Computer Aided Synthesis of Piecewise Rational Motions for Spherical 2R and 3R Robot Arms

Anurag Purwar
e-mail: anurag.purwar@stonybrook.edu

Zhe Jin

Q. J. Ge
e-mail: qiaode.ge@stonybrook.edu

Department of Mechanical Engineering,
Stony Brook University,
Stony Brook, NY 11794-2300

This paper deals with the problem of synthesizing smooth piecewise rational spherical motions of an object that satisfies the kinematic constraints imposed by a spherical robot arm with revolute joints. This paper brings together the kinematics of spherical robot arms and recently developed freeform rational motions to study the problem of synthesizing constrained rational motions for Cartesian motion planning. The kinematic constraints under consideration are workspace related constraints that limit the orientation of the end link of robot arms. This paper extends our previous work on synthesis of rational motions under the kinematic constraints of planar robot arms. Using quaternion kinematics of spherical arms, it is shown that the problem of synthesizing the Cartesian rational motion of a 2R arm can be reduced to that of circular interpolation in two separate planes. Furthermore, the problem of synthesizing the Cartesian rational motion of a spherical 3R arm can be reduced to that of constrained spline interpolation in two separate planes. We present algorithms for the generation of C^1 and C^2 continuous rational motion of spherical 2R and 3R robot arms. [DOI: 10.1115/1.2976444]

1 Introduction

1.1 Background and Previous Work. This paper deals with the problem of synthesizing Cartesian trajectories of spherical robot arms using piecewise rational motions. Synthesizing rational motions in Cartesian space as opposed to the joint space has the advantages of being direct and easier to visualize. The problem studied in this paper is defined as follows: Given a set of orientations of the end link of a spherical 2R or a 3R robot arm, we synthesize a smooth piecewise rational motion of the end link that passes through the given orientations and satisfies the kinematic constraints of the robot arm. Since the seminal work of Shoemake [1], the past two decades have witnessed significant progress in merging of the field of computer aided geometric design (CAGD) and kinematics for the development of rational Bézier and *B*-spline motions of rigid bodies. This includes the extension of Shoemake's spherical linear interpolation (*slerping*) of quaternions to other quaternion based splines such as Pletinckx [2], Dam et al. [3], Duff [4], Kim et al. [5], Kim and Nam [6], Nielson [7,8], Wang and Joe [9], and Barr et al. [10] as well as the extension to dual quaternions for spatial motion interpolation such as Ge and Ravani [11,12], Jüttler and Wagner [13], Purwar and Ge [14], and Purwar et al. [15]. More references on the subject can be found in a survey paper by Röschel [16]. Rational motions are an especially attractive proposition since they integrate well with the industry standard NURBS based CAD/CAM software packages. Furthermore, from a computational perspective, they can easily exploit fast and stable algorithms from CAGD. The motivation for this work lies in solving the motion synthesis problem in the areas of task specification in mechanism synthesis and robot motion planning for a large class of chains and mechanisms. Furthermore, this work advances the emerging field of computational kinematics, which deals with the development and application of general computational algorithms and numerical methods for solving a broad class of problems that arise from the analysis and synthesis of mechanisms and manipulators.

The existing works on Cartesian motion planning, such as the ones by Horsch and Jüttler [17] and Wagner and Ravani [18], seek direct application of rational motions to motion planning of robots and have not dealt with rational motions under kinematic constraints. Jin and Ge [19] recently presented their work on rational motion synthesis of planar 2R and 3R robot arms under the kinematic constraints. This paper extends their work to the synthesis of smooth rational motions under the kinematic constraints imposed by spherical 2R and 3R robot arms.

1.2 Our Approach. In this paper, the spherical displacement of the end link is represented by a quaternion; see Bottema and Roth [20] and McCarthy [21] for quaternion representation of displacements. A curve in the space of quaternions corresponds to a one-parameter rational motion in the Cartesian space. The kinematic constraints of the robot arms are transformed into geometric constraints for the design of quaternion curves. It is shown that the problem of synthesizing the Cartesian rational motion of a spherical 2R robot arm is reduced to that of circular interpolations in two separate planes. The problem of synthesizing the Cartesian rational motion of a spherical 3R robot arm is reduced to that of constrained spline interpolation in two different planes. We show that in the case of the 3R robot arm, it is possible to get an exact C^2 continuous rational motion, while the same method can be used to synthesize a C^2 continuous rational motion for a spherical 2R robot arm that approximates the kinematic constraint within the limits of a user defined value.

We note here that the rational motion interpolation of the end link of the spherical robot arms can also be performed by a composition of rational motion for each link. This will have the advantage of giving a rational motion for each moving link of the arm. However, in this paper, we are only concerned with the interpolation of the end link of a robot arm.

1.3 Organization. The organization of the paper is as follows. Section 2 reviews the kinematics of spherical 2R and 3R robot arms using quaternions. Section 3 deals with the problem of circular interpolation using piecewise rational Bézier form. Section 4 studies the problem of Cartesian motion planning for 2R and 3R robot arms, and in the end we present several examples to illustrate the effectiveness of our approach.

Contributed by the Mechanisms and Robotics Committee of ASME for publication in the JOURNAL OF MECHANICAL DESIGN. Manuscript received October 24, 2006; final manuscript received July 16, 2008; published online September 23, 2008. Review conducted by José M. Rico. Paper presented at the ASME 2006 Design Engineering Technical Conferences and Computers and Information in Engineering Conference (DETC2006), Philadelphia, PA, September 10–13, 2006.

2 Kinematics of Spherical 2R and 3R Robot Arms

In this section, we review the concept of quaternions and constraint manifold of the spherical 2R and 3R robot arms insofar as necessary for the development of the current paper.

2.1 Unit Quaternion as Rotation. Any rotation in three-dimensional space has a rotation axis and a rotation angle about this axis. Let $\mathbf{s}=(s_x,s_y,s_z)$ denote a unit vector along the axis and θ denote the angle of rotation. They can be used to define the so-called *Euler–Rodrigues parameters:*

$$q_1 = s_x \sin(\theta/2), \quad q_2 = s_y \sin(\theta/2), \quad q_3 = s_z \sin(\theta/2), \quad q_4 = \cos(\theta/2) \tag{1}$$

The Euler–Rodrigues parameters and the quaternion units, 1, **i**, **j**, and **k**, can be combined to define a quaternion of rotation:

$$\mathbf{q} = q_1\mathbf{i} + q_2\mathbf{j} + q_3\mathbf{k} + q_4 \tag{2}$$

A quaternion **q**, at times, is also written as an ordered quadruple (q_1,q_2,q_3,q_4). If $q_1^2+q_2^2+q_3^2+q_4^2=1$, **q** is also called a unit quaternion. See Bottema and Roth [20] and McCarthy [21] for more details on quaternions and the ensuing discussion.

The components of a quaternion are related to the matrix form of a rotation given by

$$[R] = \frac{1}{S^2}\begin{bmatrix} q_4^2+q_1^2-q_2^2-q_3^2 & 2(q_1q_2-q_4q_3) & 2(q_1q_3+q_4q_2) \\ 2(q_2q_1+q_4q_3) & q_4^2-q_1^2+q_2^2-q_3^2 & 2(q_2q_3-q_4q_1) \\ 2(q_3q_1-q_4q_2) & 2(q_3q_2+q_4q_1) & q_4^2-q_1^2-q_2^2+q_3^2 \end{bmatrix} \tag{3}$$

where $S^2=q_1^2+q_2^2+q_3^2+q_4^2$.

