotics. The main task of path planning for manipulator is to achieve an optimal trajectory with good dynamic and kinematic properties. It is worth mentioning that a proper trajectory can greatly reduce motion time and minimize energy consumption of robot. Robot trajectory planning has developed toward high-speed moving and high-precision controlling with the sophisticated application of interpolation technology in CNC machining system recently [1]. Various interpolation algorithms have been employed to control the robot motion trajectory [2]. Traditional interpolation of free curve usually depends on linear or circular interpolation, but there are many defects with this process [3]. In order to improve interpolation accuracy, a large number of small linear and circular segment were generated, which posed great challenge to data transmission and real-time communication of robot controlling system. Researchers have tried different kinds of curves for interpolation to overcome this problem.

Bézier curves were first developed for car surfaces design applications [4]. Once the control points are given, Bézier curve will be formed uniquely by the Bernstein basic functions and cannot be modified any more [5]. The shape of fitting curve varied with the change of control points, which hindered its application in trajectory planning to some extent. The knot vector of B-Spline curves partially overcome this deficiency. Moreover, B-Spline curves share many important properties that were particular suitable for trajectory planning [6, 7] and they are also fundamental to data fitting and approximation. B-Spline act as a special case of NURBS, the weighs of NURBS could be used to modify curve shape and thus NURBS has gained more application than B-Splines in engineering. Therefore, the concept of NURBS interpolation and its employability have been the focus of considerable research internationally [8].

Huo and Poo [9] have developed an algorithm to estimate the contour error of NURBS interpolation. In his algorithm, series of reference points adjacent to current interpolation position were used to reconstruct the NURBS curve. Contour error estimation was depended on the minimum distance between NURBS curve and present location. Park et al. [7] presented two-stages interpolation approach that used to compensated for interpolation errors. The interpolation result was confined by velocity, acceleration or jerk equation. Ahmad et al. [10] pointed out the constant federate and chord errors accuracy between two adjacent interpolation points along NURBS curves was difficult to achieve owing to the non-uniform mapping, and he has worked out a solutions of intelligent NURBS interpolation for CNC machining systems. Parametric interpolation based on NURBS has been extensively applied in CNC system but rarely used in robot trajectory planning.

Traditional robot trajectory planning algorithms mainly rely on linear or circular interpolation, which involved time-consuming and low accuracy curve and was not suitable for real-time planning. In this case, the study proposed a novel high-speed trajectory planning algorithm adopting velocity adaptive control strategy to improve the path planning accuracy, which is also the novelty and contribution of this paper.

Generally, the trajectory planning algorithm was divided into two stages: curve fitting stage and trajectory interpolating stage. In the first stage, a smooth curve passing through the sample points in sequence was constructed using NURBS curve. Then large amounts of refining interpolated points would be generated based on the first stage curve according to interpolation cycle. Chord error and interpolation velocity were monitored real-time and the consecutive interpolation point chord error was pre-calculated. The interpolation movement would continue if chord error met the accuracy requirements, otherwise velocity of end-effector (namely interpolated velocity) would be lowered to guarantee the accuracy was controlled under the error range. Then the author employed 4-DOF free floating space robot to verify the availability of the algorithm.

Trajectory planning tasks of robot manipulator mainly includes the following three aspects:

(1) Description of robot trajectory.

Operators just need to give simple description of manipulator movement rather than concrete trajectory mathematical function expression. For example, the initial and final pose (position and orientation) of end-effector were given, namely point to point control (PTP) [11], or several demonstrating points were known, the end-effector was required to travel through them by sequence, called continuous path control (CP) [12].

(2) Choice of interpolation algorithm.

Select proper interpolation algorithm according to above robot trajectory description. Other requirements such as the constraints of maximum velocity and acceleration or movement accuracy should also be satisfied. Then a series of interpolation points would be generated by the interpolation algorithm, and the real-time data was returned to the robotic control system.

(3) Inverse kinematic solution of robot joints.

The inverse kinematics solution of manipulator is realized by applying the inverse transformation methodology according to interpolation cycle [13]. Interpolation points will be transformed from working space to joint space coordinate system, so the dynamic properties of each joint can be determined. The author has employed D-H (Denavit-Hartenberg) [14] method to make a kinematics analysis of robot during this process.

2 NURBS and Curve Interpolation

This part was organized as follows. The definition of NURBS was shown in Sect. 2.1. Computational process relevant to curve fitting based on the sample data was presented in Sect. 2.2, accumulating chord length parametrization methodology [15] was adopted to construct the knot vectors according to the demonstrating points. All control points would be obtained by using control points matrix with weighs. After curve fitting stage, a NURBS trajectory curve was obtained. Section 2.3 gave a brief analysis of trajectory interpolation using the acquired NURBS fitting curve.

