Trajectory verification of parallel manipulators in the workspace

Jean-Pierre MERLET
INRIA
Centre de Sophia-Antipolis
BP 93 06902 Sophia-Antipolis, France
E-mail: merlet@cygnusx1.inria.fr

Abstract

We present a fast algorithm for solving the problem of determining if the straight line between two different postures of a parallel manipulator lie fully inside its workspace. This algorithm is based on the analysis of the algebraic inequalities describing the constraints on the workspace (link lengths range, mechanical limits on the joints, interference between the links) and enables to compute which part of the trajectory is outside the workspace. This method is exact if the orientation of the end-effector is kept constant along the trajectory and approximate in the opposite case.

1 Introduction

Let us consider a 6 d.o.f. parallel manipulator as represented on figure 1. It is constituted of a fixed base plate and a mobile plate connected by 6 variable-length links. One of the extremities of each link is articulated with the base plate through an universal joint and the other extremity is articulated with the mobile plate through a ball-and-socket joint. By changing the 6 link lengths we are able to control the position and orientation of the mobile plate.

Three types of constraint play a role for determining the workspace of parallel manipulators:

- limited range for the link lengths: the linear actuators controlling the links lengths have a limited range. The minimum length of link \( i \) will be denoted \( \rho_{\text{min}}^i \) and the maximum length \( \rho_{\text{max}}^i \)
- mechanical limits on the passive joints (universal joints and ball and socket joints).
- links interference

The problem of determining the workspace of a parallel manipulator has been addressed by many authors most of them using a discretization method in the parameter’s space, some of them assuming that the orientation of the end-effector is kept constant [1], [2], [5], [6] or assuming that a point is fixed for computing the orientation workspace [10], [11], [12], [13].

A completely different approach has been proposed by Gosselin [3] which uses a purely geometrical method for determining the workspace border due to the limited range of the links lengths. This approach has been then first extended to take into account all the constraints limiting the workspace [7] and for computing the orientation workspace when a point of the end effector is fixed [8].

To the best of our knowledge nobody has addressed the problem of verifying a trajectory with respect to the workspace i.e. being 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 us introduce now some notation. We define two frames, one fixed (\( O, x, y, z \)) (reference frame) and the other one attached to the end effector (\( C, x_r, y_r, z_r \))
(relative frame). \( C \) will be used to define the position of the end-effector in the reference frame. The following symbols and variables will be used in this paper:

- \( A_i \): center of the passive joint of link \( i \) attached to the base of the robot.
- \( B_i \): center of the passive joint of link \( i \) attached to the end effector.
- \( \psi, \theta, \phi \): angles defining the orientation of the end effector.

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 end-effector on the trajectory as \( M_1 \) and \( M_2 \). Consequently any position of the end-effector on the trajectory may be defined as:

\[
OC = OM_1 + \lambda M_1 M_2 \quad \text{with} \quad \lambda \in [0,1] \quad (1)
\]

2.1 Limitation on the links lengths

Let us calculate the length \( \rho \) of a link for any point on the trajectory between \( M_1 \) and \( M_2 \). We have:

\[
AB = AM_1 + CB + \lambda M_1 M_2 \quad (2)
\]

which yield to:

\[
\rho^2 = AB \cdot AB^T = a\lambda^2 + b\lambda + c \quad (3)
\]

Now let us consider the following equation:

\[
a\lambda^2 + b\lambda + c - \rho_{\text{max}}^2 = 0 \quad (4)
\]

If this equation has no root and as \( a > 0 \) then for all \( \lambda \) the equation is positive and the link length is greater than its maximum value on all the trajectory.

Assume now that the equation has two roots \( x_1, x_2 \) sorted by higher value. As \( a > 0 \) the equation will be positive for \( \lambda \) in the interval \( -\infty, x_1, x_2, +\infty \). The intersection of these intervals with the interval \([0,1]\) will define the intervals on \( \lambda \) (i.e. the portion of the trajectory) where the link length will be greater than the maximum link length value.

By using this algorithm for the 6 links and calculating the union \( I_{\text{max}} \) of all the obtained intervals we will get the portions of the trajectory where at least one link length will be greater than its allowed maximum value.

By changing \( \rho_{\text{max}} \) by \( \rho_{\text{min}} \) in equation (4) we then get the union \( I_{\text{min}} \) of the intervals which define the portions of the trajectory where at least one link length will be lower than its allowed minimum value.