Note that when q_i is replaced by $Q_i=wq_i$ $(i=1,2,3,4)$, where w is a nonzero scalar, the matrix $[R]$ is unchanged. Thus, the quaternion components of **q** can be considered as homogeneous coordinates of a rotation. Ravani and Roth [22] considered these components as defining a point in a projective three-space, called the image space of spherical kinematics. In this way, a polynomial curve in the image space corresponds to a rational motion. By applying CAGD techniques for designing curves in the image space, we obtain rational motions in the Cartesian space. Let $\mathbf{Q}_i$, $i=0,\ldots,n$ be given quaternions, and then the following represents a B-spline quaternion curve in the space of quaternions:

$$\mathbf{Q}(u) = \sum_{i=0}^{n} N_{i,p}(u)\mathbf{Q}_i \tag{4}$$

where $N_{i,p}(u)$ are pth-degree basis functions. See Farin [23], Hoschek and Lasser [24], and Piegl and Tiller [25] for details on the B-spline curves. A representation for the rational B-spline motion in the Cartesian space can be obtained by substituting the coordinates of $\mathbf{Q}(u)$ given by Eq. (4) into the homogeneous matrix $[R]$ (Eq. (3)). It is easy to see that if the B-spline curve $\mathbf{Q}(u)$ is expressed as a polynomial function of degree p with parameter u treated as time, and then the matrix $[R]$ (where each element is a ratio of 2 quadratic functions) represents a rational B-spline motion of degree $2p$.

2.2 Kinematic Constraints of Spherical 2R Robot Arm. In this subsection, we review the kinematic constraint of a spherical 2R robot arm that specifies the orientations obtainable by the end link of the arm. Consider a spherical 2R robot arm (see Fig. 1) with joint axes **a** and **b** that intersect at a fixed point **O**. The joint axes make an angle α with respect to each other. We attach a fixed frame O to **O** and moving frames A and B to each link. The joint angles are denoted as θ and ϕ for the first and the second joint, respectively. For details on the orientation of the moving frames, we refer the reader to McCarthy [21].

The orientation of the end link is given by the composition of a rotation θ about the z-axis of A, a rotation α about the displaced x-axis, and followed by another rotation ϕ about the z-axis of B. These three rotations are given by the following quaternions:

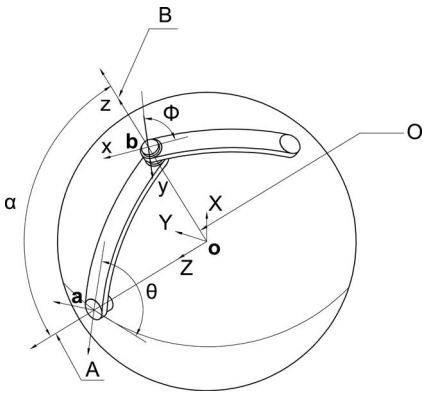

Fig. 1 A Spherical 2R robot arm

$$\mathbf{Z}(\theta) = (0,0,\sin(\theta/2),\cos(\theta/2))$$

$$\mathbf{X}(\alpha) = (\sin(\alpha/2),0,0,\cos(\alpha/2))$$

$$\mathbf{Z}(\phi) = (0,0,\sin(\phi/2),\cos(\phi/2)) \tag{5}$$

The product of these quaternions gives a quaternion defining a general position for the end link of the spherical 2R robot arm:

$$\mathbf{q}(\theta,\alpha,\phi) = \mathbf{Z}(\theta)\mathbf{X}(\alpha)\mathbf{Z}(\phi) \tag{6}$$

Expanding Eq. (6), we obtain $\mathbf{q}(\theta,\alpha,\phi)=(q_1,q_2,q_3,q_4)$, where

$$q_1 = \sin(\alpha/2)\cos((\theta-\phi)/2)$$

$$q_2 = \sin(\alpha/2)\sin((\theta-\phi)/2)$$

$$q_3 = \cos(\alpha/2)\sin((\theta+\phi)/2)$$

$$q_4 = \cos(\alpha/2)\cos((\theta+\phi)/2) \tag{7}$$

From Eq. (7), we can derive the following algebraic equation:

$$\frac{q_1^2+q_2^2}{q_3^2+q_4^2} = \tan^2(\alpha/2) \tag{8}$$

This equation characterizes the kinematic constraint of a spherical 2R robot arm and is said to define the constraint manifold for the 2R arm (McCarthy [21]). Equation (8) is equivalent to

$$\frac{q_1^2+q_2^2}{q_1^2+q_2^2+q_3^2+q_4^2} = \sin^2(\alpha/2) \tag{9}$$

or

$$\frac{q_3^2+q_4^2}{q_1^2+q_2^2+q_3^2+q_4^2} = \cos^2(\alpha/2) \tag{10}$$

Note that when $q_i(i=1,2,3,4)$ is replaced by $Q_i=wq_i$, where w is a scalar, Eqs. (8)–(10) are unchanged.

These constraints define a two-dimensional surface in the quaternion space, which is obtained by intersecting the unit sphere with the quadrics defined by Eq. (8), (9), or (10). The orthogonal projection of this surface into the q_1q_2 and q_3q_4 planes gives two circles. Thus, the problem of interpolating orientations of the end link of a spherical 2R arm can be reduced to that of circular interpolations in two separate planes. We note here that even though the kinematic constraint expressed by Eqs. (8)–(10) are equivalent to one another, we still need to perform circular interpolation in two different planes. This is so because Eqs. (9) and (10) each contribute only half of the four quaternion coordinates.

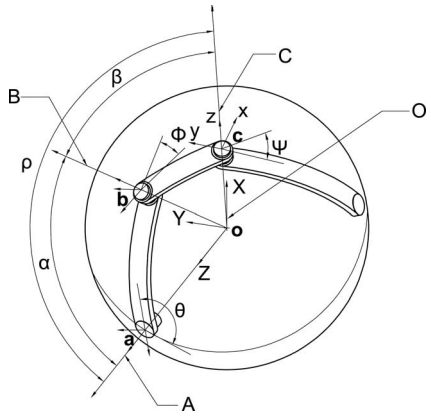

Fig. 2 A Spherical 3R robot arm

As we will show later, the problem is greatly simplified since the circular interpolation is a standard problem in CAGD and has been effectively solved.

2.3 Kinematic Constraints of Spherical 3R Robot Arm. Consider a spherical 3R robot arm (see Fig. 2) with joint axes **a**, **b**, and **c** intersecting a fixed point **O**. The axes **a** and **b** make an angle α, and **b** and **c** make an angle β. We attach a fixed frame O to **O** and moving frames A, B, and C to each links. The joint angles are denoted as θ, ϕ, and ψ for the successive joints. For details on the orientation of the moving frames, we again refer the reader to McCarthy [21]. The orientation of the end link is given by the following quaternion product:

$$\mathbf{q}(\theta,\phi,\psi) = \mathbf{Z}(\theta)\mathbf{X}(\alpha)\mathbf{Z}(\phi)\mathbf{X}(\beta)\mathbf{Z}(\psi) \tag{11}$$

where

$$\mathbf{Z}(\theta) = (0,0,\sin(\theta/2),\cos(\theta/2))$$

$$\mathbf{X}(\alpha) = (\sin(\alpha/2),0,0,\cos(\alpha/2))$$

$$\mathbf{Z}(\phi) = (0,0,\sin(\phi/2),\cos(\phi/2))$$

$$\mathbf{X}(\beta) = (\sin(\beta/2),0,0,\cos(\beta/2))$$

$$\mathbf{Z}(\psi) = (0,0,\sin(\psi/2),\cos(\psi/2)) \tag{12}$$

By expanding the product in Eq. (11), we obtain $\mathbf{q}(\theta,\phi,\psi) = (q_1,q_2,q_3,q_4)$, where