2.1 *Definition of NURBS*

A NURBS curve is defined by four components: knot vector, curve degree, control points and weight [16].

$$p(u) = \frac{\sum_{i=0}^{n} d_i \omega_i N_{i,k}(u)}{\sum_{i=0}^{n} \omega_i \omega_i N_{i,k}(u)} \tag{1}$$

where $d_i(i = 0, 1, \ldots, \mathrm{n})$ is control point, ω_i is the corresponding weight of *i*th control point and k is the degree of NURBS curve, the first and last weight factors $\omega_0, \omega_n > 0$, and others $\omega_i \geq 0$, adjacent k weight factors cannot be zero simultaneous in case of the denominator be equal to zero. $N_{i,k}(u)$ is the basic function of B-spline defined on knot vector $\mathrm{U} = [u_0, u_1, \ldots, u_i, \ldots, u_{n+k+1}]$, recursively defined by de Boor algorithm:

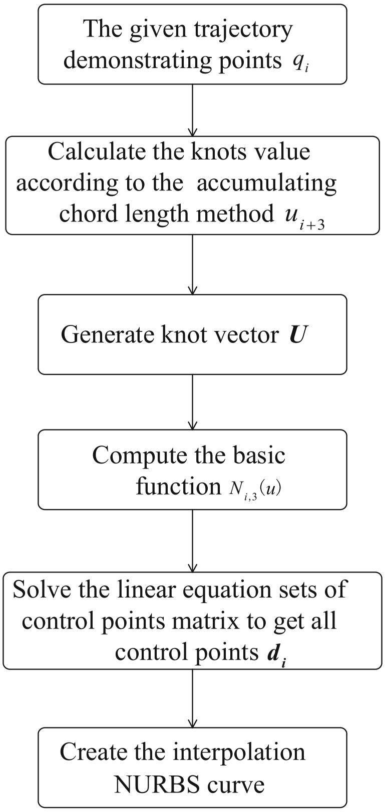
$$N_{i,0}(u) = \begin{cases} 0, & \mathit{if} \quad u_i \leq u \leq u_{i+1} \\ 1, & \quad \mathit{otherwise} \end{cases} \tag{2}$$

where

$$N_{i,k}(u) = \frac{u - u_i}{u_{i+k} - u_i} N_{i,k-1}(u) + \frac{u_{i+k+1} - u}{u_{i+k+1} - u_{i+1}} N_{i+1,k-1}(u) \tag{3}$$

In this paper, the author adopted three degrees NURBS to fit the demonstrating trajectory points. Assuming that the given demonstrating points were $\{q_0, q_1, \ldots, q_{n-2}\}$. The procedures of generating a motion trajectory of three degrees NURBS curve $p(u)$ was shown in Fig. 1.

Fig. 1 Flow chart of NURBS curve fitting


The given trajectory demonstrating points q_i
Calculate the knots value according to the accumulating chord length method u_{i+3}
Generate knot vector $\boldsymbol{U}$
Compute the basic function $N_{i,3}(u)$
Solve the linear equation sets of control points matrix to get all control points $\boldsymbol{d}_i$
Create the interpolation NURBS curve

2.2 *Calculation of Control Points*

In practical application, the non-periodic knot vector usually follows the form [17]:

$$\mathrm{U} = [\alpha, \ldots \alpha, \mathrm{u}_{\mathrm{k}+1}, \ldots, \mathrm{u}_{\mathrm{n}}, \beta, \ldots \beta] \tag{4}$$

where α and β appear $(k + 1)$ times at the endpoint. The special arrangement guarantees the NURBS curve would pass through the initial and final endpoint, which coincides with the initial and final demonstrating trajectory point. In this paper, it is assumed that the knot values lie in $u \in [0, 1]$, where $(n + 1)$ represents the number of control points, and the number of demonstrating point is $(n - 2)$.

According to NURBS Convex Hull property [18], the interpolation polynomial in knot span $[u_i, u_{i+1}]$ is merely under the control of points $d_{i-k}, d_{i-k+1}, , \ldots, d_i$. In this paper, the author selected $(n - 1)$ demonstrating points on the trajectory, so the NURBS curve ***p(u)*** of degree k was defined by $(n + 1)$ control points and a knot vector $\mathrm{U} = [u_0, u_1, \ldots, u_{n+4}]$ with the first and last four knots clamped.

$$\begin{cases} u_0 = u_1 = u_2 = u_3 = 0, \\ u_{i+3} = u_{i+2} + \Delta p_i, \quad i = 1, 2, \ldots, n-2 \\ u_{n+1} = u_{n+2} = u_{n+3} = u_{n+4} = 1 \end{cases} \tag{5}$$

where

$$\Delta p_i = \frac{p_i - p_{i-1}}{s} \tag{6}$$

where Δp_i is forward differential vector [19], s is accumulating chord length. The relationship between demonstrating point q_i and control point d_i is shown as follows:

$$q_i = \sum_{j=i}^{i+2} d_j N_{j,3}(u_{i+3}), \quad i = 0, 1, \ldots, n-1 \tag{7}$$

So, the control points can be obtained by matrix equation according to the equation with two additional boundary conditions (7), the calculation matrix is shown:

$$\begin{bmatrix} N_{0,3}(u_3) & N_{1,3}(u_3) & \cdots & 0 & 0 \\ 0 & N_{1,3}(u_4) & & 0 & 0 \\ \vdots & & \ddots & & \vdots \\ 0 & 0 & \cdots & N_{n,3}(u_{n+1}) & 0 \\ 0 & 0 & & N_{n,3}(u_{n+2}) & N_{n+1,3}(u_{n+2}) \end{bmatrix} \begin{bmatrix} d_0 \\ d_1 \\ \vdots \\ d_n \\ d_{n+1} \end{bmatrix} = \begin{bmatrix} q_0 \\ q_1 \\ \vdots \\ q_{n-2} \\ q_{n-1} \end{bmatrix} \tag{8}$$