The study of the number of real roots of the previous equations according to their coefficients yield to 6 simplification rules [9]. Due to the lack of space they are not given here.

2.2 Mechanical limits on the passive joints

The mechanical limits on joints like universal joints or ball-and-socket joints can be modeled by a surface which is the border of the allowable zone for the link connected to the joint. Using a similar method as in [7] 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 point \( A \) (figure 2).

The union of \( I_{\text{max}}, I_{\text{min}} \) 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.

As for the constraint on the passive joints attached to the end-effector we may use the same model. We define a pyramid \( P_i \) with center \( B_i \) such that if the constraint on the joint at \( B \) is satisfied then point \( A_i \) will lie inside the pyramid from which we deduce another pyramid which will be called an equivalent pyramid \( P_i^* \) to \( P_i \), which center is \( A_i^* \) such that if \( A_i \) lie inside \( P_i \) then \( B_i \) lie inside \( P_i^* \). Therefore for both kind of joints a similar model can be used.

Let \( n_i \) be the external normal of the \( i \)th face of the pyramid associated to the joint attached to the base. If point \( B \) lie inside the pyramid we have:

\[
AB \cdot n_i^T \leq 0 \quad (5)
\]
By using equation (1) we get a linear equation in \( \lambda \). Let us consider the possible intervals where the inequality is not satisfied. The intersection of these intervals with \([0,1]\) yield the portion of the trajectory where the constraints on the joint are not satisfied. This algorithm has to be used for all the faces of all the 12 pyramids defining the constraint on the joints. Let \( I_{p_1} \) be the union of all these intervals for link \( i \).

### 2.3 Links interference

We define the distance between two links as the minimal distance between any pair of points on the links. It has been shown in [7] that this distance is the minimum of the following distances:
- the distance between the lines associated to the links if their common perpendicular has a point on each link
- the distance between a point \( B \) and its projection point on the other link if this point belongs to the link
- the distance between a point \( A \) and its projection point on the other link if this point belongs to the link
- 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 will say that links \( i,j \) interfere if their distance if lower than \( d = r_i + r_j \). We will consider now the above cases for links 1 and 2.

#### 2.3.1 Distance between the lines

The distance \( d_{12} \) between the lines associated to links 1 and 2 can be written as:

\[
\begin{align*}
d_{12} &= \frac{A_1 A_2 \cdot (A_1 B_1 \times A_2 B_2)^T}{||A_1 B_1 \times A_2 B_2||} \tag{6}
\end{align*}
\]

Using equation (1) and writing that the distance between the lines is lower or equal to \( d \) yield to a second order inequality:

\[
\begin{align*}
P_1(\lambda) &= u_2 \lambda^2 + u_1 \lambda + u_0 \geq 0
\end{align*}
\]

The intervals on \( \lambda \) included in \([0,1]\) such that \( P_1(\lambda) \) is positive define the parts of the trajectory where the distance between the lines is lower or equal to \( d \).

Let \( I_d \) be the set of all these intervals. Let \( Q_1, Q_2 \) be the points on line 1,2 belonging to their common perpendicular. If these points belong to the links for some values of \( \lambda \) in \( I_d \) then there is links interference. We define \( \alpha_1, \alpha_2 \) such that:

\[
\begin{align*}
A_1 Q_1 &= \alpha_1 A_1 B_1 \\
A_2 Q_2 &= \alpha_2 A_2 B_2
\end{align*}
\]

Point \( Q_1 \) belongs to link \( i \) if \( \alpha_i \) is in \([0,1]\). \( \alpha_1, \alpha_2 \) can easily be obtained as:

\[
\begin{align*}
\alpha_1 &= \frac{P_{\alpha_1}(\lambda)}{P_{det}} = \frac{s_2 \lambda^2 + s_1 \lambda + s_0}{t_2 \lambda^2 + t_1 \lambda + t_0} \tag{9}
\end{align*}
\]

\[
\begin{align*}
\alpha_2 &= \frac{P_{\alpha_2}}{P_{det}} = \frac{r_2 \lambda^2 + r_1 \lambda + r_0}{t_2 \lambda^2 + t_1 \lambda + t_0}
\end{align*}
\]

