Proceedings of DETC'00
ASME 2000 Design Engineering Technical Conferences
and Computers and Information in Engineering Conference
Baltimore, Maryland, September 10-13, 2000

DETC2000/MECH-14106

A GEOMETRIC ALGORITHM FOR THE COMPUTATION OF THE CONSTANT-ORIENTATION WORKSPACE OF 6-RUS PARALLEL MANIPULATORS

Ilian A. Bonev and Clément M. Gosselin[*]
Département de Génie Mécanique
Université Laval
Québec, Québec, Canada G1K 7P4
e-mail: gosselin@gmc.ulaval.ca

ABSTRACT
This paper presents a geometric algorithm for the computation of the constant-orientation workspace of 6-*RUS* parallel manipulators. While the basic philosophy of this algorithm is not new, the proposed computational process involving surface intersections is original and is the major contribution of the paper. Particularly, an analytic description of the points that constitute the edges of the constant-orientation workspace is presented. The latter is done through a new algorithm for the computation of the intersection points between a general torus and a circle.

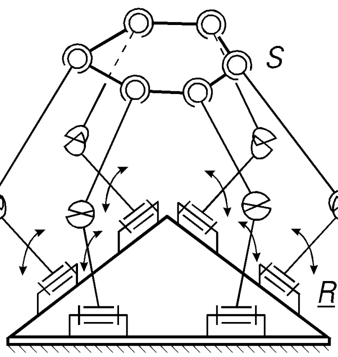

Fig. 1 Schematic of a 6-RUS parallel manipulator.

1 INTRODUCTION
It is well known that parallel manipulators (PMs) have a rather limited and complex workspace. At the same time, the size and shape of the workspace is probably one of the main design criteria. As the *complete workspace* of a 6-DOF PM is a six-dimensional entity which is practically impossible to visualize, algorithms for various subsets of it have been proposed. Apart from the brute-force approach—the discretization algorithms—all other computational schemes are strictly dependent on the particular architecture. Thus, in general, a particular research on workspace analysis can be virtually situated in a 3D array whose axes are the type of workspace subset, the type of algorithm (geometrical, numerical, analytical), and the type of PM architecture.

In the area of 6-DOF PMs, most of the research has been particularly aimed at the simplest and most popular architecture, namely the 6-*UPS* PM, commonly known as the *Stewart-Gough platform*. Another less-studied but also common design is the 6-*RUS* class of PMs (Fig. 1). In this notation, *P* stands for an actuated prismatic joint, *R* for an actuated revolute joint, *U* for a passive universal joint, and *S* for a passive spherical joint.

Undoubtedly, the most popular member of the 6-*RUS* class is the "Hexa" robot (Pierrot et al., 1990), of which an improved version is already available. The first to propose this architecture, however, was Hunt (1983). Some other prototypes have been constructed by Zamanov (Merlet, 1997), by Takeda et al. (1997), by Zabalza et al. (1999) and by Benea (1996). The latter has even performed a detailed set of analyses on this type of manipulator. Two other designs are also commercially available by Servos & Simulation Inc. as motion simulation systems. Finally, a more recent and more peculiar design has been introduced by Hexel Corp., dubbed as the "Rotary Hexapod" (Chi, 1999).

Despite the relative popularity of the 6-*RUS* PM, few researchers have analyzed in detail its workspace. What is more, all of the existing prototypes have rather particular designs which facilitate their analyses—the axes of the actuated revolute joints are either coplanar, parallel, or even coincident. Benea (1996)

[*]corresponding author, tel: (418) 656-3474, fax: (418) 656-7415.

has studied two subsets of the complete workspace. One of them is the *constant-orientation workspace* which is the three-dimensional volume that can be attained by a point from the mobile platform when the platform is kept at a constant orientation. A *discretization* algorithm has been used for this purpose. The philosophy of such an algorithm is rather simple and consists roughly in discretizing the three-dimensional space, solving the inverse kinematic problem at each point, and verifying the constraints that limit the workspace.

Such discretization algorithms are used by almost all researchers and can be applied to any type of architecture. They are clearly computationally intensive and require large amounts of disk space for storing the computed point cloud. A more advanced approach for the computation of the constant-orientation workspace is based on the geometric description of all constraints that limit the workspace. Unlike the discretization methods, the geometric methods are very fast and accurate. Furthermore, they bring insight into the problem and are very useful during the design stage.

It is for the class of 6-*U*$\underline{P}$*S* PMs, that such an approach was first introduced by Gosselin (1990) and then again, under a modified version, by Gosselin et al. (1992), considering only the limits of the actuators. In the first paper, horizontal cross-sections of the constant-orientation workspace have been determined, while in the second, the workspace edges have been defined directly. Merlet (1994) later extended this geometric approach by including the limited ranges of the passive joints and even the risk of link interference. Then, for the class of 6-$\underline{P}$*US* PMs, Merlet and Gosselin (1991) applied the same philosophy to compute horizontal cross-sections of the workspace of their “active wrist”. Recently, a more general and detailed workspace analysis following the same approach was performed by Bonev and Ryu (1999b), where the constant-orientation workspace was directly computed and represented as a solid model in the CAD/CAM system CATIA.

To the best of our knowledge, a geometric algorithm has never been applied to the general 6-$\underline{R}$*US* PM. Yet, only a moderate change in the program code used in (Bonev and Ryu, 1999b) would have been sufficient to produce a similar program for 6-$\underline{R}$*US* PMs. Besides, a similar implementation has also been carried out by Chrisp and Gindy (1999) in Pro/ENGINEER for a 6-*U*$\underline{P}$*S* PM. However, while the use of CATIA or Pro/ENGINEER results in an excellent visualization of the workspace, common experience shows the obvious disadvantages of this approach. Firstly, the two CAD/CAM systems, although quite popular, are not necessarily available to all users of PMs. Secondly, the natural trend in industry is to develop large integrated programs that perform various types of analyses and not just compute the constant-orientation workspace. That is why, in this paper, we propose an algorithm inspired by the one presented in (Gosselin et al., 1992) for computing and representing the edges of the constant-orientation workspace of 6-$\underline{R}$*US* PMs. This algorithm can be easily implemented and requires no special programming libraries.

1.1 Notation and Description of the Architecture

Following the notation used in (Bonev and Ryu, 1999b), we select a fixed reference frame, called the *base frame* with center O and axes x, y, and z. Then, we also select a *mobile frame* that is fixed to the mobile platform, with center C and axes x', y', and z' (Fig. 2). We denote the centers of the U-joints by A_i and the centers of the spherical joints attached at the mobile platform by B_i (in this paper $i = 1, \ldots, 6$). Each point A_i moves along a circular trajectory referred to as *track* i whose center is denoted by O_i. We assume that each actuated revolute joint can rotate fully, without any restriction imposed by the joint itself.

We will refer to the link connecting points O_i and A_i as *proximal link* i and to the link connecting points A_i and B_i as *distal link* i. Let the lengths of all proximal links be equal and denoted by r_A and the lengths of all distal links be equal and denoted by ℓ (Fig. 2).

