Computer–Aided Design with Spatial Rational B–Spline Motions

Bert Jüttler (University of Dundee, Scotland)
Michael G. Wagner (University of California, Davis)

December 8, 1995

Abstract

Using rational motions it is possible to apply many fundamental B–spline techniques to the design of motions. The present paper summarizes the basic theory of rational motions and introduces a linear control structure for piecewise rational motions suitable for geometry processing. Moreover it provides algorithms for the calculation of the surface which is swept out by a moving polyhedron and examines interpolation techniques. The methods presented in this paper can be applied to various problems in Computer Animation as well as in Robotics.

Keywords. Rational motions, NURBS–curves and surfaces, motion interpolation.

Introduction

During last years, it has been realized that the methods of Computer Aided Geometric Design (CAGD) provide elegant tools for various tasks in Computer Graphics, Robotics and Kinematics, especially for the design of rigid body motions. One of the first contributions to this research area was the spherical generalization of the de Casteljau algorithm introduced by Shoemake[28] in order to interpolate the orientations of a moving object, cf. also [19, 23]. More recently Ge and Ravani[8] and Park and Ravani[22] presented methods for constructing so-called Bézier motions by generalizing the subdivision algorithm of Bernstein–Bézier curves. Yet these techniques have one important disadvantage. The resulting motion splines are non–rational which leads to computationally expensive algorithms. On the other hand the trajectories of points of the moving object are transcendental curves. Of course, it is possible to compute the trajectories pointwise, but in general there exists no rational parametric representations of these curves.
In contrast to this, the methods of Computer Aided Geometric Design are mostly based on parametric representations of curves and surfaces, especially on (polynomial or rational) B–spline representations, cf. [11]. In order to apply CAGD–methods to kinematical problems, it therefore seems to be

essential to restrict to motions with only rational point trajectories. Such motions, called rational motions, have been subject of investigation in theoretical kinematic for about one century, cf. [5]. The first who applied rational motions to motion design were Ge and Ravani[7]. Their interpolation algorithm is based on rational dual quaternion curves not satisfying Plücker's condition. Another contribution has recently been given by Johnstone[12] who used normalized rational quaternion curves in order to interpolate orientations of a moving object for animation. Nevertheless neither of these approaches takes advantages of the geometric properties of the resulting motion splines.
The present paper tries to fill this gap by introducing spatial rational B–spline motions. It discusses geometric properties, computation and design of such motions. All point trajectories of the presented motion splines are rational B–spline (NURBS) curves. Moreover, many surfaces generated by a rational B–spline motion, such as sweeping surfaces or envelopes of moving planes, cylinders or cones, are rational tensor product B–spline surfaces.
The paper is organized as follows. In the first two sections we present the basic theory of rational B–spline motions which is based on a linear control structure analogously to the control polygon of a NURBS curve. In the third section we examine the motion of planes and present algorithms for the exact computation of the enveloping surface of a moving polyhedron. The fourth section finally deals with different techniques for the design of spatial rational B–spline motions, covering algorithms for motion interpolation and the optimization of rational motions under energetic constraints.

1. Preliminaries

In this section we briefly summarize some fundamentals of spatial Kinematics (see [2]), piecewise rational motions, and of the theory of univariate B-spline functions (see also [6, 11, 27]).

1.1 Spatial kinematics

Consider two coinciding real Euclidean 3–spaces, the *fixed space* E^3 and the *moving space* $\widehat{\mathrm{E}}^3$. Both spaces are associated with right-handed Cartesian coordinate frames. In order to simplify notations, points in E^3, $\widehat{\mathrm{E}}^3$ will be described with help of *homogeneous Cartesian coordinate vectors* $\mathbf{p} = (\, p_0 \; p_1 \; p_2 \; p_3 \,)^\top$, $\widehat{\mathbf{p}}$, respectively.
The space $\widehat{\mathrm{E}}^3$ is assumed to perform an Euclidean motion with respect to E^3. Each *position* of $\widehat{\mathrm{E}}^3$ can be described with help of a real 4×4–matrix

$$\mathrm{M} = \left(\begin{array}{c|ccc} v_0 & 0 & 0 & 0 \\ \hline v_1 & & & \\ v_2 & & \mathrm{R} & \\ v_3 & & & \end{array} \right) \tag{1}$$

where the 3×3–sub-matrix $\mathrm{R} = (r_{i,j})_{i,j=1,2,3}$ satisfies the orthogonality conditions

$$\mathrm{R} \cdot \mathrm{R}^{\top} = v_0^2 \mathrm{I} \quad \text{and} \quad \det(\mathrm{R}) > 0. \tag{2}$$

The letter I denotes the identity matrix. The *position* $\mathbf{p} \in \mathrm{E}^3$ of a point $\widehat{\mathbf{p}}$ of the moving space $\widehat{\mathrm{E}}^3$ results from a linear transformation

$$\widehat{\mathbf{p}} \mapsto \mathbf{p} = \mathrm{M} \cdot \widehat{\mathbf{p}} \tag{3}$$

of homogeneous coordinate vectors. The matrix M represents a *spatial displacement*, the component v_0 is called the *weight* of the displacement. Note that the homogeneous coordinate vector $\mathbf{u} = (\, v_0 \; v_1 \; v_2 \; v_3 \,)^{\top}$ describes the position of the origin of the coordinate frame of the moving space $\widehat{\mathrm{E}}^3$ whereas R defines its orientation.
A continuous one-parameter set of positions of $\widehat{\mathrm{E}}^3$ defines a *motion* $\mathrm{M} = \mathrm{M}(t)$ where the parameter t is assumed to be the time. The curve

$$\mathbf{p}(t) = \mathrm{M}(t) \cdot \widehat{\mathbf{p}} \tag{4}$$

is called the *trajectory* of the point $\widehat{\mathbf{p}} \in \widehat{\mathrm{E}}^3$.

1.2 Piecewise rational motions

If there exists a representation $\mathrm{M}(t)$ of the motion M such that *all* element functions of the matrix $\mathrm{M}(t)$ are (piecewise) polynomials of maximal degree k, M is called a *(piecewise) rational motion* of degree k. Then, all trajectories are (piecewise) rational curves of maximal degree k. The study of rational motions dates back to Darboux[5] who examined the quadratic case thoroughly in 1895. Detailed geometrical discussions of rational motions of degree $n = 3$ and $n = 4$ were given by Wunderlich[32] and Röschel[26].
According to [13, 15] any piecewise rational motion $\mathrm{M}(t)$ of degree k possesses a representation

$$\mathrm{M}(t) = \left(\begin{array}{c|ccc} \overline{v}_0(d_0^2 + d_1^2 + d_2^2 + d_3^2) & 0 & 0 & 0 \\ \hline v_1 & & & \\ v_2 & & \overline{v}_0\,\mathrm{D} & \\ v_3 & & & \end{array}\right) \tag{5}$$

with

$$\mathrm{D} = \mathrm{D}(t) = \begin{pmatrix} d_0^2 + d_1^2 - d_2^2 - d_3^2 & 2(d_1 d_2 - d_0 d_3) & 2(d_1 d_3 + d_0 d_2) \\ 2(d_1 d_2 + d_0 d_3) & d_0^2 - d_1^2 + d_2^2 - d_3^2 & 2(d_2 d_3 - d_0 d_1) \\ 2(d_1 d_3 - d_0 d_2) & 2(d_2 d_3 + d_0 d_1) & d_0^2 - d_1^2 - d_2^2 + d_3^2 \end{pmatrix}. \tag{6}$$

Here, the 8 parameters $\overline{v}_0(t)$, $v_1(t), \ldots, v_3(t)$, and $d_0(t), \ldots, d_3(t)$ are piecewise polynomials of maximal degree $k - 2l$, k, and l, respectively, where the number l satisfies $0 \le 2l \le k$. The four parameters $d_0, \ldots, d_3$ are *Euler's parameters* of the rotational part $\mathrm{D}(t)$ of the motion (6), which are well known in Kinematics, see [2]. In the sequel we will collect these parameters in a vector

$\widetilde{d} = (\, d_0\ d_1\ d_2\ d_3)^\top$ from $\mathbb{R}^4$.
Consider the first column

$$\mathbf{u}(t) = \begin{pmatrix} \overline{v}_0(t)\|\widetilde{d}(t)\|^2 \\ v_1(t) \\ v_2(t) \\ v_3(t) \end{pmatrix} \tag{7}$$

of the matrix (5) with $\|\widetilde{d}(t)\|^2 = d_0(t)^2 + d_1(t)^2 + d_2(t)^2 + d_3(t)^2$. The piecewise rational curve, which is described by the homogeneous coordinate vector $\mathbf{u}(t)$, is the trajectory of the origin of the moving space $\widehat{\mathrm{E}}^3$. On the other hand, the 3×3–matrix $\mathrm{D}(t)$ represents the rotational part of the motion $\mathrm{M}(t)$ defining the orientation of the moving space. For any instant t, the moving space results from the fixed space by a translation composed with a rotation. The translation maps the origin of the moving space to the point $\mathbf{u}(t)$ from (7), whereas the angle φ of the rotation and the normalized direction vector $\vec{\mathbf{a}}$ of the axis result from

$$\begin{aligned} d_0 &= \|\widetilde{d}\| \cos\frac{\varphi}{2} \qquad \text{and} \\ (\, d_1\ d_2\ d_3\,)^\top &= \|\widetilde{d}\| \sin\frac{\varphi}{2}\vec{\mathbf{a}} \qquad (\vec{\mathbf{a}}^\top\vec{\mathbf{a}} = 1). \end{aligned} \tag{8}$$

The vector $\widetilde{d}(t)$ formed by Euler's parameters can be identified with the quaternion which corresponds to the rotation (6). Quaternion calculus is one of the basic tools in theoretical kinematics (see [2]) since the composition of rotations can be expressed with help of quaternion multiplication.