Any coordinates of points on NURBS curve could be further obtained by Eq. (9):

$$\begin{cases} x(u) = \dfrac{\sum_{j=i-3}^{i} \omega_j x_j N_{j,3}(u)}{\sum_{j=i-3}^{i} \omega_j N_{j,3}(u)} \\ y(u) = \dfrac{\sum_{j=i-3}^{i} \omega_j y_j N_{j,3}(u)}{\sum_{j=i-3}^{i} \omega_j N_{j,3}(u)} \\ z(u) = \dfrac{\sum_{j=i-3}^{i} \omega_j z_j N_{j,3}(u)}{\sum_{j=i-3}^{i} \omega_j N_{j,3}(u)} \end{cases} \tag{9}$$

2.3 Analysis of Trajectory Interpolation

A NURBS curve $p(u)$ could be obtained based on the above calculating procedures. In order to interpolate the trajectory of robot end-effector, the first step was to determine the curve interpolation parameter u_i. Taylor expansion method was applied in interpolation algorithm. The trajectory interpolation steps and corresponding position of end-effector in working space coordinate system could be calculated [20], interpolation increment Δu should meet the requirement of manipulator velocity in each cycle. Δu is calculated by following equation:

$$u_{i+1} = u_i + T \cdot \dot{u}_i + \left(\frac{T^2}{2}\right) \cdot \ddot{u}_i + o\left(T^2\right) \tag{10}$$

where T is interpolation cycle, $T = t_{i+1} - t_i$.

Assuming that:

$$x = P_x(u), \quad y = P_y(u), \quad z = P_z(u) \tag{11}$$

So,

$$\frac{du}{dt} = \frac{V(t)}{dP/du} = \frac{V(t)}{\sqrt{\dot{x}_i^2 + \dot{y}_i^2 + \dot{z}_i^2}} \tag{12}$$

$$\frac{d^2u}{dt^2} = \frac{dV(t)/dt}{\sqrt{\dot{x}_i^2 + \dot{y}_i^2 + \dot{z}_i^2}} - \frac{V^2(t)(\dot{x} \cdot \ddot{x} + \dot{y} \cdot \ddot{y} + \dot{z} \cdot \ddot{z})}{(\dot{x}_i^2 + \dot{y}_i^2 + \dot{z}_i^2)^2} \tag{13}$$

$$\Delta u_i = \frac{\Delta L_i}{\sqrt{\dot{x}_i^2 + \dot{y}_i^2 + \dot{z}_i^2}} - \frac{(\Delta L_i)^2}{2} \cdot \frac{(\dot{x} \cdot \ddot{x} + \dot{y} \cdot \ddot{y} + \dot{z} \cdot \ddot{z})}{(\dot{x}_i^2 + \dot{y}_i^2 + \dot{z}_i^2)^2} \tag{14}$$

where $V(t)$ was the velocity of robot end-effector.

$$\Delta L_i = V(t_i) \cdot T \tag{15}$$

3 Velocity Adaptive Interpolation Algorithm

This section first explained the definition of chord error and described the overall structure of trajectory planning scheme. Then velocity adaptive interpolation algorithm was discussed to make a balance between trajectory interpolation velocity and interpolated accuracy.

3.1 Chord Error

In order to improve the accuracy of interpolation, chord error should be controlled under the range. No radial error appears as all interpolation points locate on NURBS curve [20]. Trajectory planning errors mainly come from chord error by interpolation step size as a unit. There are two different methods to calculate the chord error [21], δ_{i1}, δ_{i2} is shown in Fig. 2. The first method uses curvature of the point $p(u_i)$ to calculate the chord error, which is relevant to interpolation step ΔL_i and curvature ρ_i. The mathematical equation between them is Eq. (16).

$$\Delta L_i = \sqrt{4\delta_i(2\rho_i - \delta_i)} \tag{16}$$

where ρ_i is the curvature of $p(u_i)$, ΔL_i is interpolation step, which is also defined by Eq. (15).

Generally, $\rho_i \gg \delta_i$ in the formula, so the chord error can be solved in Eq. (17).

$$\delta_i = \rho_i - \sqrt{\rho_i^2 - \frac{\Delta L_i^2}{4}} \approx \frac{(V(t_i) \cdot T)^2}{8\rho_i} \tag{17}$$

where

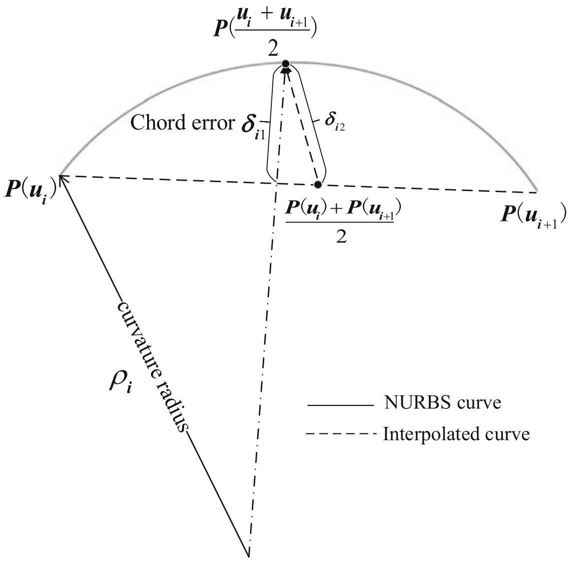


Fig. 2 Chord error in near-arc approximation

$$\rho_i = \frac{\left[[x'(u_i)]^2 + [y'(u_i)]^2 + [z'(u_i)]^2\right]^{\frac{3}{2}}}{\left[\begin{bmatrix} x'(u_i) & y'(u_i) \\ x''(u_i) & y''(u_i) \end{bmatrix}^2 + \begin{bmatrix} y'(u_i) & z'(u_i) \\ y''(u_i) & z''(u_i) \end{bmatrix}^2 + \begin{bmatrix} z'(u_i) & x'(u_i) \\ z''(u_i) & x''(u_i) \end{bmatrix}^2\right]^{\frac{1}{2}}} \tag{18}$$