Next, we select a local frame with center at point O_i and axes $x^{(i)}$, $y^{(i)}$, and $z^{(i)}$, so that $z^{(i)}$ coincides with the axis of actuated revolute joint i. We will refer to that frame as *track frame* i. The constant matrix $\mathbf{R}_i$ defines the orientation of track frame i with respect to the base frame. Finally, let us denote the angle between the $x^{(i)}$ axis and the line O_iA_i by λ_i. This angle is *articular coordinate* i.

We select also a moving frame that is fixed to proximal link i, with center at point A_i and axes $x^{(A_i)}$, $y^{(A_i)}$, and $z^{(A_i)}$, so that the $z^{(A_i)}$-axis is always parallel to the track frame's $z^{(i)}$-axis, and the $x^{(A_i)}$-axis is always along line O_iA_i, pointing away from O_i. We will call this frame *proximal link frame* i. The rotation matrix that transforms coordinates from proximal link frame i to track frame i is a function of λ_i only and will be designated by $\mathbf{R}_{A_i}$.

The mobile platform's position is defined by vector $\mathbf{OC}$, while its orientation is described by a rotation matrix $\mathbf{R}$ that is defined by three Euler angles. The three coordinates of point C and the three Euler angles constitute the so-called *generalized coordinates*. The latter define completely the *pose* (the position and orientation) of the mobile platform.

Finally, we will add the superscript $'$ to a vector when the latter is expressed in the mobile frame, the superscript $^{(i)}$ when the vector is expressed in track frame i, and the superscript $^{(A_i)}$ when the vector is expressed in proximal link frame i. No superscript will mean that the vector is expressed in the base frame.

1.2 Inverse Kinematics

The task of computing the set of articular coordinates from the set of generalized coordinates is referred to as the *inverse kinematic problem* (IKP). Geometrically, for each serial chain, the problem can be regarded as the one of finding the intersection point(s) between a sphere of radius ℓ and center B_i and the track circle. Clearly, depending on the position of point B_i, this problem may have an infinite number of real solutions, two solutions, a single one, or none at all.

The first step in the computation process is to calculate the coordinates of each point B_i, first in the base frame, and then in track frame i:

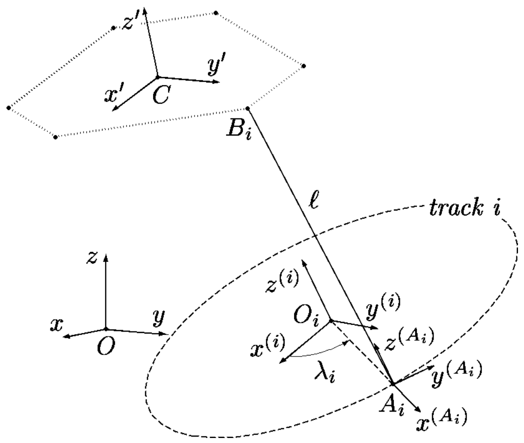

Fig. 2 Serial chain i.

$$\mathbf{OB}_i = \mathbf{OC} + \mathbf{R}\,\mathbf{CB}'_i, \quad (1)$$
$$\mathbf{O}_i\mathbf{B}_i^{(i)} = \mathbf{R}_i^T(\mathbf{OB}_i - \mathbf{OO}_i). \quad (2)$$

Now, by squaring $\mathbf{A}_i\mathbf{B}_i^{(i)} = \mathbf{O}_i\mathbf{B}_i^{(i)} - \mathbf{O}_i\mathbf{A}_i^{(i)}$, we obtain the main equation constituting the IKP:

$$\ell^2 = \|\mathbf{O}_i\mathbf{B}_i^{(i)}\|^2 + r_A^2 - 2\mathbf{O}_i\mathbf{B}_i^{(i)T}\mathbf{O}_i\mathbf{A}_i^{(i)}. \quad (3)$$

If $\mathbf{O}_i\mathbf{B}_i^{(i)T}\mathbf{O}_i\mathbf{A}_i^{(i)} = 0$, i.e. if point B_i lies on the $z^{(i)}$-axis, then eq. (3) degenerates. That is to say, if, in addition, $\|\mathbf{O}_i\mathbf{B}_i^{(i)}\|^2 = \ell^2 - r_A^2$, then the IKP has an infinite number of solutions.

From the definition of articular coordinate λ_i, we have that $\mathbf{O}_i\mathbf{A}_i^{(i)} = r_A\,[\cos\lambda_i, \sin\lambda_i, 0]^T$. Let also the components of vector $\mathbf{O}_i\mathbf{B}_i^{(i)}$ be $x_{B_i}^{(i)}$, $y_{B_i}^{(i)}$, and $z_{B_i}^{(i)}$. Then, eq. (3) reduces to

$$\cos\lambda_i\; x_{B_i}^{(i)} + \sin\lambda_i\; y_{B_i}^{(i)} = \frac{\|\mathbf{O}_i\mathbf{B}_i^{(i)}\|^2 + r_A^2 - \ell^2}{2r_A} \equiv p_i. \quad (4)$$

Now, in order to have a real solution to this equation, the following inequality should hold true:

$$p_i^2 \leq x_{B_i}^{(i)2} + y_{B_i}^{(i)2}. \quad (5)$$

This inequality is equivalent to the distal link's length constraint that will be presented in section 2.1. Unless $p_i^2 = x_{B_i}^{(i)2} + y_{B_i}^{(i)2}$, there exist two real solutions to eq. (4), given by:

$$\lambda_i = 2\tan^{-1}\left(\frac{y_{B_i}^{(i)} + b_i\sqrt{x_{B_i}^{(i)2} + y_{B_i}^{(i)2} - p_i^2}}{p_i + x_{B_i}^{(i)}}\right), \quad (6)$$

where $b_i = \pm 1$ is a so-called *branch index*. As we just mentioned, when inequality (5) turns into an equality, serial chain i is in a singular configuration—a situation that needs to be avoided. Thus, in practice, for each chain, the branch index is set to a constant. As we will define it geometrically in section 2.2, the constraint that b_i should be constant for each chain will be called the serial-chain singularity constraint. It will simply mean that once the manipulator is assembled, distal link i should never be allowed to lie in one plane with the $z^{(i)}$-axis.

The organization of the rest of this paper is as follows. In section 2, we describe the geometric method for obtaining the constant-orientation workspace of a 6-*__R__US* PM. Then, in section 3, we propose the general procedure to be used for computing each vertex space, and then the procedure to be followed for tracing the edges of the constant-orientation workspace. Section 4, presents the main contribution of this paper which is the algorithm for obtaining analytically the intersection points between a general torus and a circle. This algorithm is necessary for computing the edges of the workspace. Examples are then shown in section 5. Finally, suggestions for further work are given in section 6 and conclusions are made in the last section 7.

2 GEOMETRIC MODELING OF THE WORKSPACE

In order to describe a geometric method for computing the constant-orientation workspace, it is necessary to establish geometric models for all the constraints that limit it. The basic idea is first to regard all serial chains as independent and then to consider their interdependence (Gosselin, 1990).