$$q_1 = \cos\left(\frac{\phi}{2}\right)\sin\left(\frac{\alpha+\beta}{2}\right)\cos\left(\frac{\theta-\psi}{2}\right) + \sin\left(\frac{\phi}{2}\right)\sin\left(\frac{\alpha-\beta}{2}\right)\sin\left(\frac{\theta-\psi}{2}\right)$$

$$q_2 = \cos\left(\frac{\phi}{2}\right)\sin\left(\frac{\alpha+\beta}{2}\right)\sin\left(\frac{\theta-\psi}{2}\right) - \sin\left(\frac{\phi}{2}\right)\sin\left(\frac{\alpha-\beta}{2}\right)\cos\left(\frac{\theta-\psi}{2}\right)$$

$$q_3 = \cos\left(\frac{\phi}{2}\right)\cos\left(\frac{\alpha+\beta}{2}\right)\sin\left(\frac{\theta+\psi}{2}\right) + \sin\left(\frac{\phi}{2}\right)\cos\left(\frac{\alpha-\beta}{2}\right)\cos\left(\frac{\theta+\psi}{2}\right)$$

$$q_4 = \cos\left(\frac{\phi}{2}\right)\cos\left(\frac{\alpha+\beta}{2}\right)\cos\left(\frac{\theta+\psi}{2}\right) - \sin\left(\frac{\phi}{2}\right)\cos\left(\frac{\alpha-\beta}{2}\right)\sin\left(\frac{\theta+\psi}{2}\right) \tag{13}$$

Equation (13) further gives

$$q_1^2 + q_2^2 = \cos^2\left(\frac{\phi}{2}\right)\sin^2\left(\frac{\alpha+\beta}{2}\right) + \sin^2\left(\frac{\phi}{2}\right)\sin^2\left(\frac{\alpha-\beta}{2}\right)$$

$$q_3^2 + q_4^2 = \cos^2\left(\frac{\phi}{2}\right)\cos^2\left(\frac{\alpha+\beta}{2}\right) + \sin^2\left(\frac{\phi}{2}\right)\cos^2\left(\frac{\alpha-\beta}{2}\right) \tag{14}$$

Since α and β satisfy the conditions $0<\alpha,\beta<\pi$, Eq. (14) reduces to the following inequality:

$$\tan^2((\alpha-\beta)/2) \le \frac{q_1^2+q_2^2}{q_3^2+q_4^2} \le \tan^2((\alpha+\beta)/2) \tag{15}$$

This inequality characterizes the kinematic constraint of a spherical 3R robot arm. Equation (15) is equivalent to

$$\sin^2((\alpha-\beta)/2) \le \frac{q_1^2+q_2^2}{q_1^2+q_2^2+q_3^2+q_4^2} \le \sin^2((\alpha+\beta)/2) \tag{16}$$

or

$$\cos^2((\alpha+\beta)/2) \le \frac{q_3^2+q_4^2}{q_1^2+q_2^2+q_3^2+q_4^2} \le \cos^2((\alpha-\beta)/2) \tag{17}$$

Note once again that when q_i (i=1,2,3,4) is replaced by $Q_i=wq_i$, where w is a scalar, Eqs. (15)–(17) are unchanged.

We assert that the kinematic constraint given by Eq. (15) is frame independent. See Appendix for a proof that uses spherical trigonometry.

3 Interpolation With a Piecewise Quadratic Rational Bézier Circular Arc

We have seen earlier that the problem of interpolating spherical orientations for a spherical 2R arm is reduced to that of circular interpolation in two planes. In this section, we review the algorithm given in a companion paper by Jin and Ge [19] for interpolating an ordered set of points lying on a circle with a piecewise quadratic rational Bézier circular arc. This algorithm will be used in planning C^1 rational motions of the given orientations under the kinematic constraint of spherical 2R robot arms.

The problem of circular interpolation is defined as follows.

Given. A set of points $\mathbf{x}_0,\ldots,\mathbf{x}_L$ on a circle with radius r and the corresponding parameter values (or, knots) $u_0,\ldots,u_L$.

Find. A piecewise quadratic rational Bézier circular arc $\mathbf{b}(u)$ that interpolates the given points at the specified parameters, i.e., $\mathbf{b}(u_i)=\mathbf{x}_i$, $i=0\ldots L$.

In turn, this problem requires (1) computation of the control points ($\mathbf{b}_{2i},\mathbf{b}_{2i+1},\mathbf{b}_{2i+2}$, $i=0\ldots L-1$) of all the L segments of the rational Bézier arc and (2) weights (w_i) associated with the control points.

Due to the requirement of C^1 continuity of the Bézier arc, the middle control point of every quadratic Bézier segment should lie at the intersection of the tangent lines at the end control points. Following this, Jin and Ge [19] gave the following equation for determining the control points:

$$\mathbf{b}_{2i} = \mathbf{x}_i, \quad \mathbf{b}_{2i+1} = \frac{r^2(\mathbf{x}_i+\mathbf{x}_{i+1})}{r^2+\mathbf{x}_i\cdot\mathbf{x}_{i+1}}, \quad \mathbf{b}_{2i+2} = \mathbf{x}_{i+1} \tag{18}$$

Furthermore, by imposing the conditions that the weights be selected such that each rational Bézier segment is a circular arc of radius r and that two adjacent segments join each other with C^1 continuity, they obtain

$$s_{2i} = \frac{s_{2i+1}}{2}\left[1 + \left(\frac{\mathbf{x}_i}{r}\right)\cdot\left(\frac{\mathbf{x}_{i+1}}{r}\right)\right], \quad i = 0, \ldots, L-1 \tag{19}$$

and

$$\frac{|\mathbf{b}_{2i+3} - \mathbf{b}_{2i+2}|}{|\mathbf{b}_{2i+2} - \mathbf{b}_{2i+1}|} = \frac{\Delta_{i+1}}{\Delta_i s_{2i+1} s_{2i+2}}, \quad i = 0, \ldots, L-2 \tag{20}$$

where $|\cdot|$ denotes the magnitude of a vector, and $\Delta_i = u_{i+1} - u_i$ and $s_{2i} = w_{2i+1}/w_{2i}$ denote the weight ratio; $i = 0, \ldots, L-1$. Without any loss of generality, we choose $s_0 = 1$ ($w_0 = 1, w_1 = 1$).

In summary, the algorithm for C^1 piecewise rational Bézier circular interpolation is given by Jin and Ge [19] as follows.

Algorithm 1

1. Find control points $\mathbf{b}_{2i}, \mathbf{b}_{2i+1}, \mathbf{b}_{2i+2}$, $i = 0, \ldots, L$ from the given points $\mathbf{x}_i$ using Eq. (18).
2. For ith piece of Bézier circular arc, use $[u_i, u_{i+1}]$ as the range for the parameter u and calculate the weights associated with each of the Bézier points from Eqs. (19) and (20).
3. Generate the C^1 interpolating nonuniform rational Bézier (NURB) circular arc from its piecewise rational Bézier form.

4 Rational Motions of Spherical 2R and 3R Robot Arms

In this section, we present algorithms for synthesizing C^1 and C^2 continuous piecewise rational motions of spherical 2R and 3R robot arms. The motion interpolates a set of given orientations of the end link.