Another method of calculating chord error is a direct estimation method shown δ_{i2} in Fig. 2. The δ expression is shown in Eq. (19)

$$\delta_i = \left| P\left(\frac{u_i + u_{i+1}}{2}\right) - \frac{P(u_i) + P(u_{i+1})}{2} \right| \tag{19}$$

Obliviously, the second method is not as precise as the first one, so the author adopted the first method to calculate chord error in this paper.

3.2 Design of Trajectory Planning Interpolator

The architecture of trajectory planning interpolator consists of two modules: parameter calculation module based on the given demonstrating points and velocity adaptive adjustment module, which is shown in Fig. 3. As is displayed in Fig. 3, the input of the interpolator includes demonstrating points, maximal allowance of interpolation chord error, maximum permissible velocity and interpolation cycle. The geometric information of NURBS includes knots vector, control points and corresponding weights as aforementioned.

Parameter calculation module is applied to compute relevant parameter of curve fitting based on sampling points. Velocity adaptive adjustment module was working under the conditional of chord error going out of range. It has been divided into two parts: chord error detection part and re-interpolation part. Real time detection of interpolation chord error is based on the fitting geometrical curve of NURBS. At each interpolation point, the chord error is compared with the given tolerance. If the chord error does not exceed the required range, the interpolator will continue to interpolate the following points along the path, otherwise velocity adaptive adjustment module will lower the manipulator velocity by an increment until the chord error meet the requirement.

The trajectory planning algorithm is conducted by the following steps:

(1) Using the given demonstrating points to calculate relevant parameters available for curve fitting based on NURBS.
(2) Make a density of u_i according to the interpolation cycle between [0,1].
(3) Calculate the knot value u_{i+1} from u_i and end-effector's velocity V.
(4) Calculate the chord error δ_i at the point of u_i.

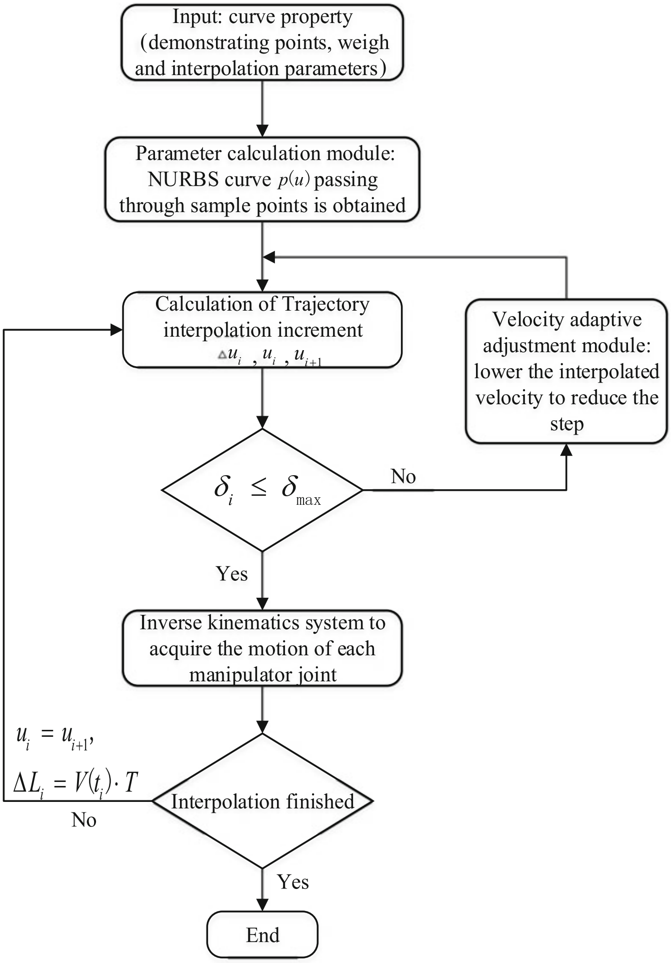

Input: curve property (demonstrating points, weigh and interpolation parameters)
Parameter calculation module: NURBS curve $p(u)$ passing through sample points is obtained
Calculation of Trajectory interpolation increment Δu_i, u_i, u_{i+1}
Velocity adaptive adjustment module: lower the interpolated velocity to reduce the step
$\delta_i \leq \delta_{\max}$
No
Yes
Inverse kinematics system to acquire the motion of each manipulator joint
$u_i = u_{i+1}$,
$\Delta L_i = V(t_i) \cdot T$
Interpolation finished
No
Yes
End

Fig. 3 Architecture of trajectory planning interpolator

(5) Make a comparison of δ_i and δ_{max}, if $\delta_i < \delta_{max}$ jump to next step. Otherwise, lower the velocity by an stable increment and to make the chord error gradually meet requirements.