Thus, for a constant orientation of the mobile platform, let us define *vertex space i* as the volume that can be attained by vertex B_i from chain i, ignoring the constraints imposed by all other serial chains. The constraints that determine each vertex space are (i) the distal link's length, (ii) the serial-chain singularity, (iii) the ranges of the base and platform joints, and (iv) the proximal link's length. However, we will explain later why the constraints on the mobile platform joints cannot be geometrically described in the determination of the vertex spaces. Also, as we already mentioned, we assume that the revolute actuators can fully rotate, i.e. there are no actuator limits. Next, we will investigate each constraint individually to finally construct the vertex space.

2.1 Distal Link's Length

In the proximal link's frame, the set of points reachable by B_i is a sphere S_i of radius ℓ and center A_i.

2.2 Serial-Chain Singularity

As we already outlined in section 1.2, an *__R__US* chain is at a singular configuration when a distal link is coplanar with the axis of the corresponding actuated revolute joint, but its vertex B_i does not lie on that axis. In this singularity, the two branches of the inverse kinematics of the serial chain meet and the mobile platform loses one degree of freedom (Gosselin and Angeles, 1990). In addition, if vertex B_i lies on the axis of the actuated revolute joint, then the two branches degenerate to an infinite number.

In any case, since passing through such singularities is undesirable, the motion of each distal link should be restricted so that the angle between vector $\mathbf{A}_i\mathbf{B}_i$ and the proximal link frame's

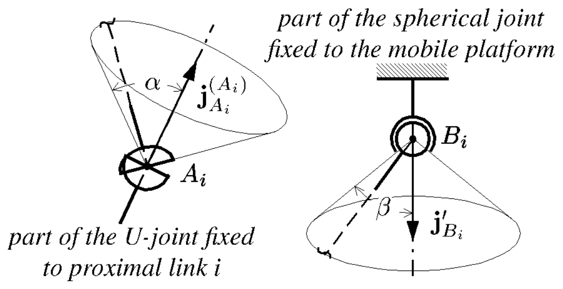
part of the spherical joint fixed to the mobile platform
α
$\mathbf{j}_{A_i}^{(A_i)}$
A_i
B_i
β
$\mathbf{j}'_{B_i}$
part of the U-joint fixed to proximal link i

Fig. 3 Ranges of motion of the passive joints.

$y^{(A_i)}$-axis be always in only one of the two ranges $[0°, 90°)$ (corresponding to $b_i = -1$) or $(90°, 180°]$ (corresponding to $b_i = +1$). Hence, we split sphere S_i by the $x^{(A_i)}z^{(A_i)}$-plane. Depending on the branch index, we take one of the two hemispheres. The great circle formed by the intersection of S_i with that plane will be denoted by $C_{i,N}$.

2.3 Range of a Base Joint

The physical constraints that limit the range of a base joint can be modeled by a general conical surface whose vertex is the center of the joint. We already mentioned that the distal links are attached to the proximal links through U-joints, but, in practice, spherical joints are often used instead. Thus, we choose to model the constraint imposed by the base joint as a circular cone, within which the corresponding distal link is restrained to stay (Fig. 3). If indeed, U-joints are used, then a better model would be a pyramid as used in (Merlet, 1994), which, however, will inevitably make the workspace analysis slightly more complicated.

Let α be the *maximum misalignment angle* of the base joints ($\alpha < 90°$) and let $\mathbf{j}_{A_i}^{(A_i)}$ be the unit vector along the axis of symmetry of the joint at point A_i, expressed in proximal link frame i. This vector is constant when expressed in that frame and depends only on the design of the PM. Then, the allowable region for point B_i consists of a *spherical cap* of radius ℓ and center A_i. The *base circle* of that spherical cap will be designated by $C_{i,A}$.

2.4 Range of a Mobile Platform Joint

The same cone model could be used for the platform spherical joints. Let β be the maximum misalignment angle of the platform joints ($\beta < 90°$) and let $\mathbf{j}'_{B_i}$ be the constant unit vector along the axis of symmetry of the spherical joint with center B_i expressed in the mobile frame. Then, the allowable region for point A_i, referred to the mobile frame, consists of a spherical cap of radius ℓ and center B_i. Let $-\mathbf{j}_{B_i}^{(A_i)}$ be the opposite unit vector, expressed in the proximal link frame i, and defined as:

$$\mathbf{j}_{B_i}^{(A_i)} = \mathbf{R}_{A_i}^T \mathbf{R}_i^T \mathbf{R}\, \mathbf{j}'_{B_i}. \tag{7}$$

Thus, with respect to the proximal link frame, point B_i is located on an equivalent spherical cap of radius ℓ but center A_i. Now, for the 6-*P̲US* PMs (Bonev and Ryu, 1999b), the orientations of the proximal link frames are fixed with respect to the base frame. Thus, when point A_i moves along the linear track, the equivalent spherical cap representing the constraint on spherical joint i remains unchanged in proximal link frame i. This, however, is not true for the 6-*R̲US* PMs, since $\mathbf{R}_{A_i}$ in eq. (7) is not constant. Therefore, the spherical cap has a different orientation in proximal link frame i for each different position of point A_i on the circular track.

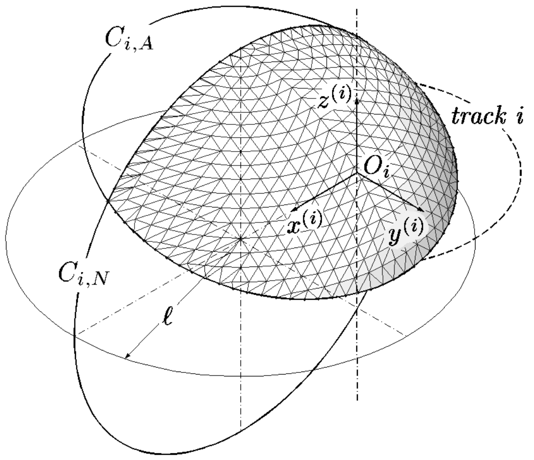

Fig. 4 The allowable spherical region.

Fortunately, simulations performed by the authors on several different designs have shown that the constraint imposed by the spherical joint ranges is much less frequently violated than the constraint on the U-joints. Thus, we ignore this constraint for the sake of describing geometrically the constant-orientation workspace. If this constraint is too tight to be neglected, then the use of a discretization or a numerical method is the only alternative.

2.5 Proximal Link's Length

The *allowable spherical region* for point B_i in proximal link frame i is the intersection of the hemisphere defined in section 2.2 and the spherical cap defined in section 2.3 (Fig. 4). Now, in track frame i, the allowable spatial region for vertex B_i is the volume swept by the allowable spherical region by revolving it about the $z^{(i)}$-axis. This volume is *vertex space* i (Fig. 5). Note that *ignoring the constraints on the mobile platform joints makes each vertex space independent from the orientation of the mobile platform*. Thus, for a given design, we can store the data defining all vertex spaces and avoid computing them for each different orientation of the mobile platform.