1.3 B–spline functions

Let k be a positive integer, and let

$$T = (t_0, t_1, t_2, \ldots, t_{m+k+1}) \tag{9}$$

be a nondecreasing sequence of $m+k+2$ real numbers with $t_i < t_{i+k+1}$ for $i = 0, \ldots, m$. The $m+1$ associated B–spline basis functions, denoted by $N^k_{i.T}(t)$, are defined by the recurrence relation

$$N^q_{i.T}(t) = \omega^q_{i.T}(t) N^{q-1}_{i.T}(t) + (1 - \omega^q_{i+1.T}(t)) N^{q-1}_{i+1.T}(t) \qquad (q = 1, \ldots, k) \tag{10}$$

where

$$\omega^q_{i.T}(t) = \begin{cases} \dfrac{t - t_i}{t_{i+q} - t_i} & \text{if } t_i < t_{i+q} \\ 0 & \text{otherwise,} \end{cases} \tag{11}$$

and

$$N^0_{i.T}(t) = \begin{cases} 1 & \text{if } t_i \le t < t_{i+1} \\ 0 & \text{otherwise.} \end{cases} \tag{12}$$

The vector T and the number k are called the *knot vector* and the *degree* of the B–spline basis functions $N^k_{i.T}(t)$, respectively. The $N^k_{i.T}(t)$ obviously are piecewise polynomials of degree k, which

are $(k - v_i)$–times continuously differentiable at $t = t_i$, where v_i denotes the multiplicity of the knot t_i in the knot vector T. The basic tool for handling B–spline functions is the knot insertion algorithm providing possibilities for evaluation and subdivision. For details we refer the reader to [6, 11, 27].

2. Rational B–spline motions and their control structure

Starting with an algorithm for the construction of piecewise rational motions, this section introduces a linear control structure for piecewise rational motion splines which is an analogon to the control polygon of a NURBS curve.

2.1 Construction of piecewise rational motions

In the following we construct a piecewise rational motion $\mathrm{M}(t)$ of degree k by choosing the parameters $\overline{v}_0(t)$, $\vec{\mathbf{v}}(t) = (\, v_1(t)\ v_2(t)\ v_3(t)\,)^\top$, and $\widetilde{d}(t) = (\, d_0(t)\ \ldots\ d_3(t)\,)^\top$ from equation (5) as (vector–valued) B–spline functions of degree $k - 2l$, k, and l, respectively, defined over certain knot vectors (l fixed with $0 \le 2l \le k$). We will derive a representation of the motion (5) as a matrix–valued B–spline function

$$\mathrm{M}(t) = \sum_{i=0}^{m} N_{i,T}^{k}(t)\, \mathrm{A}_i \tag{13}$$

with $m+1$ constant coefficient matrices A_i. According to (5) we therefore have to compute certain products of the B–spline functions $\overline{v}_0(t)$, $v_1(t), v_2(t), v_3(t)$, and $d_0(t), \ldots, d_3(t)$, which can be done by using product formulae for B–spline basis functions (see [18]). Since these formulae turn out to be relatively complex we propose a different technique based on knot insertion and knot removal. Consider the knot set

$$\mathcal{K} = \{t_0, t_1, \ldots, t_s\} \subset \mathbb{R} \tag{14}$$

which is obtained as the union of the knot sets of the three B–spline functions $\overline{v}_0(t)$, $\vec{\mathbf{v}}(t)$, and $\widetilde{d}(t)$. Note that each knot t_i appears only once. The number $s+1$ denotes the cardinality of this set. We moreover assume that the knots t_i are ordered such that $t_i < t_{i+1}$ $(i = 0, \ldots, s-1)$.

Algorithm 1.

1. Split the B–spline functions $\overline{v}_0(t)$, $\vec{\mathbf{v}}(t)$, and $\widetilde{d}(t)$ into s polynomial segments by applying Böhm's algorithm, see [6, 11]. The jth segment is defined over the interval $[t_j, t_{j+1}]$ $(j = 0, \ldots, s-1)$.
2. `For each segment:`
   Compute a B–spline representation for the motion segment by inserting the corresponding polynomial segments of $\overline{v}_0(t)$, $\vec{\mathbf{v}}(t)$, and $\widetilde{d}(t)$ into the representation formula (5). The details of this computation can be found in the Appendix of this paper.

3. Collect the segments (44) in order to obtain a B–spline representation of the whole motion (13). This representation is defined over the knot vector

$$T = (\underbrace{t_0, \ldots, t_0}_{k+1 \text{ times}}, \quad \underbrace{t_1, \ldots, t_1}_{k+1 \text{ times}}, \quad \ldots, \quad \underbrace{t_s, \ldots, t_s}_{k+1 \text{ times}}). \tag{15}$$

The constant matrices A_p ($p = 0, \ldots, m$ with $m = (k+1)s - 1$) result from

$$\mathrm{A}_{(k+1)j+i} = \mathrm{A}_i^{(j)} \qquad (i = 0, \ldots, k;\ j = 0, \ldots, s-1) \tag{16}$$

where $\mathrm{A}_i^{(j)}$ denotes the ith coefficient matrix of the jth motion segment.

4. Remove unnecessary knots from the knot vector (15).

The number of unnecessary knots which can be removed in step 4 depends on the order of differentiability of the components. If the functions $\overline{v}_0(t)$, $\vec{\mathbf{v}}(t)$, and $\widetilde{d}(t)$ are r–times continuously differentiable at instant $t = t_i$, the knot t_i can be removed $(r+1)$ times from the knot vector (15). Algorithms for knot elimination are described in [9]. Note that this removal reduces the number of coefficient matrices A_j of (13). The result is a *rational B–spline motion* defined over a minimal knot vector.

2.2 The control structure of a rational B–spline motion

Let us now consider a rational B–spline motion $\mathrm{M}(t)$ of type (13). Obviously, the $m+1$ constant coefficient matrices have the form

$$\mathrm{A}_i = \left(\begin{array}{c|ccc} w_{i,0} & 0 & 0 & 0 \\ \hline w_{i,1} & & & \\ w_{i,2} & & \mathrm{S}_i & \\ w_{i,3} & & & \end{array}\right). \tag{17}$$

In general, the $m+1$ sub–matrices S_i do not fulfill the orthogonality condition (2). Thus, the matrices A_i describe some *affine* mappings $\widehat{\mathrm{E}}^3 \to \mathrm{E}^3$.

Let us further consider an object $\mathcal{O} \subset \widehat{\mathrm{E}}^3$ in the moving space. The images $\mathrm{A}_i \cdot \mathcal{O} \subset \mathrm{E}^3$ of this object under the affine mappings A_i will be called the *control positions* of the rational B–spline motion (13) of the object $\mathcal{O}$. These control positions are distorted copies of $\mathcal{O}$ since they are formed by the images of all points of the object $\mathcal{O}$ under the affine mappings A_i.

The set of all control positions is called the *control structure* of the rational B–spline motion (13) of the moving object. As a first example, Figure 1 shows a rational B–spline motion of degree 4 illustrated by the control positions of a moving cube and by some uniformly distributed positions of the moving unit cube.

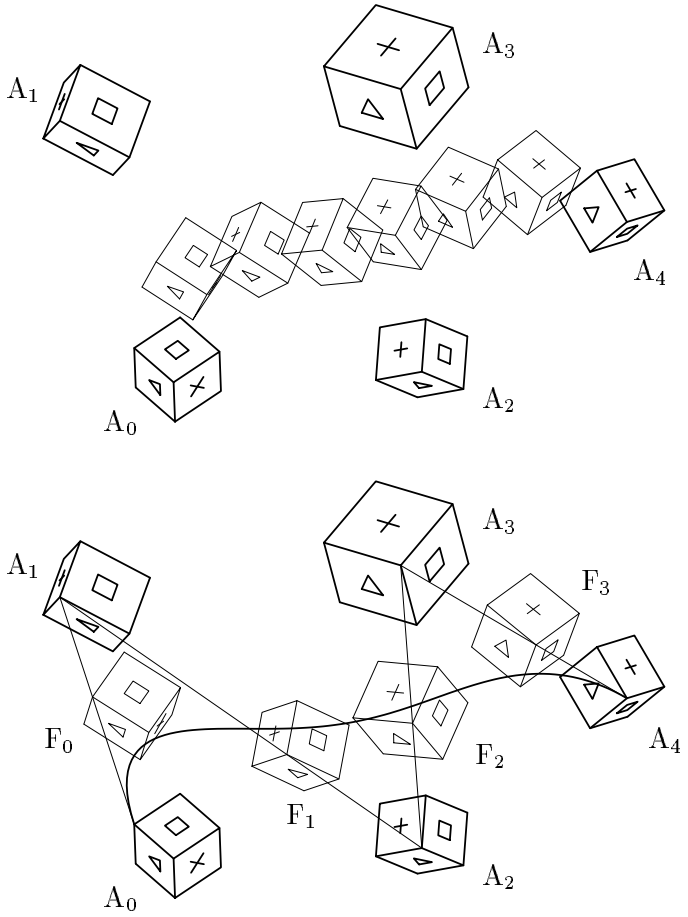

Figure 1: Control structure of a moving cube. The control positions and weight positions are described by the affine displacements A_j and F_i, respectively.

The trajectory $\mathbf{p}(t)$ of a point $\widehat{\mathbf{p}} \in \mathcal{O} \subset \widehat{\mathrm{E}}^3$ is the non-uniform rational B–spline (NURBS) curve

$$\mathbf{p}(t) = \mathrm{M}(t) \cdot \widehat{\mathbf{p}} = \sum_{i=0}^{m} N_{i,T}^{k}(t) \mathrm{A}_i \cdot \widehat{\mathbf{p}} \tag{18}$$

of degree k. The affine control positions

$$\mathbf{p}_i = (\, p_{i.0} \; p_{i.1} \; p_{i.2} \; p_{i.3} \,)^{\mathsf{T}} = \mathrm{A}_i \cdot \widehat{\mathbf{p}} \tag{19}$$

of the moving point $\widehat{\mathbf{p}}$ are the *control points* of this NURBS–curve, whereas the components $p_{i.0}$ are the so-called *weights* of the control points. According to Farin (cf. [6]) the weights of a NURBS–curve can be expressed by auxiliary *weight points* $\mathbf{f}_i = \mathbf{p}_i + \mathbf{p}_{i+1}$, $i = 0, \ldots, m-1$ (sometimes also called *Farin points* or *frame points*). The weight points divide the edges of the control polygon by the ratio of the neighbouring weights. Analogously one can introduce *weight positions* $\mathrm{F}_i = \mathrm{A}_i + \mathrm{A}_{i+1}$ of a rational B–spline motion. The ith weight position of the moving object $\mathcal{O} \subset \widehat{\mathrm{E}}^3$ is formed by

the ith weight points of its points.
The use of rational B–spline motions (13) allows to apply the whole variety of B–spline techniques to problems in motion design. This close relationship to NURBS–curves let rational motion splines fit well into a standard CAD–package.

