Trajectory Verification in the Workspace for Parallel Manipulators

Jean-Pierre MERLET
INRIA Sophia-Antipolis
BP 93, 06902 Sophia-Antipolis,France

Abstract

We present a fast algorithm for solving the problem of the verification of a trajectory for a 6-DOF parallel manipulator with respect to its workspace: i.e., given two different postures for the end-effector, is the straight line joining these two postures in the parameters space fully inside the workspace ? This algorithm is based on the analysis of the algebraic inequalities describing the constraints on the workspace and provides a technique for computing those parts of the trajectory that lie outside the workspace. This method is exact if the orientation of the end-effector is kept constant along the trajectory and approximate if the orientation is allowed to vary.

1 Introduction

A general 6-DOF parallel manipulator is shown in Fig. 1, which has six linear adjustable actuators connecting a mobile platform and a base platform. As the length of the actuators change, an end-effector attached to the mobile platform can be moved in 6-DOF space. Each link is connected to the base

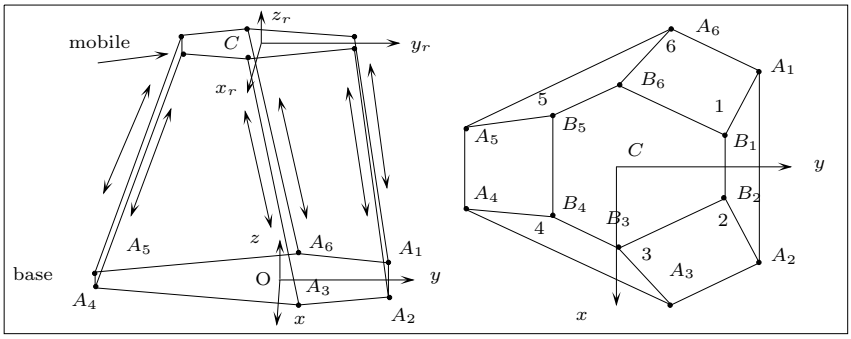

Figure 1: A 6-DOF parallel manipulator.

platform through a universal joint and to the mobile platform through a ball-and-socket joint.

The workspace of a parallel manipulator is limited owing to three types of constraints:

- Limited range for the link lengths. The minimum length of link i will be denoted ρ^i_{min} and the maximum length ρ^i_{max}
- Mechanical limits on the passive joints (universal joints and ball and socket joints).
- Link interference

The problem of determining the workspace of parallel manipulators has been addressed by many authors. The positioning workspace (i.e., the region of the three-dimensional Cartesian space that can be attained by a manipulator with a given orientation) has been described through methods based on a complete discretization of the Cartesian space [**?**].

A geometric approach for determining the positioning workspace border owing to the limited range of the links lengths has been proposed by Gosselin and coworkers [**?**]. This approach has been extended to take into account all the constraints limiting the workspace [**?**, **?**], enabling fast, exact calculation of the border of the positioning workspace.

A variety of methods based on discretization have been proposed for describing the orientation workspace [**?**]. Contrasting these techniques is the geometric approach described in Merlet cite1993a, which provides a strategy for computing and representing the orientation workspace when a point of the end-effector is fixed.

To the best of our knowledge no one has addressed the problem of verifying a trajectory with respect to the workspace: given two points in the parameter's space (i.e., two postures for the end effector), is the straight line joining these two points fully inside the workspace of the robot ? Clearly this problem is very important for the motion planning of a parallel manipulator.

Let the reference frame (O, x, y, z) be fixed in the base platform and the relative frame (C, x_r, y_r, z_r) be embedded in the mobile platform. A subscript $_r$ will be used for vectors whose coordinates are expressed in the relative frame. The following variables will be employed in this article:

- A_i, B_i: center of the passive joint of link i attached to the base of the robot and to the mobile platform.
- ψ, θ, ϕ: three angles defining the orientation of the end effector
- $\mathbf{R}$: rotation matrix relating the relative frame to the fixed frame.
- ρ_i: length of link i

2 Trajectory with a fixed orientation

In this section we will assume that the orientation of the end effector is kept constant all along the trajectory. Let us define the start and goal points of the trajectory as M_1, M_2 and let C be a point on

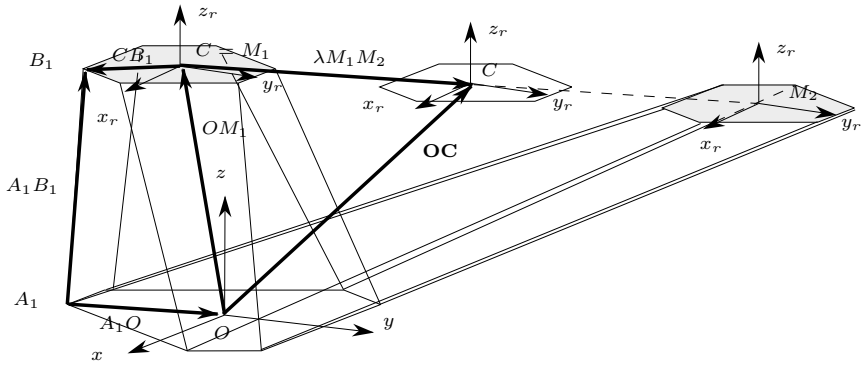

Figure 2: The various vectors used in the equations.

the trajectory (i.e., a point lying on the segment M_1M_2). Any such point (Fig. 2) can be defined as:

$$\mathbf{OC} = \mathbf{OM_1} + \lambda \mathbf{M_1 M_2} \quad \text{with } \lambda \in [\mathbf{0}, \mathbf{1}] \tag{1}$$