Let \( I_{p_1}^i \) be the intervals included in \([0,1]\) such that \( P_{\alpha_i} \) is positive or equal to zero (i.e. \( \alpha_i \geq 0 \)), \( I_{p_1}^i \) the intervals in \([0,1]\) where \( P_{\alpha_i} - P_{det}(\lambda) \) is negative or equal to zero (i.e. \( \alpha_i \leq 1 \)). The set \( I_D \) of intervals of \( \lambda \) in \([0,1]\) where the distance between the links is the distance between the lines and is lower than \( d \) is therefore:

\[
\begin{align*}
I_D = I_d \cap (I_{p_1}^1 \cap I_{p_1}^2) \cap (I_{p_2}^1 \cap I_{p_2}^2)
\end{align*}
\]

If \( I_d = \emptyset \) the distance between the lines (which is a lower bound of the distance between the links) is always greater than \( d \) and therefore links interference cannot occur. If \( I_d \neq \emptyset \) and \( I_D = \emptyset \) the distance between the links will always be different and greater than 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:

\[
\begin{align*}
l_{B_1^2} &= \frac{||B_1 B_2 \times A_2 B_2||}{||A_2 B_2||}
\end{align*}
\]

Using equation (1) and writing that \( l_{B_1^2} \) is lower than \( d \) yield to:

\[
\begin{align*}
P_{B_1^2}(\lambda) = a_2^1 \lambda^2 + a_1^1 \lambda + a_0^1 \geq 0
\end{align*}
\]

and collision will occur if the projection \( Q_1 \) of \( B_1 \) on line 2 belongs to link 2. Let:

\[
\begin{align*}
A_2 Q_1 &= \beta_1 A_2 B_2
\end{align*}
\]

the above condition will be fulfilled if \( \beta_1 \) belongs to \([0,1]\). We have:

\[
\begin{align*}
\beta_1 &= \frac{P_{B_1^2}(\lambda)}{Q(\lambda)} = \frac{\beta_2^1 \lambda^2 + \beta_1^1 \lambda + \beta_0^1}{\beta_2^2 \lambda^2 + \beta_1^2 \lambda + \beta_0^2}
\end{align*}
\]

Let \( I_{B_1^2} \) the intervals included in \([0,1]\) such that \( P_{B_1^2} \geq 0 \) (i.e. \( l \leq d \)), \( P_{B_2^1} \geq 0 \) (i.e. \( \beta_1 \geq 0 \)), \( P_{B_2^1} - Q(\lambda) \leq 0 \) (i.e. \( \beta_1 \leq 1 \)). The set of intervals \( I_{B_1^2}, i,j \in [1,6], i \neq j \) 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 \( l_{A_1^1} \) from point \( A_1 \) to line 2 is:

\[
\begin{align*}
l_{A_1^1} &= \frac{||A_1 B_2 \times A_2 B_2||}{||A_2 B_2||}
\end{align*}
\]
Using equation (1) and writing that \( l_{A_2} \) is lower than \( d \) yield to:
\[
P_{A_2}^2(\lambda) = w_2^2\lambda^2 + w_1^2\lambda + w_0^2 \geq 0
\]
under the condition that the projection \( Q_1 \) of \( A_1 \) on line 2 belongs to link 2. Let:
\[
A_2Q_1 = \mu_1A_2B_2
\]
\( Q_1 \) belongs to link 2 if \( \mu_1 \) is in \([0,1]\). We have:
\[
\mu_1 = \frac{P_{A_2}^2(\lambda)}{Q(\lambda)} = \frac{\mu_2^2\lambda + \mu_1^2}{\mu_2^2\lambda^2 + \mu_1^2\lambda + \mu_0^2}
\]
Let \( I_{A_i^j} \) the intervals included in \([0,1]\) such that \( P_{A_i^j} > 0 \) (\( l_{A_i^j} \leq d \)), \( P_{A_i^j}^2 \geq 0 \) (\( \mu_1 \geq 0 \)), \( P_{A_i^j}^2 - Q(\lambda) \leq 0 \) (\( \mu_1 \leq 1 \)).

The set of intervals \( I_{A_i^j}, i, j \in [1,6], i \neq j \) defines the components of the trajectory on which interference between link \( i \) and \( j \) occurs.