(6) Exit the velocity adaptive adjustment module with the original velocity to interpolated the consecutive cycle.

(7) Make a dynamic analysis of manipulator and acquire joints movement using the inverse transformation method.

(8) Transmit the kinematic parameter to robot controlling system and console output control singles to motors.

For the trajectory planning problems based on given demonstrating points, the path interpolation points are continuously generated according to the above steps. Corresponding interpolation command will be sent to the robot servo system to drive the robot joints movement. Generally, the longer of interpolation steps is, the larger of chord error will be, so a designed scheme of closed loop controlling system for chord error detection is set up.

4 Experiment Verification

In this part, a 4-DOF free floating space robot (FFSR) was selected as the research object to testify the availability of the trajectory planning algorithm, as it is presented in Fig. 4. This robot consists of a free-floating platform and a manipulator with its first link mounted on the base. No external forces are applied on this system. Manipulator with a pneumatic nozzle installed at the end-effector is used to adjust the pose and position of the platform when the satellite needs to change orbit in space. Powerful gasses expelled by the pneumatic nozzle would generate great recoil force to realize the change of satellite in different operational orbits, which requires precise control of end-effector trajectory.

In order to describe the kinematic properties such as displacement, velocity and acceleration of different joints, Denavit-Hartenberg (D-H) modeling approach is introduced to construct the robot kinematical equation. A transformation matrix

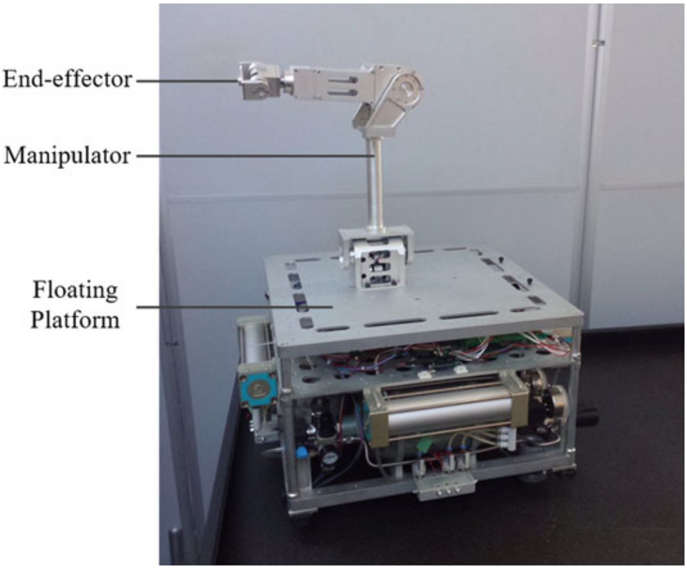


Fig. 4 4-DOF free floating space robot

from initial to final position is applied to describe the spatial relationship of different links, which simplifies the analysis process into 4*4 homogeneous transformation matrix that link the working space coordinate with joints space coordinate. Figure 5 is the link coordinate system, and Table 1 is the relevant parameters of D-H methodology.

The production of the link transformations yields forward kinematic solution of robot manipulator. The transformation matrix between the platform and end effector of the free floating space robot is defined by Eq. (20)

$$ {}_{4}^{0}T = {}_{1}^{0}T(q_1){}_{2}^{1}T(q_2){}_{3}^{2}T(q_3){}_{4}^{3}T(q_4) \tag{20} $$

So, the joint variable values with the given pose and position of robot end effector can be obtained by inverse kinematics solution transformations. In order to evaluate the proposed interpolated algorithm, eleven demonstrating points located in robotic working space were chosen for case study. Corresponding interpolation parameters can be calculated following the law of Eqs. (1)–(8).

- The sequential control points and weights are shown in Table 2.
- The knots vector is:

$$ U = \left\{ \begin{array}{c} 0, 0, 0, 0, 0.08, 0.17, 0.25, 0.34, 0.42, \\ 0.5, 0.59, 0.67, 0.75, 0.86, 0.93, 1, 1, 1, 1 \end{array} \right\} $$

- The interpolation cycle is $T = 2$ ms.
- The maximal allowed chord error is $\delta = 2$ μm.