After all six vertex spaces have been defined, we must consider the fact that all points B_i are fixed to the mobile platform. Let us call this the *closure constraint*. Since the mobile platform is kept at a constant orientation, then the allowable spatial region for point C—taking into account the restrictions imposed by only serial chain i—is obtained by translating vertex space i along vector $\mathbf{B}_i\mathbf{C}$. Thus, the intersection of all six translated vertex spaces is the constant-orientation workspace of the PM.

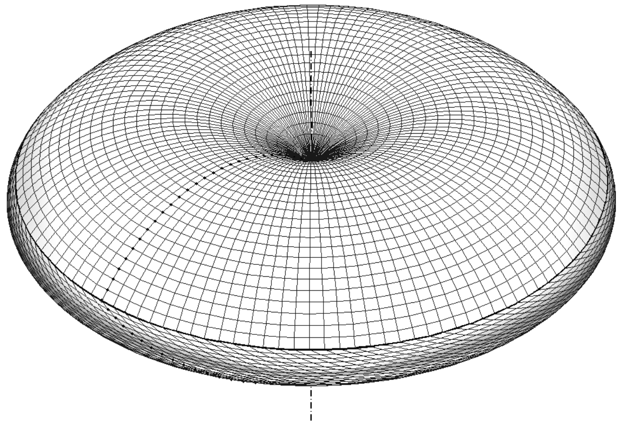

Fig. 5 Vertex space i.

3 IMPLEMENTATION PROCEDURES

Next, we will propose the procedure for determining each vertex space. The idea is to obtain explicitly the *contour* of the allowable spherical region and then to construct the boundary representation of the vertex space as the surface of revolution obtained by revolving that contour. After that, we will consider the problem of obtaining the edges of the constant-orientation workspace.

3.1 Procedure for the Vertex Spaces

Each of the two circles $C_{i,N}$ and $C_{i,A}$, can be defined by a parametric equation, in which the parameter $u_{i,1}$ (respectively $u_{i,2}$) varies from $-\pi$ to $+\pi$. Then, for each vertex space, we calculate the intersection points between the two circles, storing the values of the parameter $u_{i,1}$ (respectively $u_{i,2}$) corresponding to each intersection point. Two different situations may occur depending on the value of $\xi = \cos^{-1}\left([0, -b_i, 0]^T \mathbf{j}_{A_i}^{(A_i)}\right)$.

Case 1: The number of intersection points is 0 or 1;
If $\xi \geq \pi/2 + \alpha$, then stop—vertex space i and, consequently, the constant-orientation workspace do not exist for the current orientation and design. Else, if $\xi \leq \pi/2 - \alpha$, then the contour of the allowable spherical region is the whole circle $C_{i,A}$ and the vertex space constitutes a *general torus*[†], i.e. $u_i \in [-\pi, +\pi]$.

Case 2: The number of intersection points is 2;
If $\pi/2 + \alpha > \xi > \pi/2 - \alpha$, then there exist two distinct intersection points. For each circle, calculate the center point of each of the two arcs connecting the intersection points. If the center point lies on the hemisphere and on the spherical cap, then the arc belongs to the allowable spherical region. Store the arc's range as the ordered couple $\{u_{i,k}^s, u_{i,k}^e\}$, so that if $u_{i,k}^s < u_{i,k}^e$, then $u_{i,k} \in [u_{i,k}^s, u_{i,k}^e]$, else $u_{i,k} \in [u_{i,k}^s, +\pi] \cup (-\pi, u_{i,k}^e]$, where $k = 1, 2$. In this case, the vertex space has two distinct boundary surfaces, one of which is a portion of a right circular torus, and the other, a portion of a general torus.

[†]a surface obtained by revolving a circle about an axis that does not necessarily lie in the plane of the circle (Fichter and Hunt, 1975).

The number of boundary surfaces (1 or 2), the data for circle(s) $C_{i,A}$ (and $C_{i,N}$), and the range limits $\{u_{i,k}^s, u_{i,k}^e\}$ is all that needs to be saved for vertex space i. In addition, in Case 2, we also need to calculate and save the data for the two circles that define the edges of the vertex space. Finally, we modify vector $\mathbf{OO}_i$ that positions the origin of track frame i by adding to it vector $\mathbf{B}_i\mathbf{C}$.

3.2 Procedure for the Workspace

Since each vertex space has up to two boundary surfaces, our problem can be decomposed into a finite number of intersections between surfaces. Each of the surfaces is a parametric toroidal surface depending on two parameters, $u_{i,k} \in [u_{i,k}^s, u_{i,k}^e]$ or $u_{i,k} \in [u_{i,k}^s, +\pi] \cup (-\pi, u_{i,k}^e]$, and $v_{i,k} \in [-\pi, +\pi]$, where i corresponds to the vertex space to which the surface belongs, and $k = 1, 2$. In general, it is not possible to obtain directly analytic (parametric) expressions for the intersection curves. Thus, we proceed as in (Johnstone, 1993) and use circle decomposition to reduce the surface intersection problem to the problem of finding the intersection points between a general torus and a circle.

We initialize as many lists as there are pairs of boundary surfaces. There may be up to 60 *inter-space pairs* (15 pairs of vertex spaces $\times$ 4 pairs of boundary surfaces, each pair coming from different vertex spaces) and 6 *intra-space pairs*, each pair coming from the same vertex space.

To compute the edges of the workspace, we take each inter-space pair of surfaces—one belonging to vertex space i and the other to vertex space j. Then, for the torus to which one of the boundary surfaces belongs, say from vertex space j, we start to increment the parameter v_j from $-\pi$ to π. For each discrete value of v_j, we find the intersection points, in terms of the parameters u_j and u_i, between the corresponding circle and the torus to which the other boundary surface belongs using the algorithm presented in section 4. The next step is to eliminate those solutions that are not within the permissible ranges of u_j and u_i.

Then, we simply calculate the Cartesian coordinates of each point corresponding to a solution for u_j. Each such point lies on the boundaries of vertex spaces i and j. Then, at this point, we solve the IKP for all serial chains except chains i and j, and check for all constraints. The point will lie inside the four vertex spaces if all constraints are satisfied. The remaining points belong to the edges of the constant-orientation workspace and are put into a corresponding list.

Finally, for each intra-space pair of boundary surfaces, say corresponding to chain i, we already know the two circles constituting the intersection curves. Then, we intersect each circle with the (maximum 5×2) boundary surfaces of all vertex spaces except vertex space i. This is done again by using the algorithm presented in section 4. The intersection points divide each circle into a maximum number of 4×10 arcs. For each arc, we calculate its center point, and at this point, we solve the IKP for the five serial chains. If all constraints are satisfied, then the corresponding arc is discretized and put into a corresponding list.