2.1 Limitation on the Link Lengths

The length of a link for any point on the trajectory between M_1 and M_2 (Fig. 2) is the euclidian norm of vector $\mathbf{AB}$. We have:

$$\mathbf{AB} = \mathbf{AO} + \mathbf{OC} + \mathbf{CB}, \tag{2}$$

where $\mathbf{CB} = \mathbf{RCB_r}$ is a constant vector. The square of the link length ρ is given by:

$$\begin{aligned} \rho^2 &= \mathbf{AB}.\mathbf{AB^T} \\ &= ||\mathbf{AO}||^2 + ||\mathbf{OC}||^2 + ||\mathbf{CB}||^2 + \mathbf{2}(\mathbf{AO} + \mathbf{CB}).\mathbf{OC^T} + \mathbf{2AO}.\mathbf{CB^T}. \end{aligned} \tag{3}$$

Using equation (1), this equation can be rewritten as a second-order equation:

$$\rho^2 = a\lambda^2 + b\lambda + c, \tag{4}$$

with:

$$a = ||\mathbf{M_1M_2}||^2 > \mathbf{0},\ \mathbf{bb} = \mathbf{2}(\mathbf{AM_1} + \mathbf{CB}).\mathbf{M_1M_2^T},\ \mathbf{c} = ||\mathbf{AM_1} + \mathbf{CB}||^2. \tag{5}$$

We consider now the equation $a\lambda^2 + b\lambda + c - \rho_{max}^2$. Because $a > 0$, the equation will be positive for all λ if this equation has no root. Consequently, in that case the link length will be greater than ρ_{max} on the whole trajectory.

Assume now that the equation has two roots x_1, x_2 sorted by value. Because $a > 0$, the equation will be positive for λ in $]-\infty, x_1[,]x_2, +\infty[$. The intersection of these intervals with the interval $[0, 1]$ will define the intervals on λ (i.e., the portion of the trajectory) for which the link length will be greater than ρ_{max}.

We perform this analysis for the 6 link lengths, and the union I_{max} of all the obtained intervals will define the portions of the trajectory for which at least one length will be greater than ρ_{max}.

Now we consider the equation $a\lambda^2 + b\lambda + c - \rho_{min}^2$. Because $a > 0$, the equation will be negative for all λ if this equation has no root. Consequently, in that case the link length will be greater than ρ_{min} on the whole trajectory.

Assume now that the above equation has two real roots x_1, x_2 sorted by value. Because $a > 0$, the equation will be negative for λ in $]x_1, x_2[$, and the intersection of this interval with $[0, 1]$ will define the interval for which the link length will be less than ρ_{min}.

We perform this analysis for the 6 link lengths, and the union I_{min} of all the obtained intervals will define the portions of the trajectory where at least one link length will be less than ρ_{min}.

The union of I_{max}, I_{min} will give the portions of the trajectory where at least one link length will be outside its allowed range. If the union is empty the trajectory is fully inside the workspace of the manipulator.

To verify a trajectory with respect to the link length limits, we have to analyze the roots of two second-order equations with respect to the interval [0,1]. An analysis of the coefficients of these equations [**?**] establishes the following rules:

Rule 1: *If, at points M_1, M_2 the link length is less than ρ_{max}, then the link length will be less than ρ_{max} on the whole trajectory.*

Rule 2: *If, at points M_1, M_2 the link length is less than ρ_{min}, then the link length will be less than ρ_{min} on the whole trajectory.*

Rule 3: *If, at points M_1, M_2, the link length is greater than ρ_{min} and $\mathbf{M_1M_2^T}$. $(\mathbf{AM_1} + \mathbf{CB})$, $\mathbf{M_1M_2^T}$. $(\mathbf{AM_2} + \mathbf{CB})$ have the same sign, then the link length will be greater than ρ_{min} on the whole trajectory.*

Rule 4: *If, at points M_1, M_2, the link length is greater than ρ_{max} and $\mathbf{M_1M_2^T}$. $(\mathbf{AM_1} + \mathbf{CB})$, $\mathbf{M_1M_2^T}$. $(\mathbf{AM_2} + \mathbf{CB})$ have the same sign, then the link length will be greater than ρ_{max} on the whole trajectory.*

Rule 5: *Let α be the angle between the vectors $\mathbf{AM_1} + \mathbf{CB}$ and $\mathbf{M_1M_2}$, and let $\rho(M_1)$ be the link length when the end-effector is at M_1. If the link length is greater than ρ_{max} for the end-effector locations M_1, M_2 and $\rho_{max}^2 < \rho^2(M_1)\sin^2\alpha$, then the link length is greater than ρ_{max} on the whole trajectory.*

Rule 6: *If, for the end-effector locations M_1, M_2 the link length is greater than ρ_{min}, then the link length will be less than ρ_{min} for some points on the trajectory if and only if:*

$$\mathbf{M_1M_2^T}.(\mathbf{AM_1} + \mathbf{CB}) < \mathbf{0} \quad \mathbf{M_1M_2^T}.(\mathbf{AM_2} + \mathbf{CB}) > \mathbf{0} \quad \boldsymbol{\rho}^{\mathbf{2}}_{\mathbf{min}} > \boldsymbol{\rho}^{\mathbf{2}}(\mathbf{M_1}) \sin^{\mathbf{2}} \alpha \tag{6}$$

2.2 Mechanical Limits on the Passive Joints

The mechanical limits on joints such as universal joints or ball-and-socket joints imply that the link that is connected to the joint cannot fully rotate and span a full sphere. Therefore, the link spans a volume that is defined by its border surface. As stated in Merlet [**?**, **?**] we assume that this surface can be approximated by a pyramid with planar faces. For the joints attached to the base, the center of this pyramid is located at A (Fig. 3-a). Although a cone model will seem to be more suitable for the ball-and-socket joint limit, very few commercially available joints have the feature of allowing a full cone to be described. Furthermore, a cone model will greatly increase the complexity of the algorithm, as we will have to deal with an inequality of degree at least four instead of two or one. Finally, a cone can be approximated by a pyramid with an appropriate number of faces.