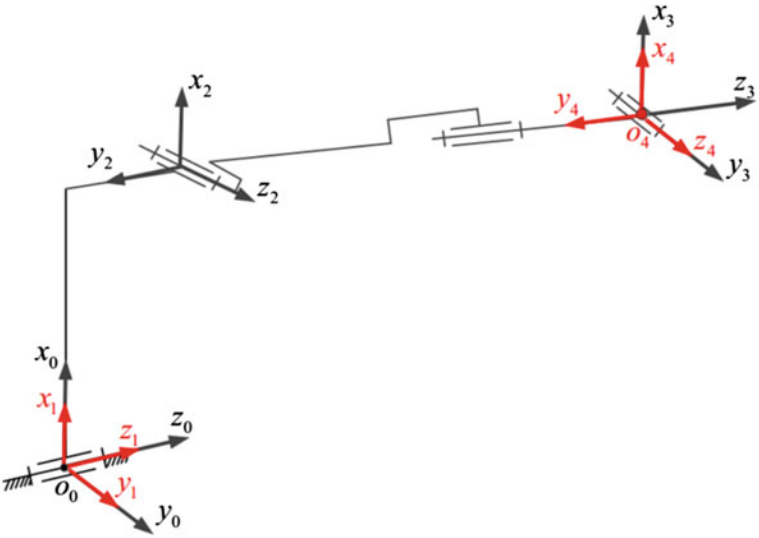


Fig. 5 The construction of D-H coordinate system

Table 1 D-H kinematic parameters for robot manipulator

Link (i)	$a_{i\text{-}1}$	$\alpha_{i\text{-}1}$	d_i	θ_i	Link parameters (mm)
1	0	0°	d	θ_1	L_1 = 299 mm
2	$\boldsymbol{L}_1$	−90°	0	θ_2	L_2 + L_3 = 232.9 mm
3	0	90°	$\boldsymbol{L}_2 + \boldsymbol{L}_3$	θ_3	d = 45 mm
4	0	−90°	0	θ_4	

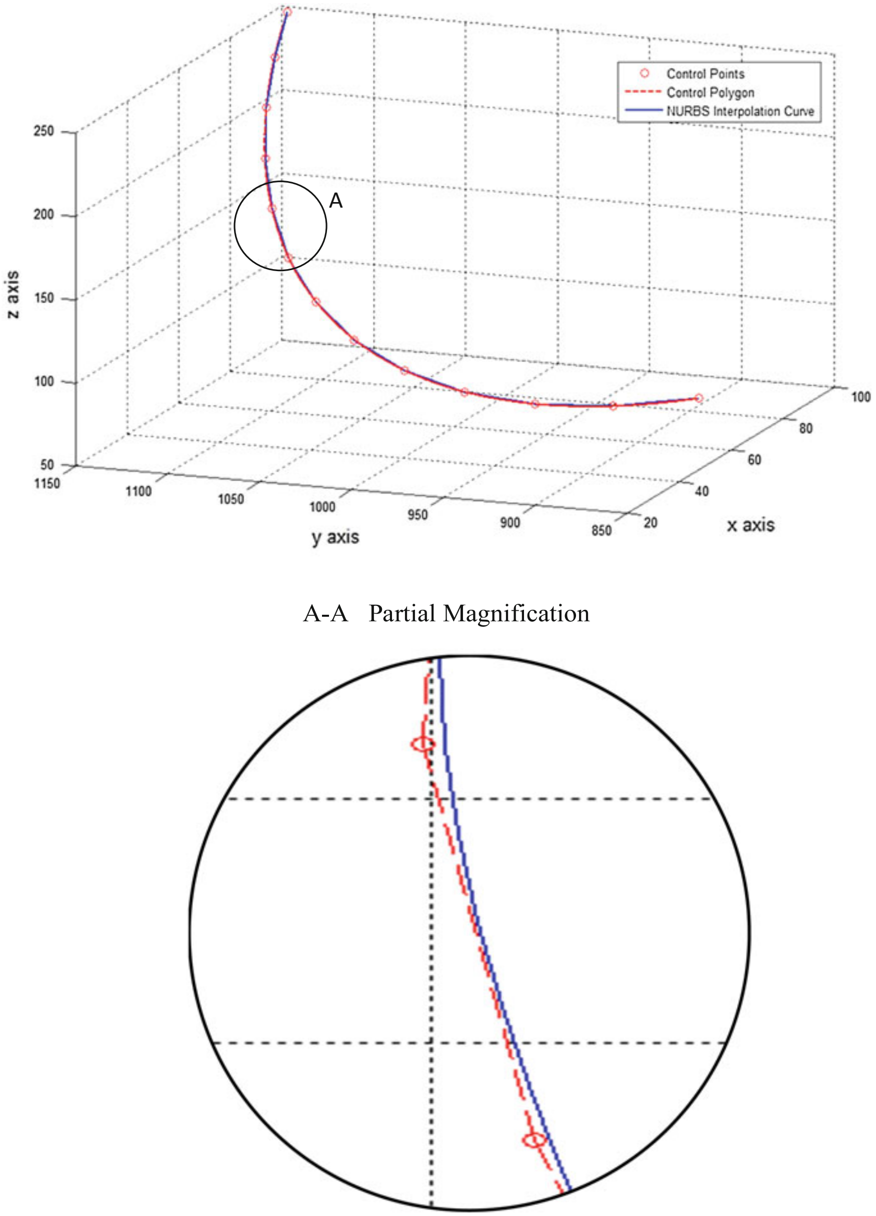
Table 2 Control points and weigh of NURBS curve

	Control points			Weight
	X	y	z	
d0	79.3244	894.368	58.6463	1
d1	61.2189	915.313	69.1001	1
d2	45.6827	936.258	83.0901	1
d3	33.3948	957.203	100.005	1
d4	24.8924	978.148	119.104	1
d5	20.547	999.093	139.554	1
d6	20.5487	1020.04	160.461	1
d7	24.8973	1040.98	180.911	1
d8	33.4027	1061.93	200.009	1
d9	45.6932	1082.87	216.922	1
d10	61.2316	1103.82	230.909	1
d11	79.3388	1124.76	241.36	1
d12	99.2232	1145.71	247.818	1

Curve interpolation of demonstrating points is shown in Fig. 6. In the following, the above curve trajectory is designed for robot end-effector automatic tracking using the velocity adaptive interpolation algorithm. Then the proposed motion planning structure including the constant and variable velocity is used to interpolate the NURBS curve to obtain the exact position of end-effector.