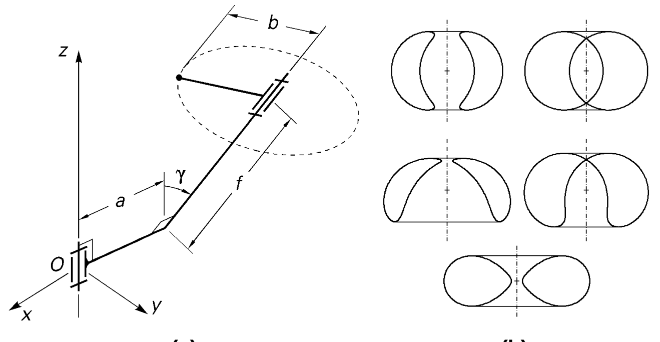

Fig. 6 (a) Tracing a general torus and (b) several possible diametral sections.

4 INTERSECTING A TORUS WITH A CIRCLE

In this section, we will define the equation of a general torus from vertex space i in track frame i (for brevity, we will omit the index $k = 1, 2$). Then, we will present our algorithm for obtaining the intersection points between that torus and a circle. Note that, to our best knowledge, no solutions for this problem have been presented before.

4.1 Algebraic Approach

A detailed study on the general torus was made by Fichter and Hunt (1975) related to the design of spatial linkages. Indeed, the general torus is the 2D locus of a point attached to a body that is joined back to the reference system through an *RR* kinematic chain (Fig. 6a). Its algebraic equation is:

$$\left(x^2+y^2+z^2-a^2-b^2-f^2\right)^2=4a^2\left[b^2-\left(\frac{z-f\cos\gamma}{\sin\gamma}\right)^2\right], \quad (8)$$

where for a right circular torus, $f = 0$ and $\gamma = \pi/2$.

The general torus is a quartic surface that contains the imaginary spherical circle twice, therefore having full *circularity* (Hunt, 1978). Consequently, its intersection curve with a plane is a *bicircular quartic*, including its diametral sections (Fig. 6b). Thus, since a circle has circularity one, there may be at most 4 intersection points between a general torus and a circle.

A circle in space can be defined as the system of two algebraic equations—one of a sphere and one of a plane. Those two equations and eq. (8) can then be solved for the unknowns x, y, and z. In fact, we can even set up the algebraic equation for the other toroidal surface and then trace the intersection curves using some surface intersection algorithm for algebraic (implicit) surfaces. However, such an approach suffers from disadvantages such as the necessity of determining starting points and the difficulties in tracing the different branches of the intersection curves (Patrikalakis and Prakash, 1990). Furthermore, in the area of mechanism design, a parametric approach is much more relevant and may suggest other applications than the one discussed here, hence, our decision to use the parametric approach.

4.2 Parametric Approach

A circle in space can be defined by its radius r, coordinates p_x, p_y, and p_z of its center, and unit vector along the axis of symmetry defined by the angles θ and ϕ, such that its equation is

$$\mathbf{C}^{(i)}(u_i)=\begin{bmatrix} c_\phi & -s_\phi & 0 \\ s_\phi & c_\phi & 0 \\ 0 & 0 & 1 \end{bmatrix}\begin{bmatrix} c_\theta & 0 & s_\theta \\ 0 & 1 & 0 \\ -s_\theta & 0 & c_\theta \end{bmatrix}\begin{bmatrix} r\cos u_i \\ r\sin u_i \\ 0 \end{bmatrix}+\begin{bmatrix} p_x \\ p_y \\ p_z \end{bmatrix}, \quad (9)$$

where $c_\phi \equiv \cos\phi$, $s_\phi \equiv \sin\phi$, etc., θ is the angle between the $z^{(i)}$-axis and the unit vector, and ϕ is the angle between the $x^{(i)}$-axis and the projection of the unit vector onto the $x^{(i)}y^{(i)}$-plane. Thus, for example, for circle $C_{i,N}$, $r = \ell$, $p_x = r_A$, $p_y = p_z = 0$, and $\theta = \phi = \pi/2$.

Each general torus is generated by revolving the corresponding circle, whose equation was derived above, about the track's $z^{(i)}$-axis. Therefore, the parametric equation of the resulting torus with respect to track frame i is

$$\mathbf{S}^{(i)}(u_i, v_i) = \begin{bmatrix} \cos v_i & -\sin v_i & 0 \\ \sin v_i & \cos v_i & 0 \\ 0 & 0 & 1 \end{bmatrix}\mathbf{C}^{(i)}(u_i), \quad (10)$$

where $v_i \in [-\pi, +\pi]$.

The parametric equation for a general torus, $\mathbf{S}^{(j)}(u_j, v_j)$, from vertex space j expressed in track frame j has exactly the same form as the one defined by eq. (10). Setting v_j equal to a constant, $\mathbf{S}^{(j)}(u_j, v_j) \equiv \mathbf{C}^{(j)}(u_j)$ becomes the equation of a circle. Finally, $\mathbf{C}^{*(i)}(u_j) = \mathbf{OO}_j - \mathbf{OO}_i + \mathbf{R}_i^T\mathbf{R}_j\mathbf{C}^{(j)}(u_j)$ will be the expression for that circle referred to track frame i, where vectors $\mathbf{OO}_i$ and $\mathbf{OO}_j$ are the modified positions of the origins of track frames i and j (recall section 3.1).

Obviously, one can find equivalent values θ^*, ϕ^*, p_x^*, p_y^*, and p_z^*, such that $\mathbf{C}^{*(i)}(u_j)$ can be written in exactly the same form as in eq. (9). Note, however, that due to the rotation defined by $\mathbf{R}_i^T\mathbf{R}_j$, the permissible range of the variable u_j should be modified by a certain offset, depending on $\mathbf{R}_i^T\mathbf{R}_j$ and the value of v_j. Hence, the matrix equation that needs to be solved is

$$\mathbf{S}^{(i)}(u_i, v_i) = \mathbf{C}^{*(i)}(u_j). \quad (11)$$

The above is a system of three coupled sine-cosine polynomial equations in the three unknowns u_i, v_i, and u_j. Now, the first part of the solution can be applied for the intersection of any surface of revolution with any spatial curve. The idea is that the distance from the origin to any point on the surface of revolution is dependent only on the parameter u_i. Thus, we can obtain an equation that does not contain v_i by writing

$$\|\mathbf{S}^{(i)}(u_i, v_i)\|^2 = \|\mathbf{C}^{*(i)}(u_j)\|^2, \quad (12)$$

where $\|\cdot\|$ is the Euclidean norm. In our case, the above equation is not only free of v_i but also has a rather simple form:

$$A\sin u_i + B\cos u_i + C\sin u_j + D\cos u_j + E = 0, \quad (13)$$

where A, B, C, D, and E are constants given in Appendix A. The other equation that is also free of v_i is simply the last equation of the system of three equations (11). The z-component of $\mathbf{S}^{(i)}(u_i, v_i)$ is clearly not dependent on v_i, since $z^{(i)}$ is the axis of revolution. Indeed, in our case, the equation is:

$$F \cos u_i + G \cos u_j + H = 0, \tag{14}$$

where F, G, and H are again constants given in Appendix A.