The orientations of the end link of a spherical 2R robot arm can be given by either the Cartesian coordinates $(s_{xi}, s_{yi}, s_{zi}, \theta_i)$ or the joint coordinates (θ_i, ϕ_i). Similarly, the orientations of the end link of a spherical 3R robot arm can be given by either the Cartesian coordinates $(s_{xi}, s_{yi}, s_{zi}, \theta_i)$ or the joint coordinates $(\theta_i, \phi_i, \psi_i)$. Equation (1) can be used to convert Cartesian coordinates into quaternions, while Eq. (7) or Eq. (13) can be used to convert joint coordinates into quaternions for 2R and 3R robot arms, respectively. In the succeeding algorithms, we assume that if the orientations of the end link are given in terms of quaternions coordinates, they are all positive. Quaternions are unambiguous up to a sign only, and inconsistent use of sign has a profound impact on motion synthesis; see Ge and Purwar [26] for more on this issue. We ensure that no inconsistencies arise in the interpolation process by choosing all the weights to be positive.

4.1 C^1 Interpolating Rational Motion for Spherical 2R Robot Arm.

Given. A set of orientations of the end link of a spherical 2R robot arm in its workspace, the corresponding parameter values u_i, $i = 0, \ldots, L$, and the link length α of the first link.

Find. A C^1 rational motion of the end link that interpolates the given orientations at the parameter values and satisfies the kinematic constraints of the spherical 2R arm.

Algorithm 2

1. Convert the given orientations of the end link into unit quaternions $\mathbf{q}_i = (q_{i1}, q_{i2}, q_{i3}, q_{i4})$ using either Eq. (1) or Eq. (7).
2. Group quaternion components as $\mathbf{E}_i = (q_{i1}, q_{i2})$; $\mathbf{F}_i = (q_{i3}, q_{i4})$.
3. Set of ordered points $\mathbf{E}_i$ that lie on a circle of radius $\sin(\alpha/2)$ (see Eq. (9)). Set $\mathbf{x}_i = \mathbf{E}_i$, $r = \sin(\alpha/2)$ and use Algorithm 1 to interpolate $\mathbf{E}_i$ with a quadratic circular arc $\mathbf{E}(u)$.
4. Set of ordered points $\mathbf{F}_i$ that lie on a circle of radius $\cos(\alpha/2)$ (see Eq. (10)). Set $\mathbf{x}_i = \mathbf{F}_i$, $r = \cos(\alpha/2)$ and use Algorithm 1 to interpolate $\mathbf{F}_i$ with another quadratic circular arc $\mathbf{F}(u)$.
5. The image curve, $\mathbf{q}(u) = (\mathbf{E}(u), \mathbf{F}(u))$, is a C^1 continuous quadratic rational curve. Substituting $\mathbf{q}(u)$ into Eq. (3), we obtain a C^1 continuous quartic rational motion for a spherical 2R robot arm that interpolates the given set of orientations of the end link.

4.2 C^2 Interpolating Rational Motion for Spherical 3R Robot Arm.

This section presents an algorithm for the following constrained motion interpolation problem.

Given. A set of orientations of the end link of a spherical 3R robot arm in its workspace, the corresponding parameter values u_i, the associated weights w_i, $i = 0 \ldots L$, and link lengths α and β of the first and second links, respectively.

Find. A C^2 rational motion of the end link that interpolates the given orientations at the corresponding parameter values subject to the kinematic constraints of the spherical 3R arm.

Then kinematic constraint for a spherical 3R robot arm is given by inequalities in Eq. (16) or Eq. (17). Geometrically, the limits in these inequalities represent two rings of varying radius in two different planes. The radius of these two rings change along the parameter u because the denominator in the inequalities is not necessarily unity. Since the motion of the end link is required to satisfy these kinematic constraints, we can reduce the problem of designing a C^2 rational motion of the end link to that of constructing two separate spline curves confined to lie in the region defined by the two rings. In the succeeding discussion, we assume that the given orientations of the end link have been converted into unit quaternions $\mathbf{q}_i = (q_{i1}, q_{i2}, q_{i3}, q_{i4})$ using either Eq. (1) or Eq. (13) and thereafter into the nonunit quaternions $\mathbf{Q}_i (= (Q_{i1}, Q_{i2}, Q_{i3}, Q_{i4})) = w_i \mathbf{q}_i$.

We now present a sketch of the algorithm: We divide the quaternion components $\mathbf{Q}_i = (Q_{i1}, Q_{i2}, Q_{i3}, Q_{i4})$ in two groups $(\mathbf{E}_i, \mathbf{F}_i)$, where $\mathbf{E}_i = (Q_{i1}, Q_{i2})$ and $\mathbf{F}_i = (Q_{i3}, Q_{i4})$. Then, we construct two C^2 cubic B-spline curves[1] $\mathbf{E}(u)$ and $\mathbf{F}(u)$ that interpolate $\mathbf{E}_i$ and $\mathbf{F}_i$ at parameter values u_i, respectively. Let us call the plane defined by the first two quaternion coordinates (q_1, q_2) as $s_1 s_2$ plane and by the last two quaternion coordinates (q_3, q_4) as $s_3 s_4$ plane. The curve $\mathbf{E}(u)$ lies in the $s_1 s_2$ plane, while the curve $\mathbf{F}(u)$ lies in the $s_3 s_4$ plane. However, these two curves are not guaranteed to lie inside the rings described earlier. In the Cartesian space, this means that the kinematic constraint given by Eq. (15) is not satisfied at all the values of the parameter u. The kinematic constraint (Eq. (15)) for a continuous valued parametric curve can be rewritten as

$$\tan^2((\alpha - \beta)/2) \le g(u) \le \tan^2((\alpha + \beta)/2) \tag{21}$$

where

$$g(u) = \frac{\mathbf{E}(u) \cdot \mathbf{E}(u)}{\mathbf{F}(u) \cdot \mathbf{F}(u)}$$

The idea behind the algorithm to be presented shortly is to first detect all the extrema of the function $g(u)$ by using piecewise Bézier representation of the B-spline curves. Then we test if an extremum violates the constraint. If it does, then we replace such an extremum with a new point that satisfies the constraint. We call an extremum that violates the constraint as an extreme point. In adding a new point, we impose following restrictions on it: (1) It should be inside the rings, (2) it should be minimally away from the extreme point, and (3) it should be added for the same parameter value as that of the extreme point. The idea is to deform the curve minimally in the vicinity of the extreme point by the addition of this new point that is just inside the ring. This new point is added to the set of points to be interpolated and new C^2 B-spline curves that interpolate these points are constructed. This process is repeated until no extreme points are detected. At the end, we have

[1]Designing a C^2 B-spline curve is a standard scheme in CAGD (Farin [23], Hoschek and Lasser [24], and Piegl and Tiller[25]).

a C^2 B-spline constrained curve that does not violate the constraint. We note here that even though the two curves detect a common extreme point since the constraints given by Eqs. (16) and (17) are equivalent to each other, it is still necessary to construct two B-spline curves in each of the s_1s_2 and s_3s_4 plane. The curves $\mathbf{E}(u)$ and $\mathbf{F}(u)$ contribute two coordinates each to make up the complete quaternion curve $\mathbf{Q}(u)=(\mathbf{E}(u),\mathbf{F}(u))$.