The real-time pose and position of robot end-effector can be measured by sensors, then compare the interpolation chord error curves in two different occasions: with and without velocity adaptive adjustment. The result is shown in Fig. 7. According to the figure, the maximal chord error has jumped with 2.8009 μm on time 0.3897 s without adaptive velocity adjustment. The velocity adaptive adjustment module didn't work until time points 0.3897 s as the chord error wasn't going beyond the range, so the chord error curves overlapped together. After the chord error was limited, the interpolation was automatically lower the velocity and thus two errors curves were obviously different.

Figure 8 showed the angular velocity curves of all joints with an abrupt decrease at the time 0.3897 s. It is obvious that the sharp changes in angular velocity curves were in consistent with chord error juming in Fig. 7. For velocity adaptive trajectory planning algorithm lowered manipulator velocity when the chord error exceeded the required mixmial value. The end-effector travels along the NURBS



A-A Partial Magnification

Fig. 6 Curve interpolation of demonstrating points

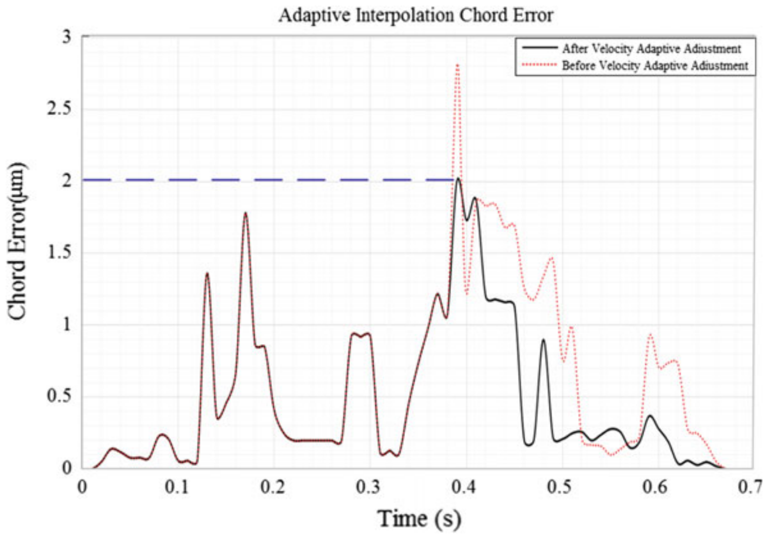


Fig. 7 Chord error measurement during interpolation

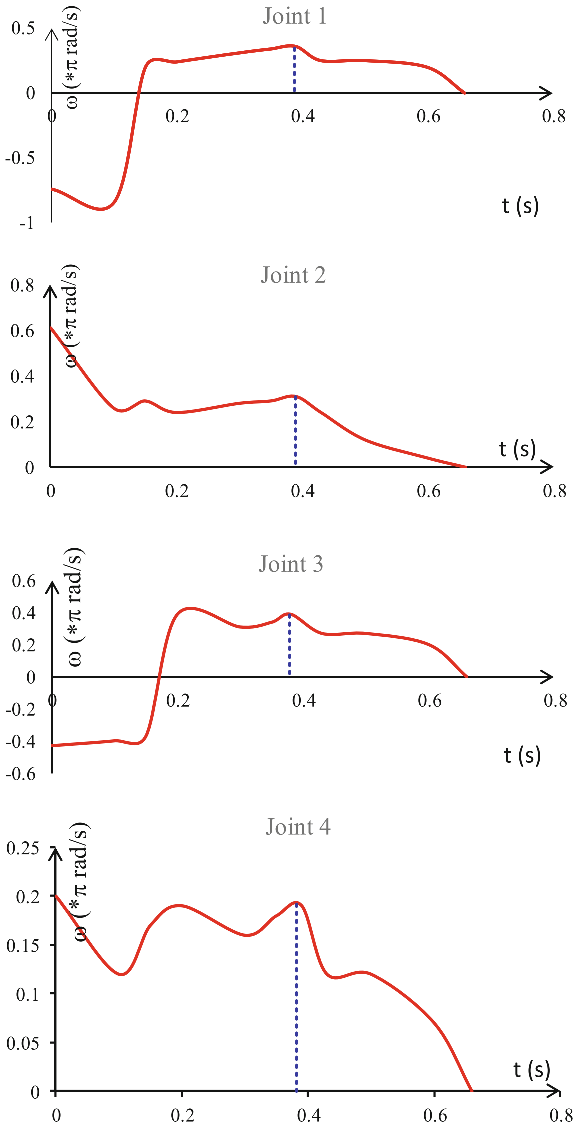
fitting curve with the maximum feedrate no more than $V_{max} = 500\ (\mathrm{mm/s})$ at each sampling step. The experienmental result have verified the correctness of the algorithm.

5 Conclusions

In the paper, a novel robot trajectory planning algorithm based on velocity adaptive adjustment is introduced using NURBS curve. The propose of this algorithm is to present a high-accuracy trajectory planning methodology on the basis of several demonstrating points, maxmimal chord error in this paper was set as 2 μm, so tracking errors of the NURBS fitting curve is smaller than 2 μm. In this way, the trajectory planning algorithm is also applicable to a demonstrating curve interpolation.