Remarks.

1. Note that for arbitrary affine displacements A_j, the motion defined by (13) in general will *not* be a Euclidean motion. Thus, it describes a so–called *affine motion* where the displacements $M(t)$ are affine displacements and the moving object is also subject to a certain distortion.
2. In case of planar motions, the control structure of a rational B–spline motion is formed by so–called *equiform* displacements. Here, the affine control positions of a moving object are similar to the object. A detailed discussion of planar B–spline motions can be found in [30].
3. The Euclidean spatial displacements (1) can be mapped to the points of a certain 6–dimensional manifold Φ of 12–dimensional affine space A^{12}, where the homogeneous coordinates of these points are the components of the matrix (1). Of course a NURBS–curve on the manifold Φ corresponds to a rational B–spline motion and vice versa. The power of this simple mapping lies in the fact that the trajectory of any point $\widehat{\mathbf{p}} \in \widehat{E}^3$ can be interpreted as the image of the curve on Φ under a linear mapping from A^{12} to E^3, that depends on $\widehat{\mathbf{p}}$ only. This linear mapping gives a geometric interpretation of the control structure of a rational B–spline motion as the preimage of the control structure of the curve on Φ. Similar to the mapping used in [30] for planar motions, it can be used for the design of rational B–spline motions and the shape discussion of their trajectories. For detailed information the reader is referred to [25].

2.3 Convex hull property and collision detection

If all control weights of the rational B–spline motion $M(t)$ are positive, i.e. $w_{i,0} > 0$, every trajectory fulfills the convex hull property for NURBS curves with respect to its control polygon (cf. [11]). Consequently $M(t)$ fulfills the convex hull property with respect to its control structure, i.e., the region that is traced out by a moving object lies in the sum of the convex hulls of $k+1$ neighbouring control positions at a time. If additional knots are inserted or $M(t)$ is subdivided into two or more motion segments by applying the knot insertion algorithm, the sum of these convex hulls converges to the region traced out by the moving object (cf. Fig. 2).
Moreover, an arbitrary obstacle which does not intersect the convex hull of the control structure cannot collide with the moving object. This can be used for simple and efficient algorithms for the detection of collision free motions.
Experimental results show that the motions obtained by the interpolation scheme presented in this paper usually have positive weights. However, if some of the weights are negative, the usual convex hull property is not valid. Nevertheless the corresponding motion fulfills a *projective version* of this property, which takes advantage of the weight positions, analogously to the projective variation diminishing property for rational Bézier curves of Pottmann, see [24].

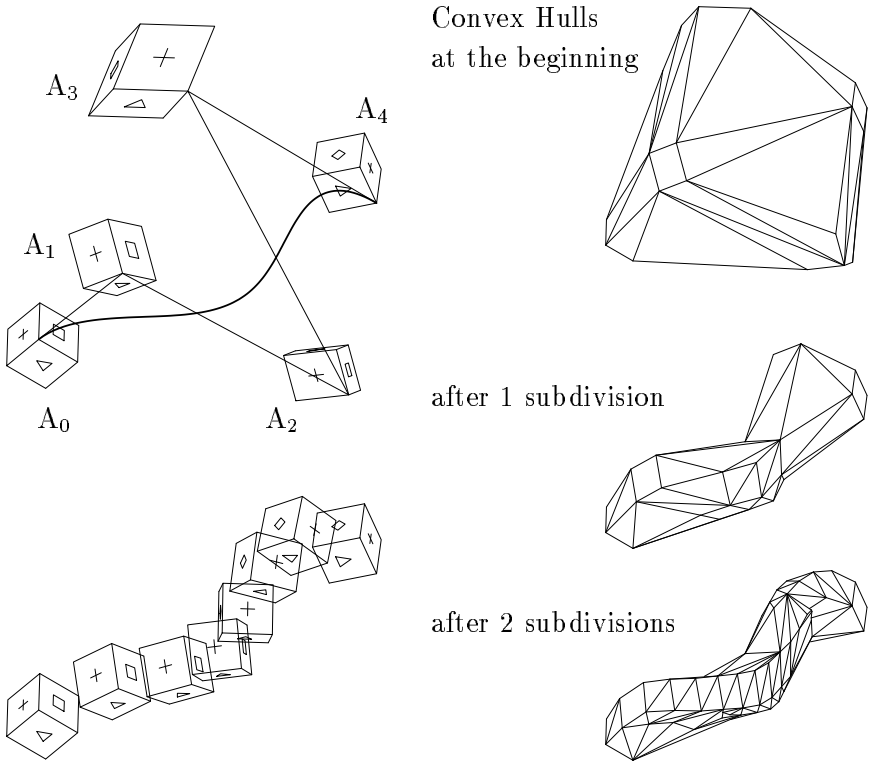

Figure 2: Control structure of a rational B–spline motion of order 5 with one segment. The sum of the convex hulls converges to the area that is traced out by the moving cube.

3. A moving polyhedron

In this section we will compute the region which is traced out by a moving polyhedron. This region is of interest for many applications, such as collision tests or the control of milling machines. Generally, its boundary surface consists of different segment types. The surface segments are either parts of the ruled surfaces which are generated by the edges of the polyhedron, or they are segments of the developable surfaces which are enveloped by the faces of the moving object, see Figure 4.

3.1 The dual representation of a rational B–spline motion

In order to compute the developable surface segment which is enveloped by a face of a moving polyhedron we will introduce the dual form of a spatial rational B–spline motion. Consider the set of all points $\mathbf{p} = (\, p_0 \; p_1 \; p_2 \; p_3 \,)^\top \in \mathrm{E}^3$ whose homogeneous coordinate vectors satisfy the linear equation

$$P_0 p_0 + P_1 p_1 + P_2 p_2 + P_3 p_3 = \mathbf{P}^\top \mathbf{p} = 0. \tag{20}$$

Obviously this set forms a plane. In the sequel we will collect the coefficients of (20) in a vector $\mathbf{P} = (\, P_0 \; P_1 \; P_2 \; P_3 \,)^\top \in \mathbb{R}^4 \setminus \{\mathbf{0}\}$ called the *homogeneous coordinate vector* of this plane (see e.g. [4]). A point $\mathbf{p} \in \mathrm{E}^3$ lies on the plane $\mathbf{P}$ iff $\mathbf{P}^\top \mathbf{p} = 0$ holds.
Further consider a spatial displacement $\mathrm{M} : \widehat{\mathrm{E}}^3 \to \mathrm{E}^3 : \widehat{\mathbf{p}} \mapsto \mathbf{p} = \mathrm{M} \cdot \widehat{\mathbf{p}}$. Obviously, this displacement maps the planes of the moving space $\widehat{\mathrm{E}}^3$ to those of the fixed space E^3. We immediately obtain that the image of a plane $\widehat{\mathbf{P}}$ in $\widehat{\mathrm{E}}^3$ results from the linear transformation

$$\widehat{\mathbf{P}} \mapsto \mathbf{P} = \mathrm{M}^* \cdot \widehat{\mathbf{P}} \tag{21}$$

where the real 4×4–matrix M^* is given by $\mathrm{M}^* = \lambda(\mathrm{M}^{-1})^\top$. The real factor $\lambda \neq 0$ can be chosen arbitrarily. The mapping (21) is called the *dual representation* of the given spatial displacement M. Owing the relationship from (1) we get

$$\mathrm{M}^* = \left(\begin{array}{c|c} v_0^2 & -\vec{\mathbf{v}}^\top \mathrm{R} \\ \hline 0 & \\ 0 & v_0 \mathrm{R} \\ 0 & \end{array}\right) \qquad \text{with } \vec{\mathbf{v}} = \begin{pmatrix} v_1 \\ v_2 \\ v_3 \end{pmatrix}. \tag{22}$$

Let us now consider a rational B–spline motion $\mathrm{M}(t)$ (see (13)) which has been constructed with help of the formula (5) by choosing the parameters $\overline{v}_0(t)$, $v_1(t), \ldots, v_3(t)$, and $d_0(t), \ldots, d_3(t)$ as B–spline functions. Using (22) we can compute the dual representation of the given rational B–spline motion motion:

$$\mathrm{M}^*(t) = \overline{v}_0 \left(\begin{array}{c|c} \overline{v}_0 \, (d_0^2 + d_1^2 + d_2^2 + d_3^2)^2 & -\vec{\mathbf{v}}^\top \mathrm{D} \\ \hline 0 & \\ 0 & (d_0^2 + d_1^2 + d_2^2 + d_3^2)\, \mathrm{D} \\ 0 & \end{array}\right). \tag{23}$$

The real 3×3–matrix $\mathrm{D}(t)$ has been introduced in (6). Thus, the dual form of a rational B–spline motion can be described by a matrix–valued B–spline function $\mathrm{M}^*(t)$. Analogously to the calculation of the control positions in Section 2.1 we can find a representation similar to (13), but the order and the knot vector of the dual form $\mathrm{M}^*(t)$ are different from those of the motion $\mathrm{M}(t)$. Again, the constant coefficient matrices of the matrix–valued B–spline function $\mathrm{M}^*(t)$ can be used in order to define a control structure. In this paper we do not discuss this structure, we restrict ourself to the computation of the developable surface segment which is enveloped by a face of a moving polyhedron. A more detailed discussion of the dual representation of a rational B–spline motion can be found in [29].