Now we show how the extreme and new points are calculated. We calculate the extrema of function $g(u)$ to detect violations of the kinematic constraint. The extrema of $g(u)$ can be calculated easily by taking the derivative of the piecewise Bézier forms of $\mathbf{E}(u)$ and $\mathbf{F}(u)$. For the ith pair of Bézier segments, $\mathbf{E}_i(u)$ and $\mathbf{F}_i(u)$, with the parameter value $u \in [u_i, u_{i+1})$, we determine the extrema of the following function:

$$g_i(u) = \frac{\mathbf{E}_i(u) \cdot \mathbf{E}_i(u)}{\mathbf{F}_i(u) \cdot \mathbf{F}_i(u)} \tag{22}$$

By differentiating Eq. (22) and setting the resulting derivative to zero, we obtain

$$\frac{\mathbf{E}_i(u) \cdot \dot{\mathbf{E}}_i(u)}{\mathbf{E}_i(u) \cdot \mathbf{E}_i(u)} = \frac{\mathbf{F}_i(u) \cdot \dot{\mathbf{F}}_i(u)}{\mathbf{F}_i(u) \cdot \mathbf{F}_i(u)} \tag{23}$$

where $\dot{\mathbf{E}}_i$ and $\dot{\mathbf{F}}_i(u)$ denote the derivatives of $\mathbf{E}_i(u)$ and $\mathbf{F}_i(u)$ with respect to u. Equation (23) is a polynomial equation of degree 11 but due to the cancellation of the highest order term, it actually reduces to an equation of degree 10 and can be solved numerically for the parameter value u in the range $[u_i, u_{i+1})$. This procedure may yield many local extrema for different Bézier segments.

Let us assume that $g(u)$ has an extremum at $u=u^*$ that violates the constraint. Thus, we would like to add a new point $(\mathbf{E}_j, \mathbf{F}_j)$ at the parameter value u^*. We connect a line from the origin to $\mathbf{E}(u^*)$ and select $\mathbf{E}_j$ on this line; similarly we connect another line from the origin to $\mathbf{F}(u^*)$ and then select $\mathbf{F}_j$ on this line, such that $g(u^*)$ satisfies the inequality given in Eq. (21). We use the following rules for locating $\mathbf{E}_j$ and $\mathbf{F}_j$ on the respective lines:

If

$$\frac{|\mathbf{E}(u^*)|}{|\mathbf{F}(u^*)|} < |\tan((\alpha-\beta)/2)|$$

then

$$\frac{|\mathbf{E}_j|}{|\mathbf{F}_j|} = |\tan((\alpha-\beta)/2)| + \delta \tag{24}$$

and if

$$\frac{|\mathbf{E}(u^*)|}{|\mathbf{F}(u^*)|} > |\tan((\alpha+\beta)/2)|$$

then

$$\frac{|\mathbf{E}_j|}{|\mathbf{F}_j|} = |\tan((\alpha+\beta)/2)| - \delta \tag{25}$$

where δ is a user defined tolerance value, which can be chosen as small as possible. If an extremum is found at the limits of the inequality in Eq. (21), the constraints are considered satisfied and thus, no new points are added in that case. We note here that due to limitations of numerical computing, we actually determine if the difference between the extremum of function $g(u)$ and the limit values is smaller than a very small number, instead of directly comparing the two numbers for equality. The tolerance parameter δ controls the change in the shape of the curve between successive iterations. These rules seek to add a new point that is minimally away from the extreme point and satisfies the kinematic constraint. We note that choosing a truly inner point instead of a new point that is just inside a boundary of the ring may yield a curve that is unlikely to go out of the ring. This would not satisfy the desired condition of minimal change in the curve. The new point $\mathbf{E}_j$, $\mathbf{F}_j$ should also satisfy

$$\mathbf{E}_j \cdot \mathbf{E}_j + \mathbf{F}_j \cdot \mathbf{F}_j = w_j^2 \tag{26}$$

Here, without any loss of generality, we choose $w_j=1$.

Thus, the magnitude of new point $\mathbf{E}_j$, $\mathbf{F}_j$ can be calculated from Eq. (24) or Eq. (25), and Eq. (26). Since we already know the slope of the lines joining origin with the extreme point in two planes, we can uniquely locate the new point. Now, we present the following algorithm.

Algorithm 3

1. Convert the given orientations of the end link into unit quaternions $\mathbf{q}_i=(q_{i1},q_{i2},q_{i3},q_{i4})$ using either Eq. (1) or Eq. (13). Convert unit quaternions $\mathbf{q}_i$ into nonunit quaternions $\mathbf{Q}_i(=(Q_{i1},Q_{i2},Q_{i3},Q_{i4}))=w_i\mathbf{q}_i$.
2. Group quaternion components as $\mathbf{E}_i=(Q_{i1},Q_{i2})$; $\mathbf{F}_i=(Q_{i3},Q_{i4})$.
3. Construct a C^2 cubic B-spline curve $\mathbf{E}(u)$ that interpolates $\mathbf{E}_i$ and construct a C^2 cubic B-spline curve $\mathbf{F}(u)$ that interpolates $\mathbf{F}_i$ at parameter values u_i.
4. Calculate the extrema of the function $g(u)$ (Eq. (23)). For each extremum, test if $(\tan^2((\alpha-\beta)/2) \le \text{extremum } (g(u)) \le \tan^2((\alpha+\beta)/2)$. If yes, then the kinematic constraint of a spherical 3R robot arm is satisfied. Go to step 5. If an extremum point fails the test, say, at $u=u^*$, do the following and repeat for each extreme point.
(a) Connect a line from the origin to $\mathbf{E}(u^*)$ and add $\mathbf{E}_j$ on this line. Calculate the magnitude of $\mathbf{E}_j$ from Eq. (24) or Eq. (25), and Eq. (26).
(b) Connect a line from the origin to $\mathbf{F}(u^*)$ and add $\mathbf{F}_j$ on this line. Calculate the magnitude of $\mathbf{F}_j$ from Eq. (24) or Eq. (25), and Eq. (26).
(c) Locate $\mathbf{E}_j$ and $\mathbf{F}_j$ on the lines.
(d) Add the new point $(\mathbf{E}_j, \mathbf{F}_j)$ at the parameter value u^* to the list of points to be interpolated and go to step 3.
5. The image curve $\mathbf{Q}(u)=(\mathbf{E}(u),\mathbf{F}(u))$ defines the C^2 interpolating piecewise rational motion of degree 6 after the substitution into Eq. (3).

Even though we do not provide any theoretical guarantee, we have observed that this algorithm always converges. In this algorithm, the B-spline curve is generated using a global interpolation scheme (Piegl and Tiller [25]). In this scheme, although moving one of the interpolating points changes the curve globally, the change in the curve diminishes away from the modification point.

4.3 C^2 Interpolating Rational Motion for Spherical 2R Robot Arm. Due to the limitations of circular interpolation, the algorithm presented previously for spherical 2R robot arms generates only C^1 (i.e., velocity-continuous) piecewise rational motion. However, in high-speed applications, a C^2 (i.e., acceleration-continuous) motion is often desired. The problem is defined as follows.

Given. A set of orientations of the end link of a spherical 2R robot arm in its workspace, the corresponding parameter values u_i, the associated weights w_i, $i=0,\ldots,L$, and link length α of the first link.

Find. A C^2 rational motion of the end link that interpolates the given orientations at the parameter values subject to the kinematic constraints of the spherical 2R arm.

Bangert and Prautzsch [27] presented a method to obtain $(n-1)$ times differentiable periodic B-spline representations for the circle of degree $2n$. Thus as an example, a circle can be represented by a periodic quartic C^1 B-spline curve. The method is exact and can be applied to get a C^2 interpolating rational motion for spherical 2R robot arm. However, in this section, we show that we can use a slightly modified form of Algorithm 3 to plan the C^2 rational motion for a spherical 2R robot arm that approximates the

Table 1 Joint coordinates for spherical 2R robot arm motion planning

i	1	2	3	4	5
α(deg)			30		
θ_i(deg)	0	45	90	125	165
ϕ_i(deg)	0	15	30	40	30
u_i	0.0	0.3	0.6	0.8	1.0

kinematic constraints but produces a lower degree motion.