The second step of the solution process is to solve eqs. (13) and (14) for $\sin u_i$ and $\cos u_i$ obtaining:

$$\sin u_i = \frac{(BG-DF)\cos u_j - CF \sin u_j + BH - EF}{AF}, \tag{15}$$

$$\cos u_i = -\frac{G \cos u_i + H}{F}. \tag{16}$$

Then, from the identity $\sin^2 u_i + \cos^2 u_i = 1$, we may obtain the following equation which is dependent only on u_j:

$$J \sin u_j + K \cos u_j + L \sin u_j \cos u_j + M \cos^2 u_j + N = 0 \tag{17}$$

where

$$\begin{aligned}
J &= 2\,(CEF^2 - BCFH), \\
K &= 2\,(A^2GH + DEF^2 + B^2GH - BEFG - BDFH), \\
L &= 2\,(CDF^2 - BCFG), \\
M &= (DF - BG)^2 + (AG)^2 - (CF)^2, \\
N &= (BH - EF)^2 + (CF)^2 + (AH)^2 - (AF)^2.
\end{aligned}$$

Next, we perform the tangent-half-angle substitution:

$$\sin u_j = \frac{2t_j}{1+t_j^2}, \quad \cos u_j = \frac{1-t_j^2}{1+t_j^2}, \tag{18}$$

where $t_j = \tan(u_j/2)$. After substituting the above identities in eq. (17), multiplying by $(1+t_j^2)^2$, and rearranging, we obtain:

$$q_4 t_j^4 + q_3 t_j^3 + q_2 t_j^2 + q_1 t_j + q_0 = 0, \tag{19}$$

where

$$\begin{aligned}
&q_4 = M + N - K,\ q_3 = 2(J - L),\ q_2 = 2(N - M), \\
&q_1 = 2(J + L),\ q_0 = N + K + M.
\end{aligned}$$

The solutions for t_j may be found analytically by solving the polynomial of degree 4 in eq. (19). Then, for each real solution for t_j, we obtain the corresponding value for u_j using $u_j = 2\tan^{-1}(t_j)$. Note that $\tan(u_j/2)$ is not defined at $u_j = \pi$, and indeed, if the latter is a solution to eq. (17), then $q_4 = 0$. Thus, if $q_4 = 0$, we add the solution $u_j = \pi$. Finally, we substitute the values for u_j into eqs. (15) and (16) and solve for u_i.

The complete solution of the system of eqs. (11) is an interesting problem involving a large number of particular cases (singularities). Indeed, this is exactly what should have been done if our task were the solution to the IKP of an *RRSR* kinematic chain. However, in our case we are not interested in the values of v_i, since we have assumed that there are no actuated limits. Hence, the only cases that need to considered are those that may prevent us from finding unique solutions for u_i and u_j.

In our particular problem, the circle that generates the general torus does not have an arbitrary position and orientation. Thus, some of the cases will drop out and should not be considered. Indeed, the constant $A = p_x \sin\phi - p_y \cos\phi$ is equal to zero if and only if the axis of the generating circle intersects the $z^{(i)}$-axis. However, this could happen if and only if $\mathbf{j}_{A_i}^{(A_i)} = [\pm 1, 0, 0]^T$, in which case $B = 0$ and the torus degenerates to a doubly-covered spherical ring (or even a sphere).

Similarly, $F = -r\sin\theta$ is equal to zero if and only if $\mathbf{j}_{A_i}^{(A_i)} = [0, 0, \pm 1]^T$ in which case the torus degenerates to a doubly-covered planar ring or annulus. If also $G = r^* \sin\theta^*$ and $H = p_z - p_z^*$ are equal to zero, than the circle from vertex space j lies in the plane of the degenerated torus which could lead to having an arc as intersection rather than distinct points.

To simplify our task, we will eliminate the above degenerate cases by imposing the requirement that $\mathbf{j}_{A_i}^{(A_i)} \neq [\pm 1, 0, 0]^T$ and $\mathbf{j}_{A_i}^{(A_i)} \neq [0, 0, \pm 1]^T$. Indeed, installing the U-joints in such a way that $\mathbf{j}_{A_i}^{(A_i)}$ is along the $x^{(A_i)}$- or $z^{(A_i)}$-axis is unlikely to lead to an optimal workspace.

Finally, it remains to consider the peculiar case in which $C = D = G = 0$ which geometrically corresponds to the circle from vertex space j lying in a plane parallel to the $x^{(i)}y^{(i)}$-plane and with center on the $z^{(i)}$-axis. In this case, eq. (17) degenerates to $N = 0$, so if it holds true, then the intersection we look for is the entire circle from vertex space j. If this happens, then we discretize the arc of the circle defined by the permissible range of u_j into a finite number of points.

Hence, in any case we end up with a finite number of intersection points (most frequently less than or equal to 4) defined by the parameters u_j and u_i.

5 EXAMPLES

To illustrate our geometric method, we take as an example a 6-*R̲US* PM whose data is given in Appendix B. In our implementation, we adopt the choice of *modified Euler angles* introduced in (Bonev and Ryu, 1999a) to represent the orientation of the mobile platform. For this choice, we rotate first the mobile platform about the base z-axis by an angle $-\phi$, then about the base y-axis by an angle θ, then about the base z-axis by an angle ϕ, and finally about the mobile z'-axis by an angle ψ.

Defined in this way, angle ψ is the *roll angle*, angle θ is the *tilt angle*, and angle ϕ is the angle between the base x-axis and the projection of the mobile z'-axis onto the base xy-plane. These angles have the property that the constant-orientation workspace for a fixed direction of the mobile z'-axis (ϕ, θ) tends to be largest at $\psi = 0°$ (for axisymmetric designs as the one used in the present example).

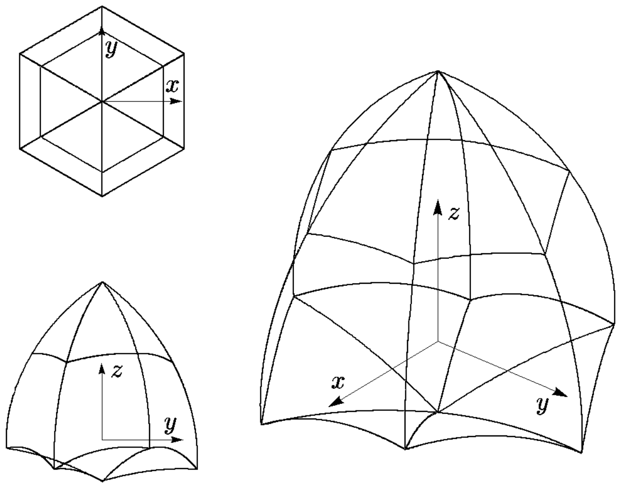

Fig. 7 The workspace of the 6-RUS PM for the reference orientation (edges displayed only).