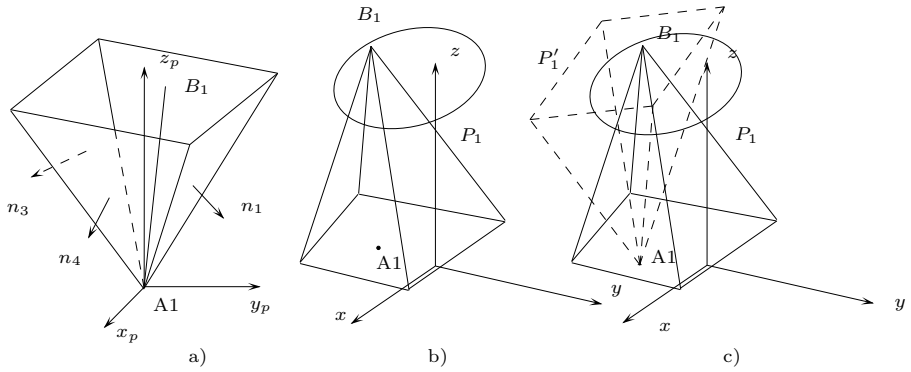

Figure 3: An example of modelization of the constraints on the passive joints. For the base joints, if the mechanical limits of the joints are satisfied, then the link A_1B_1 is inside the volume delimited by the pyramid (a) (here a pyramid with 4 faces). For the mobile platform joint, if the mechanical limits on the joints are satisfied, the point A_1 lies inside a pyramid (b). From this pyramid we deduce an equivalent pyramid P_1' centered at A_1 such that if the mechanical limits on the joint are satisfied, then point B_1 lie inside P_1' (c).

As for the constraint on the passive joints attached to the end-effector, we may use the same model. We define a pyramid P_i centered in B_i such that if the constraint on the joint at B is satisfied, then point A_i will lie inside the pyramid (Fig. 3-b). We then define another pyramid that will be called *an equivalent pyramid* P_i' to P_i, with center A_i such that if A_i lies inside P_i then B_i lies inside P_i' (Fig. 3-c).

Let $\mathbf{n_i}$ be the external normal of the ith face of the pyramid associated with the joint attached to the base. If the point B lies inside the pyramid, we have:

$$\mathbf{AB}.\mathbf{n_i^T} \leq \mathbf{0}. \tag{7}$$

By using equation (1), this equation can be rewritten as:

$$\lambda a_1 + b_1 \leq 0, \tag{8}$$

with $a_1 = \mathbf{M_2M_1}.\mathbf{n_i^T}$, $b_1 = (\mathbf{OM_1} + \mathbf{AO} + \mathbf{CB}).\mathbf{n_i^T}$. This inequality will not be satisfied if $a_1 > 0$ and $\lambda \in]-b_1/a_1, +\infty[$ or if $a_1 < 0$ and $\lambda \in]-\infty, -b_1/a_1[$.

We compute the intervals on λ where the inequality is not satisfied. The intersection of these intervals with $[0,1]$ gives the portion of the trajectory where the constraints on the joint are not satisfied. This computation is done for all the faces of all the 12 pyramids defining the constraint on the joints, and I_{pyr_i} denotes the union of all these intervals for link i.

2.3 Link Interference

We define the distance between two links i, j as the minimal distance between any pair of points on the links. It has been shown in Merlet [**?**] that this distance is the minimum of the following distances:

- The distance between the lines associated with the links if their common perpendicular has a point on each link
- The distance between a point B_i and its projection B_i^j on link j if B_i^j belongs to link j
- The distance between a point A_i and its projection A_i^j on link j if A_i^j belongs to link j
- The distance between the points of one of the two pairs of points (A_i, B_j)

We assume that link i can be approximated by a cylinder with radius r_i and links i, j will interfere if their distance is less than $d = r_i + r_j$. Without loss of generality we will consider the interference between links 1 and 2.

2.3.1 Distance Between the Lines

The distance l_{12} between the lines associated with links 1 and 2 can be written as:

$$l_{12} = \frac{\mathbf{A_1A_2}.(\mathbf{A_1B_1} \times \mathbf{A_2B_2})^{\mathbf{T}}}{||\mathbf{A_1B_1} \times \mathbf{A_2B_2}||}. \tag{9}$$

Because

$$\mathbf{A_iB_i} = \mathbf{A_iO} + \mathbf{OM_1} + \lambda\mathbf{M_1M_2} + \mathbf{CB_i}, \tag{10}$$

the inequality $l_{12} \leq d$ leads to a second-order inequality $P_1(\lambda) \geq 0$. The intervals I_d on λ included in [0,1] such that $P_1(\lambda)$ is positive define the parts of the trajectory for which the distance between the lines is less than or equal to d.