First, we transform the kinematic constraints of the spherical 2R robot arm to a form similar to that of spherical 3R robot arm. The idea is to transform the circular constraints of spherical 2R robot arm to the ring constraints. Considering that, in practice, there is always some clearance at the revolute joints of the robot arm, the link length α is not exact but actually a function of diametric clearance at both the joints. Thus, the kinematic constraint of the spherical 2R arm (Eq. (8)) can be modified as

$$\tan^2(\alpha/2) - \varepsilon \le h(u) \le \tan^2(\alpha/2) + \varepsilon \tag{27}$$

where

$$h(u) = \frac{q_1^2 + q_2^2}{q_3^2 + q_4^2} \tag{28}$$

and ε is an indicator of the clearance at joints. The value of ε can be calculated from the diametric clearance at the joints. For a smaller value of ε, the kinematic constraint is better approximated. This form of the modified constraint equation is similar to the one derived for spherical 3R robot arm and thus, Algorithm 3 can be applied. For a given value of ε, the constraints are considered violated, if the extrema of $h(u)$ do not satisfy the inequality in Eq. (27).

Let us assume that $h(u)$ has an extremum at $u=u^*$ that violates the constraint. Thus, we would like to add a new point $(\mathbf{E}_j, \mathbf{F}_j)$ at the parameter value u^*. Following the kinematic constraints of 2R robot arms given by Eqs. (9) and (10), the new point $(\mathbf{E}_j, \mathbf{F}_j)$ should satisfy

$$|\mathbf{E}_j| = w_j \sin(\alpha/2); |\mathbf{F}_j| = w_j \cos(\alpha/2) \tag{29}$$

Here, again without any loss of generality, we choose $w_j = 1$.

Now, we present the algorithm:

Algorithm 4

1. Convert the given orientations of the end link into unit quaternions $\mathbf{q}_i = (q_{i1}, q_{i2}, q_{i3}, q_{i4})$ using either Eq. (1) or Eq. (13). Convert unit quaternions $\mathbf{q}_i$ into nonunit quaternions $\mathbf{Q}_i (= (Q_{i1}, Q_{i2}, Q_{i3}, Q_{i4})) = w_i \mathbf{q}_i$.
2. Group quaternion components as $\mathbf{E}_i = (Q_{i1}, Q_{i2})$; $\mathbf{F}_i = (Q_{i3}, Q_{i4})$.
3. Construct a C^2 cubic B-spline curve $\mathbf{E}(u)$ that interpolates $\mathbf{E}_i$ and construct a C^2 cubic B-spline curve $\mathbf{F}(u)$ that interpolates $\mathbf{F}_i$ at parameter values u_i.
4. Calculate the extrema of the function $h(u)$ (Eq. (28)). For each extremum, given a value of ε, test if $(\tan^2(\alpha/2) - \varepsilon \le \text{extrema}(h(u)) \le \tan^2(\alpha/2) + \varepsilon)$. If yes, then the kinematic constraint of a spherical 2R robot arm are satisfied approximately. Go to step 5. If an extremum point fails the test, say, at $u=u^*$, do the following and repeat for each extreme point.
(a) Connect a line from the origin to $\mathbf{E}(u^*)$ and add $\mathbf{E}_j$ on this line. Calculate the magnitude of $\mathbf{E}_j$ from Eq. (29).
(b) Connect a line from the origin to $\mathbf{F}(u^*)$ and add $\mathbf{F}_j$ on this line. Calculate the magnitude of $\mathbf{F}_j$ from Eq. (29).
(c) Locate $\mathbf{E}_j$ and $\mathbf{F}_j$ on the lines.
(d) Add the new point $(\mathbf{E}_j, \mathbf{F}_j)$ at the parameter value u^* to the list of points to be interpolated and go to step 3.
5. The image curve $\mathbf{Q}(u) = (\mathbf{E}(u), \mathbf{F}(u))$ defines the C^2 interpolating piecewise rational motion of degree 6 after the substitution into Eq. (3).

5 Examples

In this section, we present three examples—one for the C^1 interpolating motion of a spherical 2R robot arm and two for the C^2 interpolating motion of spherical 2R and 3R robot arms. These examples demonstrate the working of the algorithms presented before. Tables 1 and 3 give the values of the joint coordinates used in the examples for five orientations along with their parameter values for spherical 2R and 3R robot arms, respectively.

5.1 Spherical 2R Robot Arm. In the first example, we have five orientations for Cartesian rational motion planning of a spherical 2R robot arm (see Table 1 for input data). We use Algorithm 2 to interpolate $\mathbf{E}_i$ and $\mathbf{F}_i$ with two quadratic circular arcs $\mathbf{E}(u)$ and $\mathbf{F}(u)$, respectively. Table 2 gives the resulting control points of all the four quadratic Bézier segments of the two circular arcs and the associated weights. This gives us a C^1 continuous rational motion of degree four for the 2R robot arm.

We use Algorithm 4 to plan a C^2 rational motion for the same spherical 2R robot arm and the same set of five orientations. This is demonstrated in Figs. 3 and 4, which show the results of constraining the curve $\mathbf{F}(u)$ in the $s_3 s_4$ plane, with $\varepsilon = 0.002$. Note that in all the figures shown in this section, the symbol "■" represents first or the last two coordinates (i.e., $\mathbf{E}_i = (Q_{i1}, Q_{i2})$ or $\mathbf{F}_i = (Q_{i3}, Q_{i4})$) of the quaternions associated with the given five ori-

Table 2 Control points and associated weights of circular arcs *E* and *F*

	$\mathbf{E}(u)$		$\mathbf{F}(u)$	
i	$\mathbf{b}_i$	w_i	$\mathbf{b}_i$	w_i
0	(0.2588,0.0000)	1.0000	(0.0000,0.9659)	1.0000
1	(0.2588,0.0341)	0.9914	(0.2588,0.9659)	0.9659
2	(0.2500,0.0670)	1.0000	(0.4830,0.8365)	1.0000
3	(0.2412,0.0999)	0.9914	(0.7071,0.7071)	0.9659
4	(0.2241,0.1294)	1.0000	(0.8365,0.4830)	1.0000
5	(0.2100,0.1540)	0.7946	(0.9326,0.3166)	0.8674
6	(0.1908,0.1749)	0.6389	(0.9577,0.1261)	0.7822
7	(0.1521,0.2172)	0.3925	(0.9743,0.0000)	1.3106
8	(0.0990,0.2391)	0.2530	(0.9577,0.1261)	2.2340

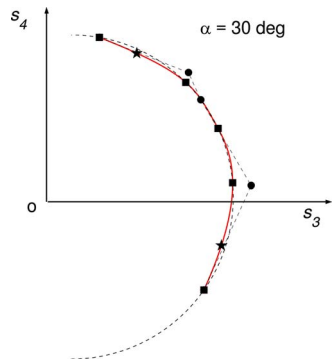

Fig. 3 An unconstrained C^2 cubic *B*-spline interpolation of a set of points in the $s_3 s_4$ plane

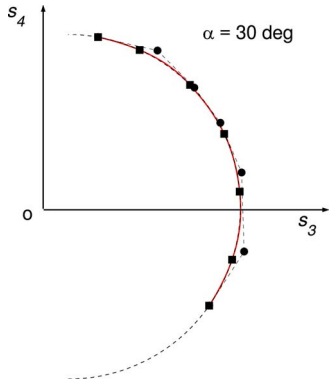

Fig. 4 A constrained C^2 cubic *B*-spline interpolation of a set of points in the s_3s_4 plane