### 2.3.4 Distance between points \( A_i \) and \( B_j \)

Using equation (1) the distance between points \( A_2 \) and \( B_1 \), \( ||A_2B_1||^2 \) is a second order polynomial in \( \lambda \)
\[
P_{A_2B_1}(\lambda).
\]
We denote by \( I_{A_iB_j} \) the intervals of \( \lambda \) included in \([0,1]\) such that \( P_{A_iB_j}(\lambda) - d^2 \leq 0 \). These intervals define the parts of the trajectory the distance from \( B_j \) to \( A_i \) is lower than \( d \).

The union of the intervals defining forbidden value for \( \lambda \) for each constraint defines the set \( I_{bad} \) of intervals forbidden for \( \lambda \) from which we deduce the forbidden parts of the trajectory. We get:
\[
I_{bad} = I_{max} \cup I_{min} \cup I_{pyr} \cup I_{D_{ij}} \cup I_{B_2} \cup I_{A_1} \cup I_{A_iB_j}
\]
\( 20 \)

### 2.4 Computation time

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

The computation time for the verification of a trajectory is approximatively 2ms for the links lengths constraints, 25 ms for checking the interference between each pair of links and 0.3ms for checking a face of a pyramid for the mechanical limits on the joints.

The verification of all the constraints the computation time for a trajectory is approximatively 29 ms.

### 2.5 Examples

We have performed trajectory verification for a prototype of parallel manipulator developed by Arai [1] at the MEL in Tsukuba. Figures 3,4 show trajectories on which forbidden parts are singled out.
3 General Trajectory

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 \( \lambda \) which describe the trajectory. If we introduce now a varying orientation we have now more algebraic constraints as the sines and cosines of the rotation matrix will appear.

In order to get again algebraic constraints equations we will split the trajectory in elementary parts such that the change in the orientation between the extremal points of one elementary part will be small.

For each elementary part a linear interpolation for the angles will be used to determine the orientation of the end-effector. As the orientation will affect only the value of the vector \( \mathbf{CB} \) we will use a first or second order approximation for this vector. Let denote \( M_1, M_2 \) the extremal points of one elementary part of the trajectory, \( \psi_1, \theta_1, \phi_1 \) the Euler’s angles describing the orientation of the end-effector at point \( M_1 \) and \( \psi_2, \theta_2, \phi_2 \) the Euler’s angles of the end-effector at point \( M_2 \). Between points \( M_1 \) and \( M_2 \) the position of point \( C \) is defined by equation (1) and the rotation angles can be written as:

\[
\psi = \psi_1 + \lambda(\psi_2 - \psi_1) \quad \theta = \theta_1 + \lambda(\theta_2 - \theta_1) \\
\phi = \phi_1 + \lambda(\phi_2 - \phi_1)
\]

If we use a first order approximation we get:

\[
\mathbf{CB}(\psi, \theta, \phi) = \mathbf{CB}(\psi_1, \theta_1, \phi_1) + \lambda \mathbf{U}_1 \tag{21}
\]

and a second order approximation yield to:

\[
\mathbf{CB}(\psi, \theta, \phi) = \mathbf{CB}(\psi_1, \theta_1, \phi_1) + \lambda \mathbf{U}_1 + \lambda^2 \mathbf{U}_2 \tag{22}
\]

where the vectors \( \mathbf{U}_1, \mathbf{U}_2 \) are only dependent upon the relative position of \( B \) and the angles \( \psi_1, \theta_1, \phi_1, \psi_2, \theta_2, \phi_2 \).

Under these assumptions we may now analyze the various constraints on an elementary part \( T \) of the trajectory.

3.1 Links lengths constraints

By using equation (1) and a second order approximation (22) we obtain the link length as:

\[
P_\rho(\lambda) = a_3 \lambda^3 + a_2 \lambda^2 + a_1 \lambda + a_0 \tag{23}
\]

where the \( a_i \) are coefficients which are only dependent upon the trajectory and the design of the robot. As in the constant orientation case the analysis of the polynomial \( P_\rho(\lambda) - \rho_{\text{max}}^2 \), \( P_\rho(\lambda) - \rho_{\text{min}}^2 \) enables to compute the intervals on \( \lambda \) 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

A second order approximation of \( \mathbf{CB} \) (22) is used together with equation (1) to express the constraint equation (5) which yield to a second order inequality.

The analysis of this inequality enables to determine the intervals on \( \lambda \) such that some point of the link lie outside the pyramid. By considering all the set of faces of every pyramid we get which parts of the trajectory does not satisfy the joints constraints. A similar analysis can be done for the passive joints of the mobile plate.