The algorithm is divided into two modules: parameter calculation module based on the given demonstrating points and velocity adaptive adjustment module. Detailed calculation process of curve fitting and interpolation was given, and finally the author adopted a 4-DOF free floating space robot to testify the adaptation of the interpolated algorithm, the result confirmed that the designed algorithm have a good property to reduce the interpolating error by adjusting the velocity. Future work will mainly be focused on how to optimize the acceleration and jerk based on this algorithm.

Fig. 8 Angular velocity of robot joints



References

1. Zhang L, Bian Y, Chen H, Wang K. Implementation of a CNC NURBS curve interpolator based on control of speed and precision. Int J Prod Res. 2009;47(6):1505–19.
2. Yoshida N, Fukuda R, Saito T, Saito T. Quasi-Log-aesthetic curves in polynomial Bézier form. Comput Aided Design Appl. 2013;10(6):983–93.
3. Mcgee JK. Linear and circular interpolation contouring control using repeated computation. US; 1972.

4. Saito T, Yamada M, Yoshida N. Shape analysis of cubic Bézier curves—correspondence to four primitive cubics. Comput Aided Design Appl. 2014;11(5):568–78.
5. Yang L, Zeng X-M. Bézier curves and surfaces with shape parameters. Int J Comput Math. 2009;86(7):1253–63.
6. Irk D. Sextic B-spline collocation method for the modified Burgers' equation. Kybernetes. 2009;38(9):1599–620.
7. Park J, Nam S, Yang M. Development of a real-time trajectory generator for NURBS interpolation based on the two-stage interpolation method. Int J Adv Manuf Technol. 2005;26 (4):359–65.
8. Pal P. A reconstruction method using geometric subdivision and NURBS interpolation. Int J Adv Manuf Technol. 2008;38(3):296–308.
9. Huo F, Poo A-N. Free-form two-dimensional contour error estimation based on NURBS interpolation. Appl Mech Mater. 2012;157–158(5):236–40.
10. Ahmad F, Liu X, Yamazaki K, Zhang X, Mori M. A real time scheme of intelligent nurbs interpolation for CNC systems to machine sculptured surfaces. Trans North Am Manuf Res Inst SME. 2004;32:223–30.
11. Zhao Z, Sun S. 6-DOF robot kinematics analysis and PTP motion control study based on Matlab. In: Jin D, Lin S, editors. Advances in mechanical and electronic engineering, vol. 1. Berlin: Springer; 2012. p. 21–6.
12. Slušný S, Zerola M, Neruda R. Real time robot path planning and cleaning. In: Huang D-S, Zhang X, Reyes García CA, Zhang L, editors. Advanced intelligent computing theories and applications with aspects of artificial intelligence: 6th international conference on intelligent computing, ICIC 2010, Changsha, China, 18–21 Aug 2010. Berlin, Heidelberg: Springer Berlin Heidelberg; 2010. p. 442–9.
13. Kim B-H. A Method for inverse kinematic solutions of 3R manipulators with coupling. In: Lee J, Lee MC, Liu H, Ryu J-H, editors. Intelligent robotics and applications: 6th international conference, ICIRA 2013, Busan, South Korea, 25–28 Sept 2013, Part I. Berlin: Springer; 2013. p. 742–9.
14. Young K-Y, Chen J-J. Implementation of a variable D-H parameter model for robot calibration using an FCMAC learning algorithm. J Intell Robot Syst. 1999;24(4):313–46.
15. Sánchez-Reyes J, Fernández-Jambrina L. Curves with rational chord-length parametrization. Comput Aided Geomet Design. 2008;25(4–5):205–13.
16. Gopi M, Manohar S. A unified architecture for the computation of B-spline curves and surfaces. IEEE Trans Parallel Distrib Syst. 1997;8(12):1275–87.
17. Fang Y, Liu HX, Zhang N, Guo GL, Wan YJY, Fang J. NURBS. Bioinformatics. 2013;29 (2):295–7.
18. Zhu W. The multilevel structures of NURBs and NURBlets on intervals. Dissertations & Theses—Gradworks; 2009.
19. Li YE, Xie MH. Third-order non-uniform rational B-spline curve fitting based on method of accumulating chord length. J Huaqiao Univ. 2010.
20. Sikder S, Barari A, Kishawy HA. Global adaptive slicing of NURBS based sculptured surface for minimum texture error in rapid prototyping. Rapid Prototyping J. 2015;21(6):649–61.
21. Tianmiao W, Cao Y, Chen Y, et al. NURBS interpolation and feedrate adaptive control based on de Boor algorithm. China Mech Eng. 2007;18(21):2608–13.

Author Biographies

Qingxiao Zou born in 1990, is currently a master candidate in Institute of Robotics, School of Mechanical Engineering and Automation, Beihang University, China. His research interests include robot control engineering and spatial kinematics.

Weidong Guo born in 1962, is currently a professor and master candidate supervisor at Institute of Robotics, Beihang University, China. His main research interests include spatial kinematics, robotics and haptics.

Fouaz Younès Hamimid born in 1991, is currently a Master candidate at Beihang University, China. He has Bachelor degree and his first Master Degree in Polytechnic Nationa School of Oran (ENPO), Algeria in 2015. His research interests include Machinery design.