3.2 The surface which is enveloped by a moving polyhedron

We now consider a polyhedron in the moving space $\widehat{\mathrm{E}}^3$. At first we compute the segments of developable surfaces which are enveloped by one fixed face of the moving polyhedron. The corners of

this face are denoted by $\widehat{\mathbf{p}}_0, \ldots, \widehat{\mathbf{p}}_f$, where $f+1$ is the number of the corners. The 0th coordinates of the corner points are assumed to be equal to 1. The letter $\widehat{\mathbf{F}}$ stands for the homogeneous coordinate vector of the plane which contains the face $\widehat{\mathbf{p}}_0, \ldots, \widehat{\mathbf{p}}_f$. Then, the parametric representation

$$\mathbf{F}(t) = \mathrm{M}^*(t) \cdot \widehat{\mathbf{F}} \tag{24}$$

describes the motion of the plane $\widehat{\mathbf{F}} \in \widehat{\mathrm{E}}^3$. The enveloping surface of this moving plane is well known to be a developable ruled surface, see [3, p. 195]. Generally, such a surface is a tangent surface which is formed by the tangents of a twisted curve, the so-called *line of regression*. Its rulings are the lines in which the two planes $\mathbf{F}(t) = \mathrm{M}^*(t) \cdot \widehat{\mathbf{F}}$ and $\frac{d}{dt}\mathbf{F}(t) = \dot{\mathbf{F}}(t) = \dot{\mathrm{M}}^*(t) \cdot \widehat{\mathbf{F}}$ intersect. At any instant t, the moving plane $\mathbf{F}(t)$ and the developable surface have contact along the instantaneous ruling, i.e., along the intersecting line of the plane $\mathbf{F}(t)$ with the *derivative plane* $\dot{\mathbf{F}}(t)$, see Figure 3.

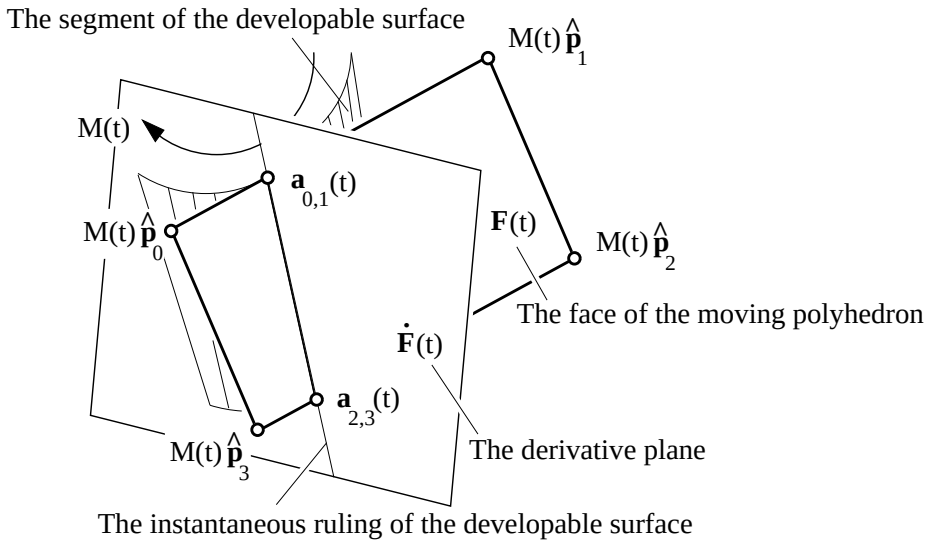

Figure 3: The generation of a developable surface by a moving plane.

Recently, the application of the dual form of twisted curves to the design of developing surfaces has been investigated by several authors [1, 24].

Of course, we are only interested in those segments of the developable surface which are enveloped by the face $\widehat{\mathbf{p}}_0, \ldots, \widehat{\mathbf{p}}_f$ of the moving polyhedron. In order to compute these segments we have to intersect the instantaneous ruling with the line segments between neighbouring corners of the face. Let $\widehat{\mathbf{p}}_i$ and $\widehat{\mathbf{p}}_j$ be two neighbouring corners of the face. Then, at any instant t, the intersection of the instantaneous ruling with the line through $\widehat{\mathbf{p}}_i$ and $\widehat{\mathbf{p}}_j$ is given by

$$\mathbf{a}_{i,j}(t) = \mathrm{M}(t)((1-\mu_{i,j})\,\widehat{\mathbf{p}}_i + \mu_{i,j}\,\widehat{\mathbf{p}}_j) \quad \text{where} \quad \mu_{i,j} = \frac{\widehat{\mathbf{p}}_i^{\top}\mathrm{M}(t)^{\top}\dot{\mathrm{M}}^*(t)\widehat{\mathbf{F}}}{(\widehat{\mathbf{p}}_i - \widehat{\mathbf{p}}_j)^{\top}\mathrm{M}(t)^{\top}\dot{\mathrm{M}}^*(t)\widehat{\mathbf{F}}}. \tag{25}$$

The point $\mathbf{a}_{i,j}(t)$ lies between $\widehat{\mathbf{p}}_i$ and $\widehat{\mathbf{p}}_j$ if the inequality $0 \le \mu_{i,j} \le 1$ is fulfilled. Note that $\mathbf{a}_{i,j}(t)$ describes a piecewise rational curve, therefore it is possible to find a B–spline representation of this

curve.
Assume the face $\widehat{\mathbf{p}}_0, \ldots, \widehat{\mathbf{p}}_f$ of the moving polyhedron to be convex. (Otherwise, the face should be subdivided in convex segments.) Then, at any instant t, the instantaneous ruling of the developable surface enveloped by $\mathbf{F}(t)$ intersects the boundaries of the face in at most 2 points. If two intersection points exist, the face envelopes a segment of the developable surface, and the line segment between the intersection points is contained in this surface. Otherwise, the instantaneous ruling is not contained in the boundary surface of the region traced out by the moving polyhedron.
The developable surface segments which are enveloped by the face $\widehat{\mathbf{p}}_0, \ldots, \widehat{\mathbf{p}}_f$ of the moving polyhedron can be found with help of the following

Algorithm 2.

1. `For all boundaries` $\widehat{\mathbf{p}}_i, \widehat{\mathbf{p}}_j$ `of the moving face:`
   Compute numerically those intervals of the time t, for which $0 \leq \mu_{i,j} \leq 1$ holds, and store them.
2. `For all pairs` $\widehat{\mathbf{p}}_i, \widehat{\mathbf{p}}_j$ `and` $\widehat{\mathbf{p}}_k, \widehat{\mathbf{p}}_l$ `of different boundaries:`
   Find the intervals of the time t, where the two inequalities $0 \leq \mu_{i,j} \leq 1$ and $0 \leq \mu_{k,l} \leq 1$ are fulfilled. If $[a, b] \subset \mathbb{R}$ is such an interval, then the surface segment

$$\mathbf{x}(s,t) = (1-s)\mathbf{a}_{i,j}(t) + s\mathbf{a}_{k,l}(t) \qquad (s,t) \in [0,1] \times [a,b] \subset \mathbb{R}^2 \tag{26}$$

   is a segment of the developable surface which is enveloped by the face.

In order to compute the intervals of interest in step 1 we have to find the roots of certain polynomials. This should be done with help of appropriate methods from Numerical Analysis, e.g., using the method of false position (Regula falsi).
The computation of the ruled surface segments which are generated by the edges of the moving polyhedron is much easier than that of the developable surfaces. Consider an edge $\widehat{\mathbf{p}}_i, \widehat{\mathbf{p}}_j$ of the moving polyhedron. The parametric representation

$$\mathbf{y}(s,t) = (1-s)\,\mathrm{M}(t)\,\widehat{\mathbf{p}}_i + s\,\mathrm{M}(t)\,\widehat{\mathbf{p}}_j \qquad (s,t) \in [0,1] \times [t_0, t_s] \subset \mathbb{R}^2 \tag{27}$$

describes the ruled surface segment which is obtained by moving the edge through the fixed space $\widehat{\mathrm{E}}^3$.
Clearly, the developable surface segments (26) as well as the ruled surface segments (27) can be described by rational B–spline surfaces. The following algorithm summarizes the computation of the boundary surface of the region which is traced out by the moving polyhedron:

Algorithm 3.

1. `For all edges` $\widehat{\mathbf{p}}_i, \widehat{\mathbf{p}}_j$ `of the moving polyhedron:`
   Compute the ruled surfaces which are generated by moving the edge through the fixed space E^3, see (27).
2. `For all faces` $\widehat{\mathbf{p}}_0, \ldots, \widehat{\mathbf{p}}_f$ `of the polyhedron:`
   Compute the segments of developable surfaces which are enveloped by the moving face, see Algorithm 2.
3. `For each surface segment:`
   Compute all self intersection curves, for instance, by applying an appropriate subdivision method (see [11], chapter 12.2). If necessary, remove those surface parts that are contained in the interior of the region. Such surface parts can be detected approximately by searching for intersections with a set of positions of the moving polyhedron. Note that the accuracy of this computation depends on the number and distribution of the used positions.
4. `For each pair of surface segments:`
   Compute their intersection curves and remove interior surface parts analogously to step 3.

Note that the intersection of two ruled surface segments (27) that are generated by intersecting edges contains the trajectory of the intersection point. Further note that for an edge $\widehat{\mathbf{p}}_i, \widehat{\mathbf{p}}_j$ of the face $\widehat{\mathbf{p}}_0, \ldots, \widehat{\mathbf{p}}_f$ in $\widehat{\mathbf{F}}$ the corresponding generated surfaces have first order contact along the rational curve $\mathbf{a}_{i,j}(t)$ from (25).
As an example, Figure 4b shows the region which is traced out by the rational motion from Figure 4a of the unit cube. The developable surface segments are marked in white, whereas the remaining ruled surface segments have been drawn in gray. The motion is illustrated with help of the control and weight positions of the moving cube.

Remarks.