Let Q_1, Q_2 be the points on lines 1,2 belonging to their common perpendicular. If these points belong to the links for some values of λ in I_d, then there is link interference. We define α_1, α_2 by:

$$\mathbf{A_1Q_1} = \alpha_1\mathbf{A_1B_1} \qquad \mathbf{A_2Q_2} = \alpha_2\mathbf{A_2B_2}. \tag{11}$$

Consequently, point Q_i belongs to link i if α_i is in [0,1]. Using equation (10), α_1, α_2 can easily be obtained as:

$$\alpha_1 = \frac{P_{\alpha_1}(\lambda)}{P_d(\lambda)} = \frac{s_2\lambda^2 + s_1\lambda + s_0}{{}_2\lambda^2 + t_1\lambda + t_0} \qquad \alpha_2 = \frac{P_{\alpha_2}(\lambda)}{P_d(\lambda)} = \frac{r_2\lambda^2 + r_1\lambda + r_0}{t_2\lambda^2 + t_1\lambda + t_0}, \tag{12}$$

where the r, s, t are constants. Let $I_+^{P_i}$ be the intervals included in [0,1] such that P_{α_i} is positive or equal to 0 (i.e., $\alpha_i \geq 0$) and $I_1^{P_i}$ be the intervals in [0,1] where $P_{\alpha_i} - P_d(\lambda)$ is negative or equal to 0 (i.e., $\alpha_i \leq 1$). We may remark that all these intervals can be easily obtained from the above equations. The set I_D of intervals of λ in [0,1] where the distance between the links is the distance between the lines and is less than d is therefore:

$$I_D = I_d \cap (I_+^{P_1} \cap I_1^{P_1}) \cap (I_+^{P_2} \cap I_1^{P_2}). \tag{13}$$

If I_d is an empty set, the distance between the lines (which is a lower bound of the distance between the links) is always greater than d, and therefore link interference cannot occur. If I_d is not empty and I_D is empty, we cannot state whether the distance between the links is less than d, as this distance is always greater or equal to the distance between the lines. The distance between the links is therefore different from the distance between the lines.

2.3.2 Distance Between the Points B_i and Their Projections

The distance l from point B_1 to line 2 can be written as:

$$l = \frac{||\mathbf{B_1B_2} \times \mathbf{A_2B_2}||}{||\mathbf{A_2B_2}||}. \tag{14}$$

The inequality $l \leq d$ leads to a second-order inequality $P_1^{B_1^2}(\lambda) \geq 0$, and interference will occur if the projection Q_1 of B_1 on line 2 belongs to link 2. We define β_1 such that $\mathbf{A_2Q_1} = \beta_1\mathbf{A_2B_2}$, and the above condition will be fulfilled if β_1 belongs to [0,1]. Equations (2) and (1) lead to:

$$\beta_1 = \frac{\mathbf{A_2B_1}.\mathbf{A_2B_2^T}}{||\mathbf{A_2B_2}||^2} = \frac{P_2^{B_1^2}(\lambda)}{Q(\lambda)} = \frac{b_2^1\lambda^2 + b_1^1\lambda + b_0^1}{f_2\lambda^2 + f_1\lambda + f_0}. \tag{15}$$

Let $I_{B_i^j}$ be the intervals included in [0,1] such that $P_1^{B_i^j} \geq 0$ (i.e., $l \leq d$), $P_2^{B_i^j} \geq 0$ (i.e., $\beta_1 \geq 0$), $P_2^{B_i^j} - Q(\lambda) \leq 0$ (i.e., $\beta_1 \leq 1$). The set of intervals $I_{B_i^j},\ i,j \in [1,6], i \neq j$ defines the components of the trajectory for which interference occurs between links i and j.

2.3.3 Distance Between the Points A_i and Their Projections

The distance $l_{A_1^2}$ from point A_1 to line 2 is:

$$l_{A_1^2} = \frac{||\mathbf{A_1B_2} \times \mathbf{A_2B_2}||}{||\mathbf{A_2B_2}||}. \tag{16}$$

The components of the trajectory for which link interference occurs are defined by the intervals such that $l_{A_1^2} - d \leq 0$, which is equivalent to a second-order inequality $P_1^{A_1^2}(\lambda) \geq 0$ under the condition that the projection Q_1 of A_1 on line 2 belongs to link 2. We define μ_1 such that $\mathbf{A_2Q_1} = \mu_1\mathbf{A_2B_2}$ and Q_1 belongs to link 2 if μ_1 is in [0,1]. Equations (2) and (1) lead to:

$$\mu_1 = \frac{\mathbf{A_2A_1}.\mathbf{A_2B_2^T}}{||\mathbf{A_2B_2}||^2} = \frac{P_2^{A_1^2}(\lambda)}{Q(\lambda)} = \frac{b_1^1\lambda + b_0^1}{f_2\lambda^2 + f_1\lambda + f_0}, \tag{17}$$

where the a, b, f are constants. Let $I_{A_i^j}$ denote the intervals included in [0,1] such that $P_1^{A_i^j} > 0$ ($l_{A12} \leq d$), $P_2^{A_i^j} \geq 0$ ($\mu_1 \geq 0$), $P_2^{A_i^j} - Q(\lambda) \leq 0$ ($\mu_1 \leq 1$). The set of intervals $I_{A_i^j},\ i,j \in [1,6],\ i \neq j$ define the components of the trajectory on which interference between links i and j occurs.