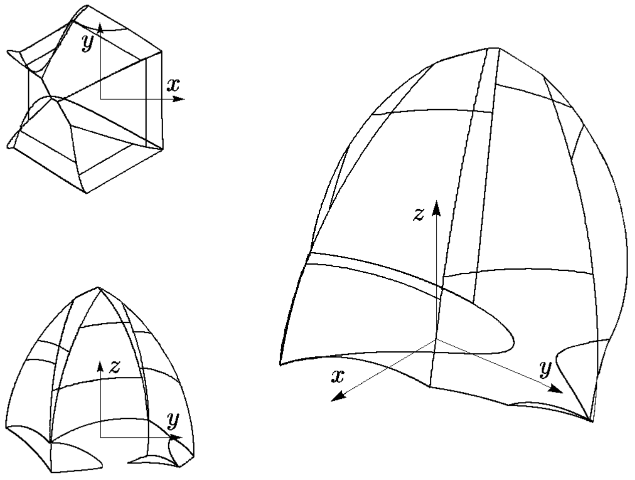

Fig. 8 The workspace of the 6-RUS PM for a tilt angle of 10° (edges displayed only).

The proposed methodology was implemented in MATLAB 5, taking full advantage of the newly introduced data structures and cell arrays. Two examples of the constant-orientation workspace of the 6-$\underline{R}$*US* PM are presented here. The first one is at the *reference orientation* (Fig. 7), $\phi = \theta = \psi = 0°$, while the second one is at a tilted orientation (Fig. 8), $\phi = \psi = 0°$ and $\theta = 10°$. In the first two figures, only the edges are displayed as computed using the algorithm proposed in this paper. The nearly-horizontal edges that are situated somewhere in the middle of the workspace are circular arcs coming from the intra-space pairs of boundary surfaces. In the second example, curves from all types of inter-space and intra-space pairs of boundary surfaces are present. In this example, we also have a large patch of general torus which accounts for the absence of an expected edge, as seen in the top view in Fig. 8.

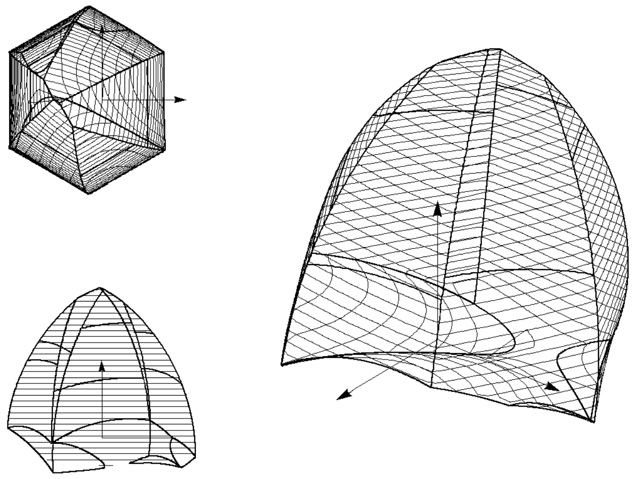

Fig. 9 The workspace of the 6-RUS PM for a tilt angle of 10° (edges and horizontal cross-sections).

Finally, we used a discretization method for computing the horizontal cross-sections of the constant-orientation workspace. This was, naturally, done to test our method but also to compare the quality of the two types of visualization techniques. In Fig. 9, the same example as the one in Fig. 8 is presented again but this time together with a set of equally-spaced horizontal cross-sections. It is not difficult to imagine that a representation of horizontal slices only would have been quite poor. On the other hand, edges only do not give a perfect impression of the workspace either, though changing interactively the viewpoint on the display is quite helpful. In addition, the computation of the edges does not allow the computation of the workspace volume which is an important design criterion. Thus, it is advisory to compute and visualize both the edges and the horizontal slices.

6 FURTHER WORK AND SUGGESTIONS

Having computed the edges of the constant-orientation workspace, its boundary surfaces might be extracted from the vertex spaces to allow a better visualization, similar to the one obtained by Bonev and Ryu (1999b) by using CATIA. Alternatively, there are programming libraries (quite expensive though) that allow directly the numerical computation of intersections between solids. So, if such a library is used, the use of the algorithm proposed in section 4 is avoided.

Another common alternative is finding horizontal cross-sections of the workspace. Thus, once the vertex spaces are found, we need to find the intersection curves between each vertex space and the horizontal plane in which a given workspace slice is to be computed. This time, the most appropriate method for representing the intersection curves is to use the algebraic form by simply substituting the equation of the plane in the algebraic equation of the general torus, eq. (8). As mentioned before, these planar curves are bicircular quartics. Thus, the workspace slice can be obtained by intersecting the six quartics coming from the six vertex spaces, for which a numerical method may be used. Best results, however, can be obtained by computing and displaying both the workspace edges and the horizontal cross-sections.

Note that for some of the existing 6-$\underline{R}US$ PMs, the computation of the horizontal cross-sections is a simple task. Namely, for Hexel's "Rotary Hexapod" and for one of the motion systems of Servos & Simulation Inc., the axes of the actuated revolute joints are all vertical, and hence, the cross-sections are simply concentric circles.

On the other hand, for some 6-$\underline{R}US$ PMs, other constraints may exist. For example, for the same "Rotary Hexapod", all U-joints are mounted on sliders that move on the same circular guide. Thus, we must consider also a so-called *slider-slider constraint*, ensuring that no sliders collide.

Finally, it is interesting to mention the work done by Zabalza et al. (1999) in which the authors purposefully seek the configurations of a 6-$\underline{R}US$ PM in which all serial chains are at singularity. The idea is that the precision of the PM in such configurations is much higher. In the light of the current study, we may roughly state that the precision of the PM tends to be higher near the edges of its constant-orientation workspace.

7 CONCLUSIONS

The original algorithm for the intersection between a general torus and a circle that we have proposed nearly concludes the theoretical research in the area of constant-orientation workspace computation for 6-$\underline{R}US$ PMs. The proposed intersection algorithm may also be of interest for the CAD community as well as for the researchers working in the area of singularity and workspace analysis of serial robots.

REFERENCES

Benea, R., 1996, "Contribution à l'étude des robots pleinement parallèles de type 6 R-RR-S," *Ph.D. Thesis*, Université de Savoie, Annecy, France.

Bonev, I. A., and Ryu, J., 1999a, "Orientation Workspace Analysis of 6-DOF Parallel Manipulators," *Proceedings of the 1999 ASME Design Engineering Technical Conferences*, Las Vegas, NV, USA, DETC99/DAC-8646.

Bonev, I. A., and Ryu, J., 1999b, "Workspace Analysis of 6-PRRS Parallel Manipulators Based on the Vertex Space Concept," *Proceedings of the 1999 ASME Design Engineering Technical Conferences*, Las Vegas, NV, USA, DETC99/DAC-8647.

Chi, Y. L., 1999, *Systems and Methods Employing a Rotary Track for Machining and Manufacturing*, WIPO Patent No. WO 99/38646.

Chrisp, A., and Gindy, N., 1999, "Parallel Link Machine Tools: Simulation, Workspace Analysis and Component Positioning," *Parallel Kinematic Machines: Theoretical Aspects and Industrial Requirements*, C. R. Boër, L. Molinari-Tosatti, and K. S. Smith, eds., Springer-Verlag, pp. 245–256.

Fichter, E. F., and Hunt, K. H., 1975, "The Fecund Torus, its Bitangent-Circles and Derived Linkages," *Mechanism and Machine Theory*, Vol. 10, pp. 167–176.