1. In this section we have developed a method for the computation of the boundary surface of the region which is traced out by a moving polyhedron. As the main result, this surface has proved to consist of segments of rational B–spline surfaces. It should be noted, however, that the computation of the B–spline representations of these surface segments is relatively complicated and expensive. In general we will obtain a large number of segments, and the polynomial degree is relatively high. In most applications it should be sufficient to compute the boundary surface pointwise and to approximate it, e.g., by a triangulation of the obtained points. But it is always possible to find an exact B–spline representation of the boundary surface.
2. The techniques presented in this section can easily be extended to the computation of surfaces that are enveloped by moving rational developable surfaces. This also covers the important subcases of a moving cylinder or cone which are of major importance in the tool path planning of milling machines. For detailed information the reader is referred to [29].

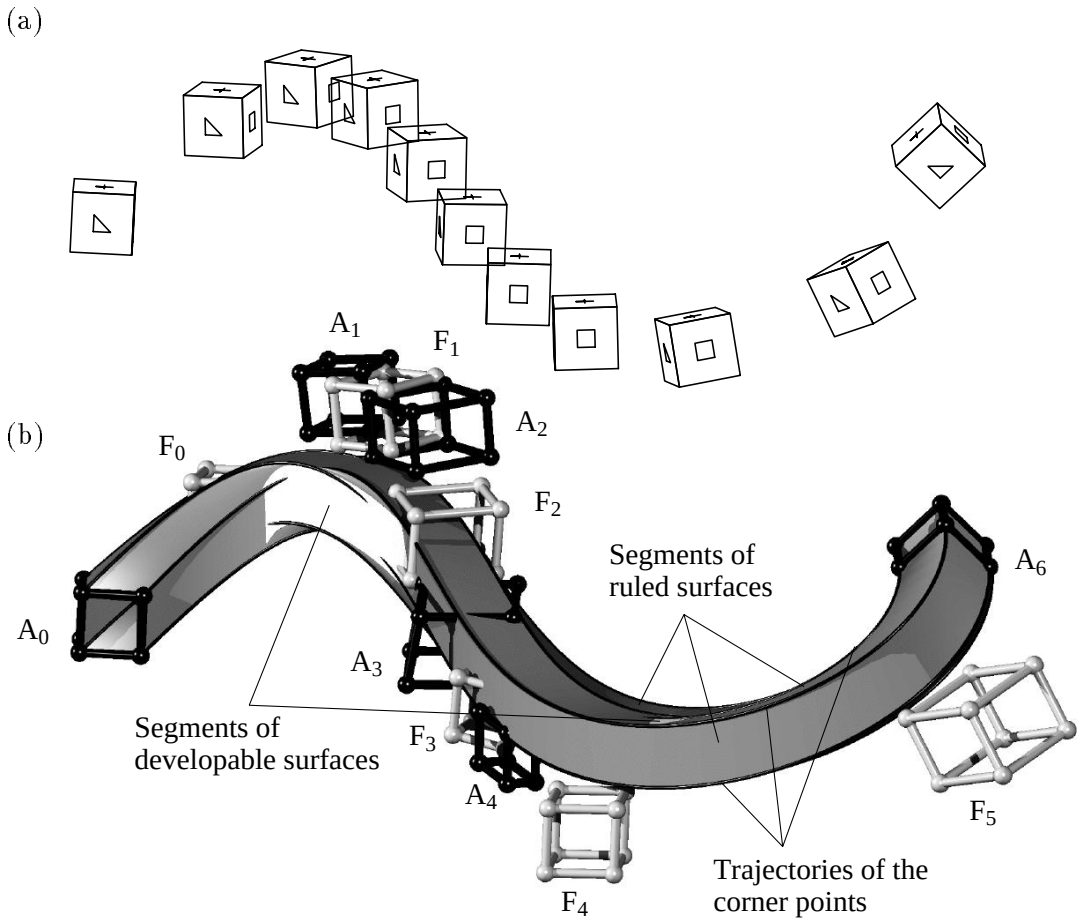

Figure 4: A rational motion of order 6 (a) and the region which is traced out by the moving unit cube (b)

4. Interpolation and optimization of rational B–spline motions

This section shows how to interpolate a certain set of positions of a moving object. The positions are described by the corresponding Euclidean spatial displacements, and the resulting motion M will be a rational B–spline motion of degree k.

4.1 Interpolation

Let us consider $n+1$ spatial displacements

$$\mathrm{P}_i = \left(\begin{array}{c|ccc} 1 & 0 & 0 & 0 \\ \hline | & & & \\ \vec{\mathbf{v}}_i & & \mathrm{R}_i & \\ | & & & \end{array}\right) \quad (\vec{\mathbf{v}}_i = \begin{pmatrix} v_{i.1} \\ v_{i.2} \\ v_{i.3} \end{pmatrix} \in \mathbb{R}^3), \quad \mathrm{i} = 0, \ldots, \mathrm{n}, \tag{28}$$

with given parameters $t_i \in \mathbb{R}$, $t_0 < t_1 < \ldots < t_n$. According to Weiss[31] Euler's parameters $\widetilde{e}_i = (\, e_{i.0}\ e_{i.1}\ e_{i.2}\ e_{i.3}\,)^\top$ of the orientational part of P_i result from

$$\begin{array}{ccccccccc} e_{i.0} & : & e_{i.1} & : & e_{i.2} & : & e_{i.3} & = & \\ 1 + r_{i.1.1} + r_{i.2.2} + r_{i.3.3} & : & r_{i.2.3} - r_{i.3.2} & : & r_{i.3.1} - r_{i.1.3} & : & r_{i.1.2} - r_{i.2.1} & = & \\ r_{i.2.3} - r_{i.3.2} & : & 1 + r_{i.1.1} - r_{i.2.2} - r_{i.3.3} & : & r_{i.1.2} + r_{i.2.1} & : & r_{i.3.1} + r_{i.1.3} & = & \\ r_{i.3.1} - r_{i.1.3} & : & r_{i.1.2} + r_{i.2.1} & : & 1 - r_{i.1.1} + r_{i.2.2} - r_{i.3.3} & : & r_{i.2.3} + r_{i.3.2} & = & \\ r_{i.1.2} - r_{i.2.1} & : & r_{i.3.1} + r_{i.1.3} & : & r_{i.2.3} + r_{i.3.2} & : & 1 - r_{i.1.1} - r_{i.2.2} + r_{i.3.3} & & \end{array} \tag{29}$$

with $\mathrm{R}_i = (r_{i.j.k})_{j,k=1,2,3}$. Notice that at least one of the relationships from (29) yields a solution different to $0:0:0:0$. In the sequel we assume normalized Euler parameters $\widetilde{e}_i$, i.e. $\widetilde{e}_i^\top \widetilde{e}_i = 1$. Nevertheless note that the vectors $\widetilde{e}_i$ are not unique since $\widetilde{e}_i$ and $-\widetilde{e}_i$ describe the same orientation. For an optimal choice of the $\widetilde{e}_i$, consider the angle $\sphericalangle(\widetilde{e}_i, \widetilde{e}_{i+1})$ between the vectors $\widetilde{e}_i$ and $\widetilde{e}_{i+1}$, which is half of the angle of the continuous screw motion that maps the ith position onto the $(i+1)$st position of $\widehat{\mathrm{E}}^3$ (cf. [2]). In order to achieve a smooth motion interpolant, we therefore choose the directions of the vectors $\widetilde{e}_i$ such, that

$$e_{i.0}e_{i+1.0} + e_{i.1}e_{i+1.1} + e_{i.2}e_{i+1.2} + e_{i.3}e_{i+1.3} \geq 0, \tag{30}$$

This guarantees, that the angles $\sphericalangle(\widetilde{\mathrm{e}}_\mathrm{i}, \widetilde{\mathrm{e}}_{\mathrm{i+1}})$ are as small as possible.
We now construct a motion $\mathrm{M} = \mathrm{M}(t)$ such that it satisfies the interpolation conditions

$$\mathrm{M}(t_i) = \lambda_i^2\, \mathrm{P}_i \quad (i = 0, \ldots, n) \tag{31}$$

with positive real factors λ_i^2. Moreover we assume $\lambda_i > 0$. These factors can be used as design parameters. Figure 5 shows the influence of the λ_i to the motion interpolant. By increasing λ_i, the interpolant is pulled towards the ith position. In general, it is sufficient to choose $\lambda_i = 1$ $(i = 0, \ldots, n)$. Nevertheless note that (31) does not guarantee that all weights of the resulting motion interpolant are positive.
According to our assumptions the Euler parameters have to be piecewise rational functions of degree $[k/2]$, where $[\,]$ denote the Gaussian brackets. The simplest way to achieve this, is to interpolate with a vector–valued B–spline function of degree $l = [k/2]$, i.e.

$$\widetilde{d} = (\, d_0\ d_1\ d_2\ d_3\,)^\top = \widetilde{d}(t) = \sum_{j=0}^{n} N_{j.T_{\mathrm{rot}}}^{l}(t)\, \widetilde{c}_j, \quad \widetilde{c}_j \in \mathbb{R}^4, \quad \mathrm{l} = \left[\frac{\mathrm{k}}{2}\right], \tag{32}$$

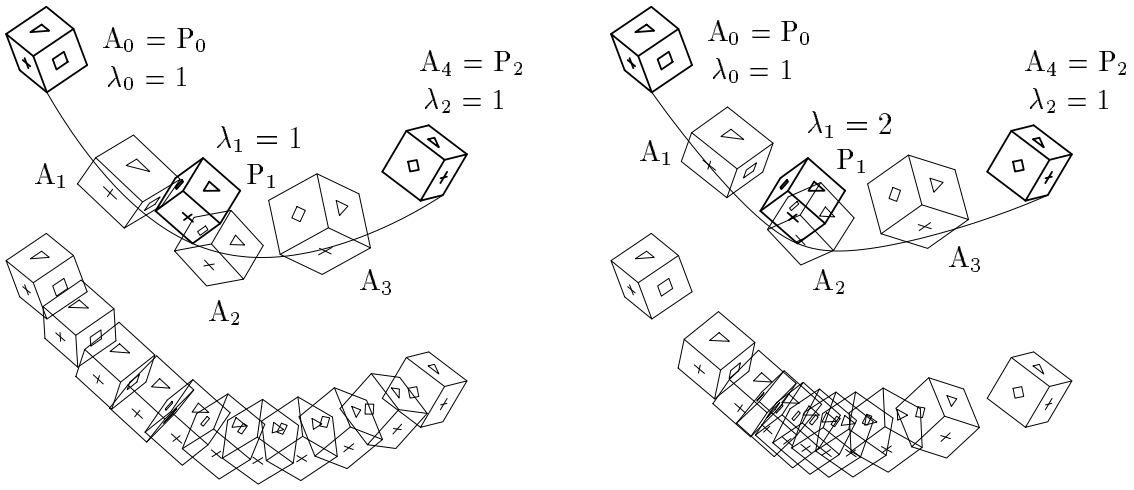

Figure 5: Rational B–spline motions of degree 4, which interpolate a given set of 3 positions P_i with given parameters t_i. By increasing λ_1, the motion is pulled towards the position defined by P_1.