2.3.4 Distance Between Points A_i and B_j

The distance between points A_2 and B_1 can be written as:

$$\begin{aligned} ||\mathbf{A_2B_1}||^2 &= P_{A_2B_1}(\lambda) \\ &= \lambda^2||\mathbf{M_1M_2}||^2 + 2\lambda(\mathbf{A_2M_1} + \mathbf{CB_1}).\mathbf{M_1M_2^T} + ||(\mathbf{A_2M_1} + \mathbf{CB_1})||^2, \end{aligned} \tag{18}$$

which is a second-order polynomial in λ. We denote by $I_{A_iB_j}$ the intervals of λ included in [0,1] such that $P_{A_iB_j}(\lambda) - d^2 \leq 0$. These intervals define the parts of the trajectory for which the distance from B_j to A_i is less than d. An analysis of this inequality [**?**] enables us to establish the following rule:

Rule 7: *Let α be the angle between the vectors $\mathbf{A_iM_1} + \mathbf{CB_j}$, $\mathbf{M_1M_2}$. If the distance between the points A_i and B_j is greater than d when the end-effector location is M_1 and M_2, then the distance between these points will be less than d for some C on the line joining M_1 and M_2 if and only if:*

$$(\mathbf{A_iM_1} + \mathbf{CB_j}).\mathbf{M_1M_2^T} < \mathbf{0} \quad ||\mathbf{M_1M_2}||^2 + (\mathbf{A_iM_1} + \mathbf{CB_j}).\mathbf{M_1M_2^T} > \mathbf{0} \quad \mathbf{d^2} > ||\mathbf{A_iM_1} + \mathbf{CB_j}||^2 \sin^2\alpha \tag{19}$$

The union I_{bad} of all the forbidden intervals for λ for each constraint describes the parts of the trajectory that are outside the workspace. We get:

$$I_{bad} = I_{max} \cup I_{min} \cup I_{pyr_i} \cup I_{D_{ij}} \cup I_{B_i^j} \cup I_{A_i^j} \cup I_{A_i B_j}. \tag{20}$$

2.4 Computation Time

The above algorithms have been implemented in a workspace computation program. This program is written in C on a Sun Sparc2 workstation.

The computation time for verifying the the link lengths constraints is approximatively 1.6 ms if the trajectory is correct and 2.1 ms if some points are outside the workspace. A computation time of 1.34 to 1.72 ms is necessary for checking link interference between a pair of links. As for the mechanical limits on the passive joints, the computation time for one face of one pyramid is approximatively 0.3 ms.

If we check all the constraints, the computation time for a trajectory is approximatively 29 ms. Such a time seems to be adequate with a real-time computation.

2.5 Examples

We have performed trajectory verification for a prototype of a parallel manipulator developed by Arai et al. [**?**] at the Mechanical Engineering Laboratory in Tsukuba (Fig. 4 and 5).

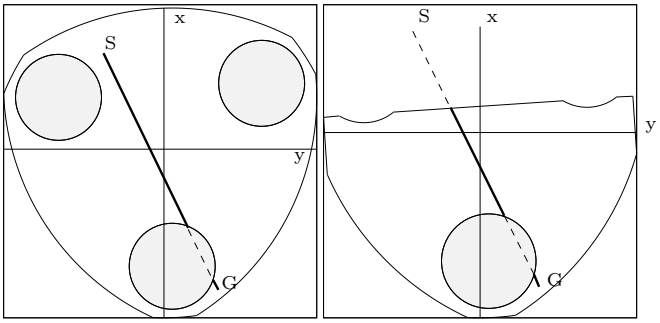

Figure 4: The forbidden parts of the trajectory are drawn in dashed lines. The gray zones are forbidden zones of the workspace. On the left the constraints are only the link lengths, and on the right there is a constraint on one of the base joints (computation time: 1.99 ms and 3.166 ms).

3 Trajectory With a Varying Orientation

In the case of a constant orientation we have seen that the constraints can be expressed under the form of algebraic equations in the variable λ. If we now introduce a varying orientation, we have no more algebraic constraints, as λ will appear in the sines and cosines of the rotation matrix.

To get algebraic constraints equations, we split the trajectory in elementary parts such that the change in the orientation will be small. As the orientation will affect only the vector **CB**, we will use a first or second-order approximation for this vector. Let M_1, M_2 denote the extremities of one elementary part of the trajectory; ψ_1, θ_1, ϕ_1 the angles describing the orientation of the end-effector at point M_1; and ψ_2, θ_2, ϕ_2 the angles of the end-effector at point M_2. Between points M_1 and M_2 (Fig. 6) the position of point C is defined by equation (1), and the orientation angles can be written as:

$$\psi = \psi_1 + \lambda(\psi_2 - \psi_1) \qquad \theta = \theta_1 + \lambda(\theta_2 - \theta_1) \qquad \phi = \phi_1 + \lambda(\phi_2 - \phi_1) \tag{21}$$

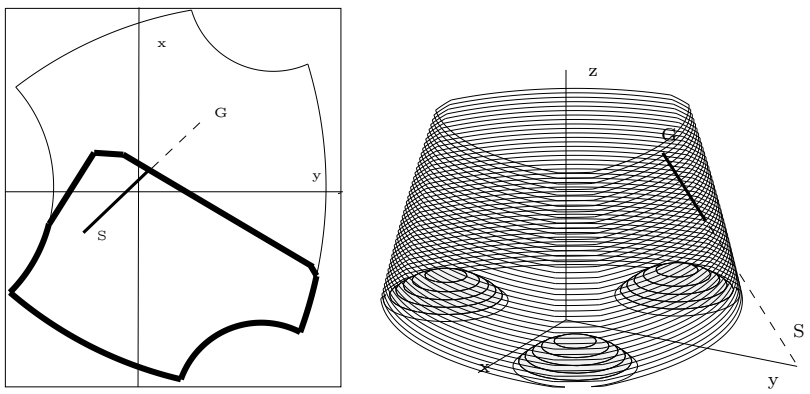

Figure 5: Examples of trajectory verification. On the left, the frontier of the workspace where there is no link interference is drawn in thick lines; on the right is drawn a 3D trajectory. The forbidden parts of the trajectory are drawn in dashed lines (computation time: 13.5 ms and 200 ms).