Gosselin, C. M., 1990, "Determination of the Workspace of 6-DOF Parallel Manipulators," *ASME Journal of Mechanical Design*, Vol. 112, September, pp. 331–336.

Gosselin, C. M., and Angeles, J., 1990, "Singularity Analysis of Closed-Loop Kinematic Manipulators," *IEEE Transactions on Robotics and Automation*, Vol. 6, No. 3, June, pp. 281–290.

Gosselin, C. M., Lavoie, E., and Toutant, P., 1992, "An Efficient Algorithm for the Graphical Representation of the Three-Dimensional Workspace of Parallel Manipulators," *Proceedings of the ASME 22nd Biennial Mechanisms Conference*, Scottsdale, AZ, USA, Vol. 45, pp. 323–328.

Hunt, K. H., 1983, "Structural Kinematics of In-Parallel-Actuated Robot Arms," *Journal of Mechanisms, Transmissions, and Automation in Design*, Vol. 105, December, pp. 705–712.

Hunt, K. H., 1978, *Kinematic Geometry of Mechanisms*, Oxford University Press, Oxford, pp. 270–273.

Johnstone, J. K., 1993, "A New Intersection Algorithm for Cyclides and Swept Surfaces Using Circle Decomposition," *Computer Aided Geometric Design*, Vol. 10, No. 1, pp. 1–24.

Merlet, J.-P., 1994, "Détermination de l'espace de travail d'un robot parallèle pour une orientation constante," *Mechanism and Machine Theory*, Vol. 29, No. 8, pp. 1099–1113.

Merlet, J.-P., 1997, *Les robots parallèles*, 2nd ed., Hermès, Paris.

Merlet, J.-P., and Gosselin, C. M., 1991, "Nouvelle architecture pour un manipulateur parallèle à 6 degrés de liberté," *Mechanism and Machine Theory*, Vol. 2, No. 26, pp. 77–90.

Patrikalakis, N. M., and Prakash, P. V., 1990, "Surface Intersections for Geometric Modeling," *ASME Journal of Mechanical Design*, Vol. 112, March, pp. 100–107.

Pierrot, F., Uchiyama, M., Dauchez, P., and Fournier, A., 1990, "A New Design of a 6-DOF Parallel Robot," *Journal of Robotics and Mechatronics*, Vol. 2, No. 4, pp. 308–315.

Takeda, Y., Funabashi, H., and Ichimaru, H., 1997, "Development of Spatial In-Parallel Actuated Manipulators with Six Degrees of Freedom with High Motion Transmissibility," *JSME International Journal*, Series C, Vol. 40, No. 2, pp. 299–308.

Zabalza, I., Pintor, J. M., Ros, J., and Jimenez, J. M., 1999, "Evaluation of the 64 'Insensitivity' Positions for a 6-RKS Hunt-Type Parallel Manipulator," *Proceeding of the Tenth World Congress on the Theory of Machine and Mechanisms*, Oulu, Finland, June 20-24, pp. 1152–1157.

APPENDIX A

The coefficients of eqs. (13) and (14) are given as

$$\begin{aligned}
A &= 2r(c_\phi p_y - s_\phi p_x),\\
B &= 2r(c_\phi c_\theta p_x - s_\theta p_z + s_\phi c_\theta p_y),\\
C &= -2r^*(c_{\phi^*} p_y^* - s_{\phi^*} p_x^*),\\
D &= -2r^*(c_{\phi^*} c_{\theta^*} p_x^* - s_{\theta^*} p_z^* + s_{\phi^*} c_{\theta^*} p_y^*),\\
E &= p_x^2 + p_y^2 + p_z^2 + r^2 - (p_x^{*2} + p_y^{*2} + p_z^{*2} + r^{*2}),\\
F &= -r s_\theta, \quad G = r^* s_{\theta^*}, \quad H = p_z - p_z^*.
\end{aligned}$$

APPENDIX B

Table 1 shows the data for the 6-*R̲US* PM used for the examples in section 5, where $\mathbf{n}_i$ is the unit vector along the $z^{(i)}$-axis of track frame i, expressed in the base frame. In addition, $\ell = 150$ mm, $r_A = 90$ mm, $\alpha = 70°$, and $b_i = (-1)^{i+1}$ ($i = 1, \ldots, 6$).

Table 1 Geometry of the 6-RUS PM (in [mm]).

i	$\mathbf{OO}_i$	$\mathbf{n}_i$	$\mathbf{CB}'_i$	$\mathbf{j}_{A_i}^{(A_i)}$
1	$\begin{bmatrix} 100.000 \\ -173.205 \\ 0.000 \end{bmatrix}$	$\begin{bmatrix} -0.500 \\ 0.866 \\ 0.000 \end{bmatrix}$	$\begin{bmatrix} 70.707 \\ -84.265 \\ 50.000 \end{bmatrix}$	$\begin{bmatrix} 0.380 \\ -0.912 \\ 0.152 \end{bmatrix}$
2	$\begin{bmatrix} 200.000 \\ 0.000 \\ 0.000 \end{bmatrix}$	$\begin{bmatrix} -1.000 \\ 0.000 \\ 0.000 \end{bmatrix}$	$\begin{bmatrix} 108.329 \\ -19.101 \\ 50.000 \end{bmatrix}$	$\begin{bmatrix} 0.380 \\ 0.912 \\ 0.152 \end{bmatrix}$
3	$\begin{bmatrix} 100.000 \\ 173.205 \\ 0.000 \end{bmatrix}$	$\begin{bmatrix} -0.500 \\ -0.866 \\ 0.000 \end{bmatrix}$	$\begin{bmatrix} 37.622 \\ 103.366 \\ 50.000 \end{bmatrix}$	$\begin{bmatrix} 0.380 \\ -0.912 \\ 0.152 \end{bmatrix}$
4	$\begin{bmatrix} -100.000 \\ 173.205 \\ 0.000 \end{bmatrix}$	$\begin{bmatrix} 0.500 \\ -0.866 \\ 0.000 \end{bmatrix}$	$\begin{bmatrix} -37.622 \\ 103.366 \\ 50.000 \end{bmatrix}$	$\begin{bmatrix} 0.380 \\ 0.912 \\ 0.152 \end{bmatrix}$
5	$\begin{bmatrix} -200.000 \\ 0.000 \\ 0.000 \end{bmatrix}$	$\begin{bmatrix} 1.000 \\ 0.000 \\ 0.000 \end{bmatrix}$	$\begin{bmatrix} -108.329 \\ -19.101 \\ 50.000 \end{bmatrix}$	$\begin{bmatrix} 0.380 \\ -0.912 \\ 0.152 \end{bmatrix}$
6	$\begin{bmatrix} -100.000 \\ -173.205 \\ 0.000 \end{bmatrix}$	$\begin{bmatrix} 0.500 \\ 0.866 \\ 0.000 \end{bmatrix}$	$\begin{bmatrix} -70.707 \\ -84.265 \\ 50.000 \end{bmatrix}$	$\begin{bmatrix} 0.380 \\ 0.912 \\ 0.152 \end{bmatrix}$