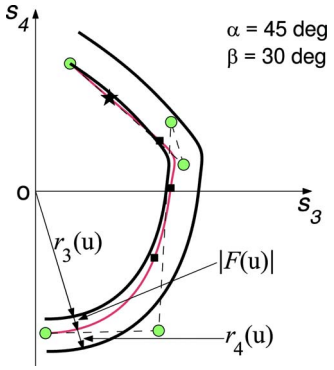

Fig. 6 An unconstrained C^2 cubic *B*-spline interpolation of a set of points in s_3s_4 plane; $r_3(u)=|w\cos((\alpha+\beta)/2)|$, $r_4(u)=|w\cos((\alpha-\beta)/2)|$, $w^2=Q_1^2(u)+Q_2^2(u)+Q_3^2(u)+Q_4^2(u)$

entations. These are the orientations to be interpolated. The symbol "●" represents the deboor control points of the resulting *B*-spline curve; the symbol "★" represents the extreme points of the *B*-spline curve. In first iteration, the algorithm finds two extreme points (Fig. 3), one each on the two Bézier segments. Thus, the algorithm adds two new points corresponding to the parameter value of the extreme points and in the next iteration re-interpolates seven (five originally given and two newly added) orientations. In the second iteration, the kinematic constraints are satisfied approximately (Fig. 4). The same procedure is repeated for the curve $\mathbf{E}(u)$ as well. In the end, we have a constrained curve $\mathbf{Q}(u)$ that satisfies the constraint of a spherical 2R robot arm approximately.

5.2 Spherical 3R Robot Arm. In the second example, we use five given orientations for motion planning of a spherical 3R robot arm (see Table 3 for input data). Figures 5–8 demonstrate Algorithm 3. In Figs. 5 and 7, the "wiggly" curves $r_1(u)$ and $r_2(u)$ depict the lower and the upper limits, respectively, of the inequality in Eq. (16), while in Figs. 6 and 8, curves $r_3(u)$ and $r_4(u)$ depict the lower and the upper limits, respectively of the inequality in Eq. (17). Figure 5 shows the unconstrained curve $\mathbf{E}(u)$ in the s_1s_2 plane and Fig. 6 shows the unconstrained curve $\mathbf{F}(u)$ in the s_3s_4 plane. The algorithm detects one extreme point outside the limits of the inequality (Eq. (21)) at $u^*=0.6241$. This extreme point is common to the curves $\mathbf{E}(u)$ and $\mathbf{F}(u)$. The algorithm adds a new point on the line joining origin and this extreme point in each plane and re-interpolates. We note here that the equivalence of the inequalities given by Eqs. (16) and (17) gives rise to a common extreme point for both the curves, but both the curves still need to be constructed and constrained independently since they each contribute only half of the quaternion coordinates of $\mathbf{Q}(u)$. Similarly, the new point can be generated only by combining the coordinates of point E_j and F_j from coordinate planes s_1s_2 and s_3s_4, respectively. The curve is constrained in the next iteration; see Fig. 7 and Fig. 8. The algorithm converges in two iterations for a value of $\delta=0.02$. Table 4 gives the deboor control orientations D_i of the constrained C_2 *B*-spline curve.

Table 3 Joint coordinates for spherical 3R robot arm motion planning

i	1	2	3	4	5
α(deg)			45		
β(deg)			30		
$\theta_i(deg)$	20	70	90	120	170
ϕ_i(deg)	0	30	40	50	80
ψ_i(deg)	10	30	40	60	90
u_i	0.0	0.2	0.5	0.7	1.0

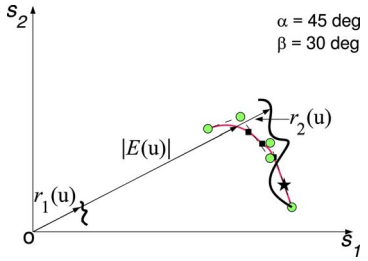

Fig. 5 An unconstrained C^2 cubic *B*-spline interpolation of a set of points in s_1s_2 plane; $r_1(u)=|w\sin((\alpha-\beta)/2)|$, $r_2(u)=|w\sin((\alpha+\beta)/2)|$, $w^2=Q_1^2(u)+Q_2^2(u)+Q_3^2(u)+Q_4^2(u)$

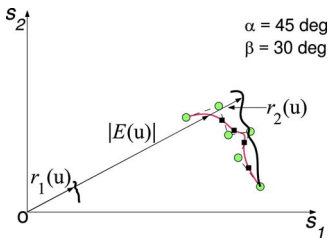

Fig. 7 A constrained C^2 cubic *B*-spline interpolation of a set of points in s_1s_2 plane; $r_1(u)=|w\sin((\alpha-\beta)/2)|$, $r_2(u)=|w\sin((\alpha+\beta)/2)|$, $w^2=Q_1^2(u)+Q_2^2(u)+Q_3^2(u)+Q_4^2(u)$

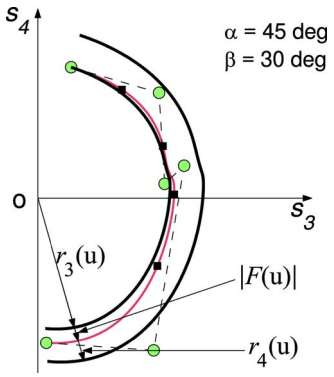

Fig. 8 A constrained C^2 cubic *B*-spline interpolation of a set of points in s_3s_4 plane; $r_3(u)=|w\cos((\alpha+\beta)/2)|$, $r_4(u)=|w\cos((\alpha-\beta)/2)|$, $w^2=Q_1^2(u)+Q_2^2(u)+Q_3^2(u)+Q_4^2(u)$

Table 4 Control points of *B*-spline curve $\mathbf{Q}(u)=(\mathbf{E}(u),\mathbf{F}(u))$

i	$\mathbf{Q}(u)=(\mathbf{E}(u),\mathbf{F}(u))$ $\mathbf{D}_i$
0	(0.6064, 0.0531, 0.2053, 0.7663)
1	(0.4951, 0.1167, 0.6587, 0.5544)
2	(0.5915, 0.2021, 0.7772, 0.0725)
3	(0.4953, 0.1791, 0.8327, 0.1707)
4	(0.3862,0.2059,0.5419,−0.7175)
5	(0.4112,0.2355,0.0559,−0.8788)

6 Conclusions

In this paper, we developed a method for synthesizing piecewise rational motions subject to the kinematic constraints of the spherical 2R and 3R robot arms. We presented several algorithms for synthesizing C^1 and C^2 continuous piecewise rational motions for spherical 2R and 3R arms. We showed that we could devise C^2 continuous piecewise rational motions for both 2R and 3R arms, although kinematic constraints are satisfied only approximately for 2R robot arms. However, kinematic constraints can be satisfied exactly for the C^1 rational motions of spherical 2R robot arms.

Acknowledgment

We would like to thank the anonymous reviewers for their valuable and insightful comments and suggestions in helping to improve this publication. The support of National Science Foundation under Grant No. DMI-0500064 is gratefully acknowledged.