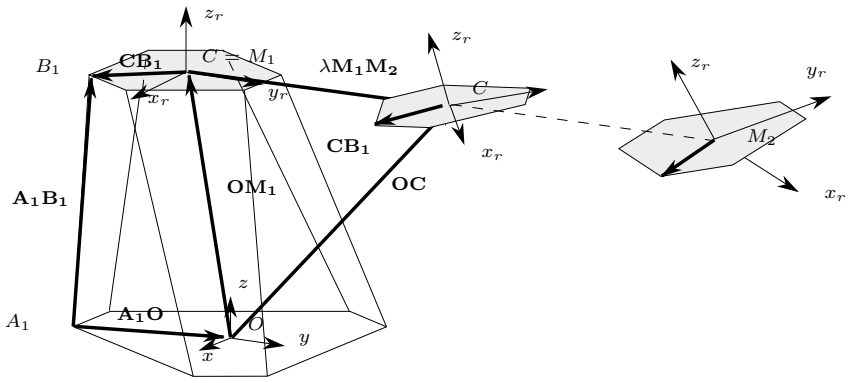

Figure 6: The various vectors used in the equations.

Using a first or second-order approximation of **CB** leads to:

$$\mathbf{CB}(\psi,\theta,\phi) = \mathbf{CB}(\psi_\mathbf{1},\theta_\mathbf{1},\phi_\mathbf{1}) + \lambda\mathbf{U_1} \ , \quad (22)$$

$$\mathbf{CB}(\psi,\theta,\phi) = \mathbf{CB}(\psi_\mathbf{1},\theta_\mathbf{1},\phi_\mathbf{1}) + \lambda\mathbf{U_1} + \lambda^\mathbf{2}\mathbf{U_2} \ , \quad (23)$$

where the vectors $\mathbf{U_1}, \mathbf{U_2}$ are only dependent on the relative position of B and the angles ψ_1,θ_1,ϕ_1 and ψ_2,θ_2,ϕ_2. Under this assumption we may now analyze the various constraints on an elementary part $\mathcal{T}$ of the trajectory.

3.1 Link Lengths Constraints

By using equation (1) and a second-order approximation (23), we obtain the square of the link length ρ^2 as a third-order polynomial $P_\rho(\lambda)$. As for the constant orientation case, the analysis of the polynomial $P_\rho(\lambda) - \rho^2_{max}, P_\rho(\lambda) - \rho^2_{min}$ enables us to compute the intervals of λ in [0,1] such that the link length is greater than its maximum value or lower than its minimal value.

3.2 Constraints on the Passive Joints

Using a second-order approximation of **CB** (23) together with equation (1), the constraint equation (7) leads to a second-order inequality. Analysis of this inequality yields the intervals on λ such that some point of the link lies outside the pyramid. By considering all the set of faces of every pyramid, we get those parts of the trajectory that do not satisfy the joints constraints. A similar analysis can be done for the passive joints of the mobile plate. The following simplification rules can be established [**?**]:

Rule 8: *Let* $\mathbf{n_i}$ *be the external normal to the face i of the pyramid describing the constraints on the base joint. Let* $\mathbf{U_1}, \mathbf{U_2}$ *be the vectors of the second-order approximation of* **CB**. *If, at the extreme points of* $\mathcal{T}$, *the vector* **AB** *lies inside the pyramid with respect to face i and if* $\mathbf{U_2}.\mathbf{n_i^T} > \mathbf{0}$, *then the constraint on the joint is satisfied on the whole* $\mathcal{T}$.

Rule 9: *If, at the extreme points of* $\mathcal{T}$, *the vector* **AB** *lies inside the pyramid with respect to face i, this constraint will not be satisfied at some points of* $\mathcal{T}$ *if and only if:*

$$\mathbf{U_2}.\mathbf{n_i^T} < \mathbf{0} \quad \mathbf{M_1M_2}.\mathbf{n_i^T} + \mathbf{U_1}.\mathbf{n_i^T} > \mathbf{0} \quad \mathbf{2U_2}.\mathbf{n_i^T} + \mathbf{M_1M_2}.\mathbf{n_i^T} + \mathbf{U_1}.\mathbf{n_i^T} < \mathbf{0} \quad (24)$$

$$(\mathbf{M_1M_2}.\mathbf{n_i^T} + \mathbf{U_1}.\mathbf{n_i^T})^\mathbf{2} > \mathbf{4U_2}.\mathbf{n_i^T}(\mathbf{AM_1}.\mathbf{n_i^T} + \mathbf{CB}(\psi_\mathbf{1},\theta_\mathbf{1},\phi_\mathbf{1}).\mathbf{n_i^T}) \quad (25)$$

3.3 Link Interference

3.3.1 Distance Between the Lines

Let l_{12} denote the distance between lines 1 and 2. By using equation (1) and a first-order approximation (22), the inequality $l_{12} \leq d$ can be written as a fourth-order polynomial in λ $P(\lambda) \geq 0$.