where the $N^l_{j,T_{\rm rot}}$ $(j = 0, \ldots, n)$ are defined over the knot vector

$$T_{\rm rot} = (\underbrace{t_0, \ldots, t_0}_{l+1 \text{ times}}, \tau_1, \tau_2, \ldots, \tau_{n-l}, \underbrace{t_n, \ldots, t_n}_{l+1 \text{ times}}). \tag{33}$$

We choose $(l+1)$–fold knots at the end of $T_{\rm rot}$ in order to achive endpoint interpolation. On the other hand the knots τ_i are arbitrary, but have to satisfy the Schoenberg–Whitney conditions with respect to the time parameters t_i (see [21]). For instance, one can choose an appropriate subsequence of the given parameter sequence (t_i).
The remaining calculations are straightforward. First, (31) yields a system

$$\lambda_i \widetilde{e}_i = \sum_{j=0}^{n} N^l_{j,T_{\rm rot}}(t_i)\, \widetilde{c}_j \quad (i = 0, \ldots, n). \tag{34}$$

of $n+1$ linear equations for the computation of the $n+1$ coefficient vectors $\widetilde{c}_j$. In the second step, one has to calculate the knot vector T of the motion interpolant. If all knots τ_i are simple, i.e. if $\tau_i < \tau_{i+1}$, the function $\widetilde{d}$ is $l-1$ times continuously differentiable. In this case the B–splines $N^k_{i,T}$ $(i = 0, \ldots, m)$ from (13) have to be C^{l-1} continuous functions of degree k. Hence we get

$$T = (\underbrace{t_0, \ldots, t_0}_{k+1 \text{ times}}, \underbrace{\tau_1, \ldots, \tau_1}_{k-l+1 \text{ times}}, \underbrace{\tau_2, \ldots, \tau_2}_{k-l+1 \text{ times}}, \ldots, \underbrace{\tau_{n-l}, \ldots, \tau_{n-l}}_{k-l+1 \text{ times}}, \underbrace{t_n, \ldots, t_n}_{k+1 \text{ times}}). \tag{35}$$

For the sake of simplicity we assume $\overline{v}_0 \equiv 1$. With the use of the techniques presented in the previous section one now is able to calculate the submatrices S_i and the weights $w_{i,0}$ of the

$$m + 1 = (k - l)(n - l + 1) + n + 1 \tag{36}$$

coefficient matrices A_i of the motion (13). These calculations are independent of the choice of the orientation of E^3 as well as of the orientation of $\widehat{E}^3$. It might happen, that some of the weights are negative. In this case, the motion interpolant does not fulfill the convex hull property with respect to its control structure. If such a situation appears, one can change the design parameters λ_i to get a better solution.
Finally, the translational part of M results from

$$\lambda_i^2 \, \vec{\mathbf{v}}_i = \sum_{j=0}^{m} N_{j,T}^k(t_i) \, \vec{\mathbf{w}}_j \quad (i = 0, \ldots, n). \tag{37}$$

This yields an $(m-n)$–dimensional solution space of M.

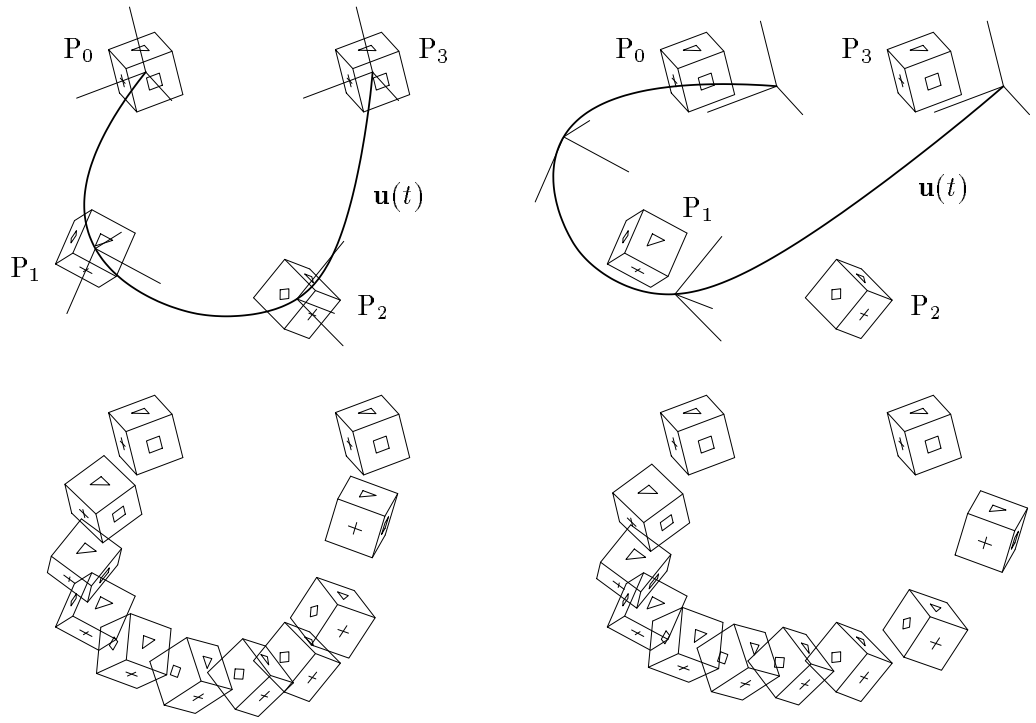

Figure 6: Rational B–spline motions of degree 4, which interpolate a given set of 4 positions. In the left example $\widehat{\mathbf{u}}$ is the center of gravity, which leads to a smoother motion interpolant.

For most applications in mechanics or computer graphics, it is useful to choose the origin $\widehat{\mathbf{u}}$ of the moving system $\widehat{E}^3$ such that it has a special meaning for the moving object, e.g. is the center of gravity. In this context, the trajectory of $\widehat{\mathbf{u}}$ should have minimum length. We propose to calculate an optimal solution by minimizing the sum of the squared distances between neighbouring control positions of $\widehat{\mathbf{u}}$, i.e.

$$\sum_{j=0}^{m-1} \left(\frac{1}{w_{j+1.0}} \, \vec{\mathbf{w}}_{j+1} - \frac{1}{w_{j.0}} \, \vec{\mathbf{w}}_j \right)^2 \to \text{Min.} \tag{38}$$

Consequently, the moving object is expected to stay within a certain minimum area, if $\hat{\mathbf{u}}$ is the center of gravity (cf. Figure 6). The minimization (38) leads to a system of $m-n$ linear equations, which in general yields a unique solution of M. All examples in this section have been calculated with this approach.
Alternatively, another solution can be obtained by solving

$$\lambda_i^2 \, \vec{\mathbf{v}}_i = \sum_{j=0}^{n} N^l_{j.T_{\mathrm{rot}}}(t_i) \, \vec{\mathbf{w}}'_j \quad (i = 0, \ldots, n) \tag{39}$$

instead of (37), where the B–splines $N^l_{j.T_{\mathrm{rot}}}$ are defined over the knot vector T_{rot}. In this case, the vectors $\vec{\mathbf{w}}_i$ $(i = 0, \ldots, m)$ result from the vectors $\vec{\mathbf{w}}'_j$ $(j = 0, \ldots, n)$ by elevating the degree of the vector valued B–spline function

$$\vec{\mathbf{v}}(t) = \sum_{j=0}^{n} N^l_{j.T_{\mathrm{rot}}}(t) \, \vec{\mathbf{w}}'_j \tag{40}$$

$k-l$ times (see [18]).
Summing up, the presented interpolation scheme consists of three steps:

Algorithm 4.

1. Interpolate Euler's parameters d_i by solving (34). Therefore, choose the parameters λ_i, e.g. $\lambda_i = 1$.
2. Calculate the weights $w_{j.0}$ and the submatrices S_j.
3. Calculate the remaining entries of the matrices A_j by solving (37) and the minimization (38).

Remarks.

1. If $\widetilde{d}(t)$ passes through the origin of $\mathbb{R}^4$, i.e. if $||\widetilde{d}(t^*)||^2 = 0$ for some $t^* \in [t_i, t_i + 1]$, the weight function $v_0(t) = ||\widetilde{d}(t)||^2$ has an at least quadratic factor with all entries of $\mathrm{R}(t)$ from (1) in common. In this case, the maximal degree of the rotational part of the ith motion segment is $2(l-2)$. If $\widetilde{d}(t)$ gets close to the origin of $\mathbb{R}^4$, but does not pass, the interpolation scheme produces undesired loops. These situations can be avoided if one chooses the design parameters λ_i such that the resulting weight function $v_0(t)$ satisfies $v_0(t^*) \approx 1$ for all $t^* \in [t_0, t_n]$.
2. The interpolation scheme also applies to periodic motions. In this case one only has to choose appropriate knot vectors T_{rot} and T.
3. If the parameters t_i are unknown, they can be estimated with help of the translational and rotational parts of the displacements $\Delta\mathrm{P}_i := \mathrm{P}_{i+1} \cdot \mathrm{P}_i^{-1}$. The displacement $\Delta\mathrm{P}_i$ maps the ith position of the moving object onto the $(i+1)$st position. A list of different possibilities can be found in [14]. The simplest way to estimate the parametrization is by choosing $\Delta t_i = t_{i+1} - t_i$ constant, e.g. $\Delta t_i = 1$. In order to estimate the parametrization independent of the choice of the reference frames, one might choose $\Delta t_i = \sphericalangle(\tilde{\mathrm{e}}_\mathrm{i}, \tilde{\mathrm{e}}_{\mathrm{i+1}})$. Note that this is only possible if $\sphericalangle(\tilde{\mathrm{e}}_\mathrm{i}, \tilde{\mathrm{e}}_{\mathrm{i+1}}) \neq 0$, i.e. if the displacement $\Delta\mathrm{P}_i$ does not represent a translation.