Appendix: Frame Independent Kinematic Constraint for Spherical 3R Arm

In this Appendix, we show that kinematic constraint for spherical 3R robot arm derived earlier are frame independent. From spherical trigonometry, we have the following equation (see Fig. 9):

$$\cos(\gamma)=\cos(\beta)\cos(\alpha)-\sin(\beta)\sin(\alpha)\cos(\phi) \tag{A1}$$

Since the range of $\cos(\phi)$ is $[-1,1]$, Eq. (A1) reduces to

$$\cos(\beta+\alpha)\le\cos(\gamma)\le\cos(\beta-\alpha) \tag{A2}$$

After using double-angle formulas in trigonometry and performing algebra manipulation, inequality (A2) changes to

$$\tan^2((\beta-\alpha)/2)\le\tan^2(\gamma/2)\le\tan^2((\beta+\alpha)/2) \tag{A3}$$

Similarly, using double-angle formulas, Eq. (A1) can be changed to

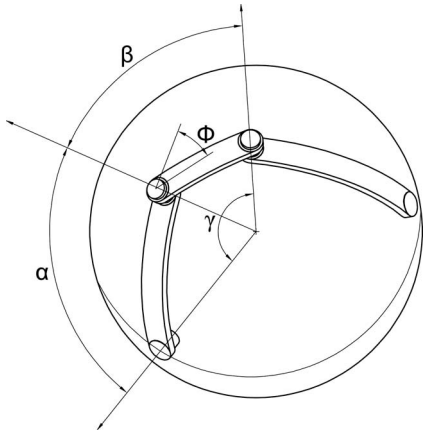

Fig. 9 A Spherical 3R robot arm

$$\sin^2(\gamma/2)=-\tfrac{1}{2}\cos(\beta)\cos(\alpha)+\tfrac{1}{2}\sin(\beta)\sin(\alpha)\cos(\phi)+\tfrac{1}{2} \tag{A4}$$

and

$$\cos^2(\gamma/2)=\tfrac{1}{2}\cos(\beta)\cos(\alpha)-\tfrac{1}{2}\sin(\beta)\sin(\alpha)\cos(\phi)+\tfrac{1}{2} \tag{A5}$$

We can expand the first equation of Eq. (14) to

$$q_1^2+q_2^2=\cos(\phi)\sin(\beta)\sin(\alpha)/2-\cos(\beta)\cos(\alpha)/2+1/2 \tag{A6}$$

Similarly, we can expand the second equation of Eq. (14) to

$$q_3^2+q_4^2=\cos(\beta)\cos(\alpha)/2-\cos(\phi)\sin(\beta)\sin(\alpha)/2+1/2 \tag{A7}$$

From Eqs. (A4)–(A7), we find the following relationship:

$$\tan^2(\gamma/2)=\frac{\sin^2(\gamma/2)}{\cos^2(\gamma/2)}=\frac{q_1^2+q_2^2}{q_3^2+q_4^2} \tag{A8}$$

Equations (A3) and (A8) together constitute the kinematic constraint of a spherical 3R robot arm.

References

[1] Shoemake, K., 1985, "Animating Rotation With Quaternion Curves," SIGGRAPH Computer Graphics, **19**(3), pp. 245–254.
[2] Pletinckx, D., 1989, "Quaternion Calculus as a Basic Tool in Computer Graphics," Visual Comput., **5**, pp. 2–13.
[3] Dam, E. B., Koch, M., and Lillholm, M., 1998. "Quaternions, Interpolation and Animation," University of Copenhagen, Technical Report No. DIKU-TR-98/5.
[4] Duff, T., 1985, "Quaternion Splines for Animating Orientation," Proceedings of the USENIX Association Second Computer Graphics Workshop, Monterey CA, pp. 54–62.
[5] Kim, M.-J., Kim, M.-S., and Shin, S. Y., 1995, "A C^2 Continuous *B*-Spline Quaternion Curve Interpolating a Given Sequence of Solid Orientations," *Proceedings of the Computer Animation*, IEEE Computer Society, Washington, DC, pp. 19–21.
[6] Kim, M. S., and Nam, K. W., 1995, "Interpolating Solid Orientations With Circular Blending Quaternion Curves," Comput.-Aided Des., **27**(5), pp. 385–398.
[7] Nielson, G., 1993, "Smooth Interpolation of Orientations," *Computer Animation, Models and Techniques in Computer Animation*, Springer, New York, pp. 75–93.
[8] Nielson, G. M., 2004, "Nu-Quaternion Splines for the Smooth Interpolation of Orientations," IEEE Trans. Vis. Comput. Graph., **10**(2), pp. 224–229.
[9] Wang, W., and Joe, B., 1993, "Orientation Interpolation in Quaternion Space Using Spherical Biarcs," *Proceedings of the Graphics Interface'93*, Morgan-Kaufmann, San Francisco, CA, pp. 24–32.
[10] Barr, A. H., Currin, B., Gabriel, S., and Hughes, J. F., 1992, "Smooth Interpolation of Orientations With Angular Velocity Constraints Using Quaternions," Comput. Graph., **26**(2), pp. 313–320.
[11] Ge, Q. J., and Ravani, B., 1994, "Computer-Aided Geometric Design of Motion Interpolants," ASME J. Mech. Des., **116**(3), pp. 756–762.
[12] Ge, Q. J., and Ravani, B., 1994, "Geometric Construction of Bezier Motions," ASME J. Mech. Des., **116**(3), pp. 749–755.
[13] Juttler, B., and Wagner, M. G., 1996, "Computer-Aided Design With Spatial Rational *b*-Spline Motions," ASME J. Mech. Des., **118**(2), pp. 193–201.
[14] Purwar, A., and Ge, Q. J., 2005, "On the Effect of Dual Weights in Computer Aided Design of Rational Motions," ASME J. Mech. Des., **127**(5), pp. 967–972.
[15] Purwar, A., Chi, X., and Ge, Q. J., 2008, "Automatic Fairing of Two-Parameter Rational *b*-Spline Motion," ASME J. Mech. Des., **130**(1), p. 011003.
[16] Röschel, O., 1998, "Rational Motion Design: A Survey," Comput.-Aided Des., **30**(3), pp. 169–178.
[17] Horsch, T., and Juttler, B., 1998, "Cartesian Spline Interpolation for Industrial Robots," Comput.-Aided Des., **30**(3), pp. 217–224.
[18] Wagner, M., and Ravani, B., 1996, "Computer Aided Design of Robot Trajectories Using Rational Motions," *Recent Advances in Robot Kinematics*, J. Lenarcic and V. Parenti- Castelli, eds., Kluwer, Dordrecht, pp. 151–158.
[19] Jin, Z., and Ge, Q. J., 2007, "Computer Aided Synthesis of Piecewise Rational Motion for Planar 2*r* and 3*r* Robot Arms," ASME J. Mech. Des., **129**(10), pp. 1031–1036.
[20] Bottema, O., and Roth, B., 1979, *Theoretical Kinematics*, North-Holland, Amsterdam.

[21] McCarthy, J. M., 1990, *Introduction to Theoretical Kinematics*, MIT, Cambridge, MA.
[22] Ravani, B., and Roth, B., 1984, "Mappings of Spatial Kinematics," ASME J. Mech., Transm., Autom. Des., **106**(3), pp. 341–347.
[23] Farin, G., 1996, *Curves and Surfaces for Computer-Aided Geometric Design: A Practical Guide*, 4th ed., Academic, New York.
[24] Hoschek, J., and Lasser, D., 1993, *Fundamentals of Computer Aided Geometric Design*, A. K. Peters, Wellesley, MA.
[25] Piegl, L., and Tiller, W., 1995, *The Nurbs Book*, Springer, Berlin.
[26] Ge, Q. J., and Purwar, A., 2004, "Spears, Oriented Screw Displacements, and Their Image Spaces," Proceedings of the 11th World Congress in Mechanism And Machine Science IFToMM 2004, Aug. 18-21, Tianjin, China.
[27] Bangert, C., and Prautzsch, H., 1997, "Circle and Sphere as Rational Splines," Neural, Parallel and Scientific Computations, **5**(1-2), pp. 153–162.