If the common perpendicular points Q_1, Q_2 of lines 1 and 2 belong to the links, we get link interference. We define α_1, α_2 such that $\mathbf{A_1Q_1} = \alpha_\mathbf{1}\mathbf{A_1B_1}, \mathbf{A_2Q_2} = \alpha_\mathbf{2}\mathbf{A_2B_2}$ and we get:

$$\alpha_1 = \frac{P_{\alpha_1}(\lambda)}{P_d(\lambda)} = \frac{s_3\lambda^3 + s_2\lambda^2 + s_1\lambda + s_0}{t_4\lambda^4 + t_3\lambda^3 + t_2\lambda^2 + t_1\lambda + t_0} \qquad \alpha_2 = \frac{P_{\alpha_2}(\lambda)}{P_d(\lambda)} = \frac{r_3\lambda^3 + r_2\lambda^2 + r_1\lambda + r_0}{t_4\lambda^4 + t_3\lambda^3 + t_2\lambda^2 + t_1\lambda + t_0}, \quad (26)$$

where r, s, t are constants. We compute the intervals of [0,1] where $P(\lambda) \geq 0$ (i.e., $l_{12} \leq d$), $P(\alpha_i) \geq 0$ (i.e., $\alpha_i \geq 0$), $P(\alpha_i) - P_d \leq 0$ (i.e., $\alpha_i \leq 1$). All these intervals can be easily derived from the analysis of the various polynomials. The intersection I_D of all these intervals defines the components of $\mathcal{T}$ for which link interference occurs. If I_d is empty, the distance between the lines (which is a lower bound of the distance between the links) is always greater than d. Consequently, the distance between the links is also always greater than d. If I_d is not empty and I_D is empty, we cannot conclude as the distance between the lines is less than d but the distance between the links is greater than the distance between the lines.

3.3.2 Distance Between the Points B_i and Their Projections

The distance l from point B_1 to line 2 is given by equation (14). Using equations (1) and (23), the inequality $l \leq d$ leads to a fourth-order inequality $P_1^{B_1^2}(\lambda) \geq 0$. Link interference will occur if this inequality is satisfied and if the projected point Q_1 of B_1 on line 2 belongs to link 2. We get:

$$\mathbf{A_2Q_1} = \beta_1 \mathbf{A_2B_2} = \frac{\mathbf{P_2^{B_1^2}}(\lambda)}{\mathbf{Q}(\lambda)} \mathbf{A_2B_2} = \frac{\mathbf{b_2^1}\lambda^2 + \mathbf{b_1^1}\lambda + \mathbf{b_0^1}}{\mathbf{f_2}\lambda^2 + \mathbf{f_1}\lambda + \mathbf{f_0}} \mathbf{A_2B_2}. \tag{27}$$

Q_1 will belong to link 2 if β_1 is in [0,1]. Let $I_{B_i^j}$ be the intervals included in [0,1] such that $P_1^{B_i^j} \geq 0$ ($l \leq d$), $P_2^{B_i^j} \geq 0$ ($\beta_1 \geq 0$), $P_2^{B_i^j} - Q(\lambda) \leq 0$ ($\beta_1 \leq 1$). The set of intervals $I_{B_i^j},\ i,j \in [1,6], i \neq j$ defines the components of $\mathcal{T}$ where there is interference between links i, j.

3.3.3 Distance Between the Points A_i and Their Projections

The distance l between the point A_1 and line 2 is given by equation (16). Using equations (1) and (22), the inequality $l \leq d$ leads to a second-order inequality $P_1^{A_1^2}(\lambda) \geq 0$.

Collision between links 1 and 2 will occur if the projection point Q_1 of A_1 on line 2 belongs to link 2. Let:

$$\mathbf{A_2Q_1} = \mu_1 \mathbf{A_2B_2} = \frac{\mathbf{P_2^{A_1^2}}(\lambda)}{\mathbf{Q}(\lambda)} \mathbf{A_2B_2} = \frac{\mathbf{b_1^1}\lambda + \mathbf{b_0^1}}{\mathbf{f_2}\lambda^2 + \mathbf{f_1}\lambda + \mathbf{f_0}} \mathbf{A_2B_2}. \tag{28}$$

The above condition will be fulfilled if μ_1 is in [0,1]. Let $I_{A_i^j}$ be the intervals in [0,1] such that $P_1^{A_i^j} \geq 0$, $P_2^{A_i^j} \geq 0$ (i.e., $\mu_1 \geq 0$), $P_2^{A_i^j} - Q(\lambda) \leq 0$ (i.e., $\mu_1 \leq 1$). The set of intervals $I_{A_i^j},\ i,j \in [1,6], i \neq j$ defines the components of $\mathcal{T}$ where collision between links i, j occurs.

3.3.4 Distance Between the Points A_i and B_j

Using a first-order approximation (22), of $\mathbf{CB_1}$ we get that $||\mathbf{A_2B_1}||^\mathbf{2}$ is a second-order polynomial in λ, $P_{A_2B_1}(\lambda)$. We can propose a sufficient and necessary condition for the distance between the points A_i and B_j being less than d.

Rule 10: *Let α be the angle between the vectors $\mathbf{V_1} = \mathbf{M_1M_2} + \mathbf{U}$, $\mathbf{V_2} = \mathbf{A_iM_1} + \mathbf{CB_j}(\psi_1, \theta_1, \phi_1)$. If, at the extreme points of $\mathcal{T}$, the distance between the points A_i, B_j is greater than d, then at some point of $\mathcal{T}$ the distance between these points will be less than d if and only if:*

$$(\mathbf{A_iM_1} + \mathbf{CB_j}(\psi_1, \theta_1, \phi_1)).(\mathbf{M_1M_2} + \mathbf{U})^\mathbf{T} < \mathbf{0} \qquad \mathbf{d^2} > ||\mathbf{A_iM_1} + \mathbf{CB_j}(\psi_1, \theta_1, \phi_1)||^\mathbf{2} \sin^\mathbf{2}\alpha \tag{29}$$

$$||\mathbf{M_1M_2} + \mathbf{U}||^\mathbf{2} + (\mathbf{CB_j}(\psi_1, \theta_1, \phi_1) + \mathbf{A_jM_1}).\mathbf{M_1M_2^T} > \mathbf{0} \tag{30}$$

3.4 Computation Time

The computation time for the verification of a trajectory is dependent on the number of elementary parts. This number is obtained by considering the orientation angles with the greatest variation and by dividing this variation by a constant angle (5 degrees in our implementation).