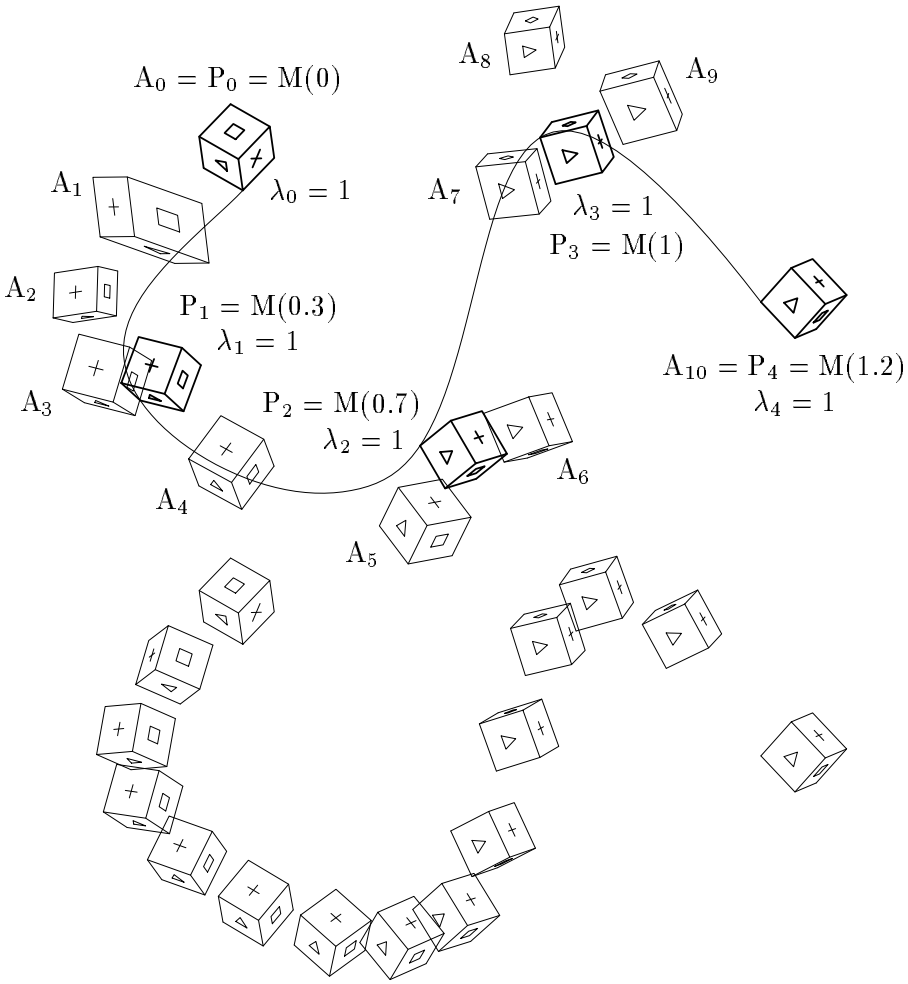

Figure 7: Rational B–spline motion of degree 6, which interpolates a given set of 5 positions.

4. The presented interpolation methods for the translational part depend on the choice of the origin of $\widehat{\mathrm{E}}^3$. Methods which are independent of the origin of $\widehat{\mathrm{E}}^3$ could be based on Rath's mapping from Remark 3 of the previous section. Under this mapping, coordinate transformations in $\widehat{\mathrm{E}}^3$ induce affine transformations in A^{12} that map Φ onto itself. Analogously to affine invariant methods in scattered data interpolation, which were introduced by Nielson (cf. [20]), one could use an affine invariant metric in A^{12} for a minimization of the control polygon of the image curve of $\mathrm{M}(t)$ instead of the Euclidean metric in (38). A slightly different approach to origin–independent motion interpolation is discussed in [14]. Nevertheless, if the origin of $\widehat{\mathrm{E}}^3$ has a special meaning to the moving object, which in general is the case, the result will be better if one exploits this property.

4.2 Rational motions satisfying energetic constraints

Till now, the moment of inertia and the mass of the moving object have not been considered in our interpolation scheme. In the following we briefly outline how the resulting rational B-spline motion can be modified such that it satisfies certain energetic constraints. We therefore assume that the origin of the moving system is the center of mass of the moving object.
Consider one segment of the motion $\mathrm{M}(t)$ which connects to consecutive positions $\mathrm{M}(t_i)$ and $\mathrm{M}(t_{i+1})$. We want to replace this motion segment by a new segment obtained by solving the energetic optimization problems

$$\int_{t_i}^{t_{i+1}} \vec{\mathbf{a}}^{\top} \vec{\mathbf{a}} \, \mathrm{d}t \to \mathrm{Min} \tag{41}$$

and

$$\int_{t_i}^{t_{i+1}} \vec{\alpha}^{\top} \vec{\alpha} \, \mathrm{d}t \to \mathrm{Min} \tag{42}$$

where $\vec{\mathbf{a}} = \vec{\mathbf{a}}(t)$ and $\vec{\alpha} = \vec{\alpha}(t)$ denote the acceleration of the origin and the angular acceleration of the moving object, respectively. As boundary conditions, the motion $\bar{\mathrm{M}}(t)$ has to interpolate the positions, the velocities and the angular velocities of the moving object at t_i and t_{i+1}.
Of course, the solution of the first variational problem (41) is well known: The trajectory of the origin has to be the polynomial cubic curve which is obtained as Hermite interpolant of the segment end points and of the velocities of the origin in these points. On the other hand, the exact solution of (42) seems to be unknown. An approximate solution, however, can be obtained with help of an appropriate optimization techniques from Numerical Analysis. Figure 8 shows the effect of the optimization for a rational B–spline motion which consists of only one segment. The optimized motion is denoted by $\overline{\mathrm{M}}(t)$. For any details of our method we refer the reader to [15].

Conclusion

Using rational B–spline motions it is possible to apply many of the fundamental techniques of Computer Aided Geometric Design to problems from Kinematics and Robotics. As an important advantage, the use of rational motions supports the data exchange with CAD systems: sweeping surfaces, surfaces enveloped by moving planes and surfaces enveloped by moving developable surfaces including cylinders or cones are rational B–spline (NURBS) surfaces.
Due to space limitations it was not possible to focus on every detail. Some methods for motion interpolation based on rational dual quaternion curves have been derived in [14, 15]. A geometric discussion of rational motions and approximation techniques can be found in [15], methods for the construction of rational sweeping surfaces of arbitrary degree in [15, 16]. The approximation algorithm presented in [15] has been used in order to approximate the motion of a human knee joint in [17]. Algorithms for interactive design of rational B–spline motions and methods for the calculation of enveloping surfaces of moving developable surfaces have been developed in [29]. This thesis also

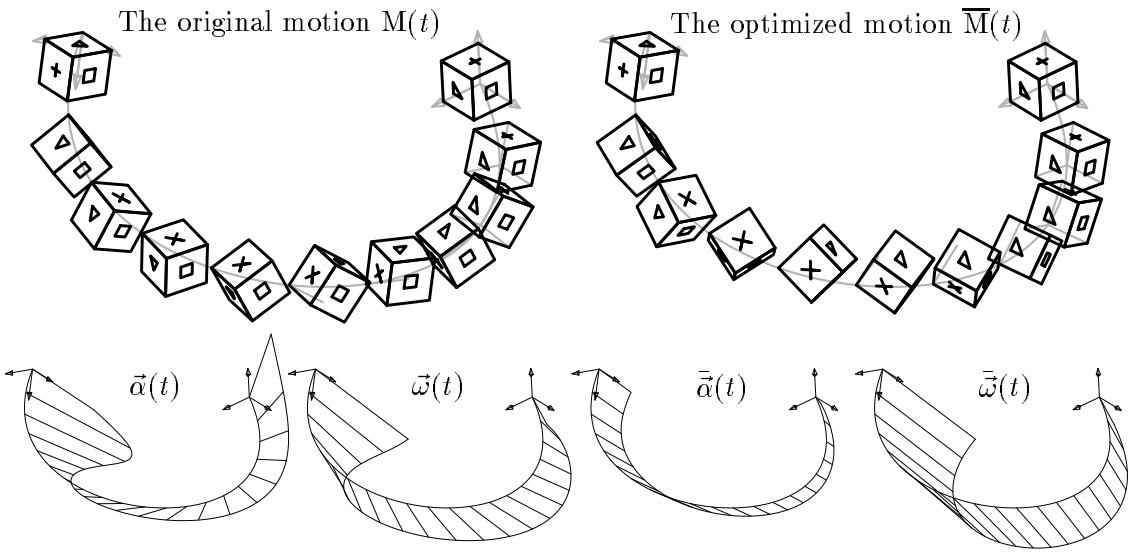

Figure 8: Optimization of a rational B–spline motion.

studies the special case of planar rational B–spline motions.
Nevertheless there still exist a lot of open questions. Most important seems to be to find an intrinsic control structure of rational B–spline motions which allows interactive motion design. Currently, the lack of interactivity turns out to be the major drawback of this approach, especially if one wants to apply rational B–spline motions to different tasks in Computer Animation. Other possibilities for future research are the contruction of obstacle avoiding motions, the generalization of subdivision schemes to rational motion splines, automatic fairing of position sets, optimized motion design for milling machines, or the design of rational motions under dynamic constraints. Finally we would like to mention that a modified version of the interpolation algorithm presented in this paper is currently subject of investigation for the motion control of industrial robots [10].

References

[1] R.M.C. Bodduluri, and B. Ravani, Design of developable surfaces using duality between plane and point geometries, *Computer–Aided Design* **25**, 621–632 (1993).

[2] O. Bottema and B. Roth, *Theoretical Kinematics*, North-Holland Publishing Company, Amsterdam, New York, Oxford (1979).

[3] M. do Carmo, *Differential Geometry of Curves and Surfaces*, Prentice–Hall, Englewood Cliffs 1976.

[4] H.S.M. Coxeter, *Projective Geometry*, Blaisdell, New York 1964.

[5] G. Darboux, Sur les mouvements algebriques, Note III in G. Kœnigs: *Leçons de cinématique*, Hermann, Paris (1895).

[6] G. Farin, *Curves and Surfaces for Computer Aided Geometric Design*, 3rd ed., Academic Press, Boston 1993.

[7] Q. J. Ge and B. Ravani, Computer Aided Geometric Design of Motion Interpolants, *Proc. ASME Design Automation Conference*, DE-Vol. **32–2**, 33–41 (1991).

[8] Q.J. Ge, and B. Ravani, Geometric Construction of Bézier Motions, *ASME Journal of Mechanical Design*, Vol. **116**, 749–755 (1994).

[9] D. Handscomb, Knot–elimination: Reversal of the Oslo Algorithm, *International Series of Numerical Mathematics* **81** (1987), Birkhäuser, Basel, 103–111.

[10] Th. Horsch and B. Jüttler, Spline interpolation for industrial robots and its application, in preparation.

[11] J. Hoschek and D. Lasser, *Fundamentals of Computer Aided Geometric Design*, Peters, Wellesley (1993).

[12] J. Johnstone and J. Williams, Rational control of orientation for animation, *Proceedings of Graphics Interface '95*, 179–186.

[13] B. Jüttler, Über zwangläufige rationale Bewegungsvorgänge, *Sitzungsber. d. Österr. Akad. d. Wiss.* **202**, 117–132 (1993).

[14] B. Jüttler, Visualization of moving objects using dual quaternion curves, *Computers & Graphics* **18**(3), 315–326 (1994).

[15] B. Jüttler, *Rationale Bézierdarstellung räumlicher Bewegungsvorgänge und ihre Anwendung zur Beschreibung bewegter Objekte*, Dissertation, Technische Hochschule Darmstadt, Verlag Shaker, Aachen 1994.

[16] B. Jüttler, Spatial rational motions and their application in Computer Aided Geometric Design. in M. Dæhlen, T. Lyche, und L.L. Schumaker (eds.): *Mathematical Methods for Curves and Surfaces*, Vanderbilt University Press, La Vergne TN, 1995.

[17] C. Keil, *Approximation gemessener Kniebewegungen*, Diplomarbeit, TH Darmstadt, 1995.

[18] K. Mørken, Some identities for products and degree raising of splines, *Constructive Approximation* **7**, 195–208 (1991).

[19] G. Nielson, Bernstein/Bézier curves and splines on spheres based upon a spherical de Casteljau algorithm, Arizona State University Technical Report TR–88–028 (1988).

[20] G. Nielson, Applications of an affine invariant metric. in L.L. Schumaker, T. Lyche (eds.): *Mathematical Methods in Computer Aided Geometric Design*, Academic Press, 1989.

[21] G. Nürnberger, *Approximation by Spline Functions*, Springer, Berlin, (1989).

[22] F.C. Park and B. Ravani, Bézier Curves on Riemannian Manifolds and Lie Groups with Kinematic Applications, to appear (1994).

[23] D. Pletinckx, Quaternion calculus as a basic tool in computer graphics, *The Visual Computer* **5**, 2–13 (1989).

[24] H. Pottmann, and G. Farin, Developable rational Bézier and B–spline surfaces, *Computer Aided Geometric Design* **5**, 513–531 (1995).

[25] W. Rath, Matrix Groups and Projective Kinematics, *Abh. Math. Sem. Univ. Hamburg* **63**, 177–196 (1993).

[26] O. Röschel, Rationale räumliche Zwangläufe vierter Ordnung, *Sitzungsber. d. Österr. Akad. d. Wiss.* **194**, 185–202 (1985) (abstracted in Math. Reviews 87k: 53016).

[27] L.L. Schumaker, *Spline Functions: Basic Theory*, Wiley–Interscience, New York (1981).

[28] K. Shoemake, Animating Rotation with Quaternion Curves, *ACM Siggraph* **19**, 245–254 (1985).

[29] M.G. Wagner, A Geometric Approach to Motion Design, PhD thesis, Technische Universität Wien (1994).

[30] M.G. Wagner, Planar Rational B–Spline Motions, *Computer Aided Design* **27**, 129–137 (1995).

[31] E.A. Weiss, *Einführung in die Liniengeometrie und Kinematik*, Teubners Mathematische Leitfäden **41**, Leipzig und Berlin, (1935).

[32] W. Wunderlich, Kubische Zwangläufe, *Sitzungsber. d. Österr. Akad. d. Wiss.* **193**, 45–68 (1984) (abstracted in Math. Reviews 86h: 53011).

Appendix: Computing one segment of a rational B–spline motion

Applying Böhm's algorithm to the B–spline functions $\overline{v}_0(t)$, $\vec{\mathbf{v}}(t)$, and $\widetilde{d}(t)$ we get the Bernstein–Bézier representation

$$\begin{aligned} \overline{v}_0^{(j)}(\tau) = \sum_{i=0}^{k-2l} B_i^{k-2l}(\tau)\,\overline{v}_{i.0}^{(j)}, \qquad \vec{\mathbf{v}}^{(j)}(\tau) = \sum_{i=0}^{k} B_i^{k}(\tau)\,\vec{\mathbf{v}}_i^{(j)}, \\ \text{and } \widetilde{d}^{(j)}(\tau) = \sum_{i=0}^{l} B_i^{l}(\tau)\,\widetilde{d}_i^{(j)}, \end{aligned} \tag{43}$$

with $\vec{\mathbf{v}}_i^{(j)} = (\, v_{i.1}^{(j)}\ v_{i.2}^{(j)}\ v_{i.3}^{(j)}\,)^\top$ and $\widetilde{d}_i^{(j)} = (\, d_{i.0}^{(j)} \ldots d_{i.3}^{(j)}\,)^\top$ of their polynomial segments, where each segment $t_i < t < t_{i+1}$ is represented with respect to the *local* parameter $\tau = (t - t_j)/(t_{j+1} - t_j)$. Hereby, $B_q^p(\tau) = \binom{p}{q}\tau^q(1-\tau)^{p-q}$ denotes the q–th Bernstein polynomial of degree p. We therefore obtain from (5) the jth segment of the spline motion $\mathrm{M}(t)$, i.e.

$$\mathrm{M}^{(j)}(\tau) = \sum_{i=0}^{k} B_i^k(\tau)\,\mathrm{A}_i^{(j)} \tag{44}$$

where

$$\mathrm{A}_i^{(j)} = \frac{1}{\binom{k}{i}} \sum_{i_1+i_2+i_3=i} \binom{l}{i_1}\binom{l}{i_2}\binom{k-2l}{i_3} \left(\begin{array}{c|ccc} \overline{v}_{i_3,0}^{(j)}\,\delta_{i_1,i_2}^{(j)} & 0 & 0 & 0 \\ \hline v_{i,1}^{(j)} & & & \\ v_{i,2}^{(j)} & & \overline{v}_{i_3,0}^{(j)}\mathrm{D}_{i_1,i_2}^{(j)} & \\ v_{i,3}^{(j)} & & & \end{array}\right), \tag{45}$$

$$\delta_{r,s}^{(j)} = d_{r,0}^{(j)}\,d_{s,0}^{(j)} + d_{r,1}^{(j)}\,d_{s,1}^{(j)} + d_{r,2}^{(j)}\,d_{s,2}^{(j)} + d_{r,3}^{(j)}\,d_{s,3}^{(j)} \quad \text{and} \tag{46}$$

$$\mathrm{D}_{r,s}^{(j)} = \begin{pmatrix} d_{r,0}^{(j)}\,d_{s,0}^{(j)} + d_{r,1}^{(j)}\,d_{s,1}^{(j)} - d_{r,2}^{(j)}\,d_{s,2}^{(j)} - d_{r,3}^{(j)}\,d_{s,3}^{(j)} & 2(d_{r,1}^{(j)}\,d_{s,2}^{(j)} - d_{r,0}^{(j)}\,d_{s,3}^{(j)}) & 2(d_{r,1}^{(j)}\,d_{s,3}^{(j)} + d_{r,0}^{(j)}\,d_{s,2}^{(j)}) \\ 2(d_{r,1}^{(j)}\,d_{s,2}^{(j)} + d_{r,0}^{(j)}\,d_{s,3}^{(j)}) & d_{r,0}^{(j)}\,d_{s,0}^{(j)} - d_{r,1}^{(j)}\,d_{s,1}^{(j)} + d_{r,2}^{(j)}\,d_{s,2}^{(j)} - d_{r,3}^{(j)}\,d_{s,3}^{(j)} & 2(d_{r,2}^{(j)}\,d_{s,3}^{(j)} - d_{r,0}^{(j)}\,d_{s,1}^{(j)}) \\ 2(d_{r,1}^{(j)}\,d_{s,3}^{(j)} - d_{r,0}^{(j)}\,d_{s,2}^{(j)}) & 2(d_{r,2}^{(j)}\,d_{s,3}^{(j)} + d_{r,0}^{(j)}\,d_{s,1}^{(j)}) & d_{r,0}^{(j)}\,d_{s,0}^{(j)} - d_{r,1}^{(j)}\,d_{s,1}^{(j)} - d_{r,2}^{(j)}\,d_{s,2}^{(j)} + d_{r,3}^{(j)}\,d_{s,3}^{(j)} \end{pmatrix} \tag{47}$$

$(j = 0, \ldots, s-1)$.