If we consider only the link length constraints, the mean computation time for the verification of one elementary part is approximatively 16 ms. The computation time for checking interference between a pair of links is about 27.6 ms. As for the mechanical limits on the passive joints, the computation time for checking one face of a pyramid is 1 ms. If we take into account all the constraints for a robot with four-faced pyramids on the base joints, we get a total computation time of 450 ms.

In conclusion, the full verification of a trajectory cannot be realized in real-time, but our method remains efficient and safe. Fig. 7 shows some examples of trajectory verification. The computation time are 423 ms and 4133 ms, respectively.

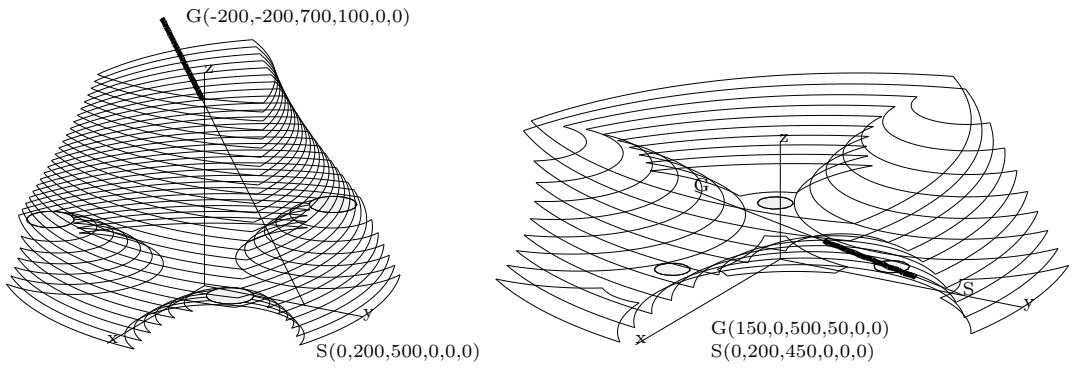

Figure 7: Examples of trajectory verification. The forbidden parts of the trajectory are drawn in thick lines. Each cross-section of the workspace has been computed for an orientation obtained by a linear interpolation between the orientations at the start and goal positions. On the left the constraints are the link lengths and the goal point is outside the workspace. On the right the constraints are the link lengths and link interference. The start and goal points are in the workspace, but a part of the trajectory is outside the workspace.

4 Application: Motion Planning

Let us assume that a given trajectory is outside the workspace of a given robot. We will assume here that the orientation is kept constant all along the trajectory and that the trajectory is in a given horizontal plane. Because we know the workspace border, we tile it with small square cells (Fig. 8).

In this tessellation we find the squares that contain the start and goal points. We then build a valued graph whose nodes are the centers of the cells, which are connected by arcs to their neighbor cells. The value of the arcs is the distance between the nodes if the line joining the nodes lies inside the workspace or an arbitrary large value if the line is outside the workspace. This can be determined by using our verification algorithm. A path between the start and goal points can be found by using a shortest path algorithm in the graph (for example an A^* algorithm [**?**]), and this path is then smoothed. Because the tessellation is computer intensive, the computation time is rather high (4.51 s in the presented example) (Fig. 8).

5 Conclusion

We have presented an algorithm enabling us to verify whether a given trajectory is fully inside the workspace of a parallel manipulator. This workspace is calculated by considering mechanical constraints that can limit the reach of the robot: link length range, mechanical limits on the passive joints and link interference. In this algorithm these constraints are expressed as algebraic inequalities which are easily solved. These algebraic inequalities exactly describe the constraints if the orientation of the end-effector is kept constant all along the trajectory and are approximately exact if changes in orientation are also considered. By solving these inequalities we can determine whether the trajectory is fully inside the workspace or find which part of the trajectory is outside the workspace. This algorithm is fast enough to be used in real time for a constant orientation trajectory.

References

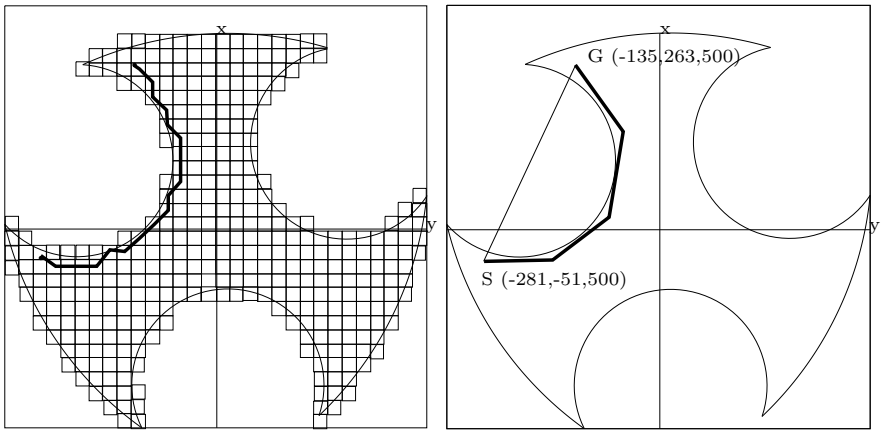

Figure 8: Motion planning: the straight line between the start and goal points is not a valid trajectory. The workspace border is calculated and is tiled with small square cells. An A^* algorithm enables us to find a path between the start and goal position in the workspace (left). A smoothing algorithm can then be used (right).

List of Figures