3.3 Links interference

3.3.1 Distance between the lines

By using equation (1) and a first order approximation (21) writing that the distance \( d_{12} \) between the lines is lower than \( d \) yield to a fourth order inequality in \( \lambda \), \( P(\lambda) \geq 0 \).

In order to find at which points of \( T \) the distance between the lines is lower or equal to \( d \) we must find the intervals \( I_4 \) on \( \lambda \), included in \([0,1]\), where \( P(\lambda) \geq 0 \).

If the common perpendicular points \( Q_1, Q_2 \) of line 1, 2 belong to the links we get a links interference. Let:

\[
\mathbf{A}_1 Q_1 = \alpha_1 \mathbf{A}_1 \mathbf{B}_1 \quad \mathbf{A}_2 Q_2 = \alpha_2 \mathbf{A}_2 \mathbf{B}_2 \tag{24}
\]

We get:

\[
\alpha_1 = \frac{P_{\alpha_1}(\lambda)}{\det} = \frac{s_3 \lambda^3 + s_2 \lambda^2 + s_1 \lambda + s_0}{t_4 \lambda^4 + t_3 \lambda^3 + t_2 \lambda^2 + t_1 \lambda + t_0} \tag{25}
\]

\[
\alpha_2 = \frac{P_{\alpha_2}(\lambda)}{\det} = \frac{r_3 \lambda^3 + r_2 \lambda^2 + r_1 \lambda + r_0}{t_4 \lambda^4 + t_3 \lambda^3 + t_2 \lambda^2 + t_1 \lambda + t_0} \tag{26}
\]

Now a procedure similar to the one described in section 2.3.1 can be used to compute the components of \( T \) where links interference will occur.

3.3.2 Distance between the points \( B_i \) and their projections

The distance \( l_{B_2} \) from point \( B_1 \) to line 2 will be lower than \( d \) if:

\[
P_1 B_2(\lambda) = g_4 \lambda^4 + g_3 \lambda^3 + g_1 \lambda^2 + g_0 \geq 0 \tag{27}
\]

Links interference will occur if the above equation is satisfied and if the projected point \( Q_1 \) of \( B_1 \) on line 2 belongs to link 2. Let:

\[
\mathbf{A}_2 Q_1 = \beta_1 \mathbf{A}_2 \mathbf{B}_2 = \frac{\beta_3 \lambda^2 + \beta_1 \lambda + \beta_0}{\beta_2 \lambda^2 + \beta_1 \lambda + \beta_0} \mathbf{A}_2 \mathbf{B}_2 \tag{28}
\]

Let: \( Q_1 \) will belong to link 2 if \( \beta_1 \) is in \([0,1]\). Now a procedure similar to the one described in section 2.3.2 can be used to compute the components of \( T \) where links interference will occur.
3.3.3 Distance between the points \( A_i \) and their projections

Using a first order approximation (21) the distance \( l_{A_i^2} \) from point \( A_1 \) to line 2 will be lower than \( d \) if:

\[
P_{A_1^2}(\lambda) = h_2^2\lambda^2 + h_1^1\lambda + h_0^1 \geq 0 \tag{29}\]

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:

\[
A_2Q_1 = \mu_1A_2B_2 = \frac{\mu_2^\lambda + \mu_0^3}{\mu_2^\lambda + \mu_1^\lambda + \mu_0}A_2B_2 \tag{30}\]

The above condition will be fulfilled if \( \mu_1 \) is in \([0,1]\).

Now a procedure similar to the one described in section 2.3.3 can be used to compute the components of \( T \) where links interference will occur.

3.3.4 Distance between the points \( A_i \) and \( B_j \)

Using a first order approximation (21) \( ||A_2B_1||^2 \leq d \) yield to a second order inequality in \( \lambda \). This situation may occur only if a condition is satisfied (see [9]).

3.4 Computation time

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

The mean computation time for the verification of one elementary part is approximatively 16 ms for the links lengths constraints, 430ms for checking the interference between each pair of links and 1ms for checking a face of a pyramid for the mechanical limits on the joint. 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.

3.5 Examples of trajectory verification

We show in figure 5 an example of trajectory verification.

![Figure 5: Example of trajectory verification](image)

Each of the 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. The constraints are the links lengths and links interference. The start and goal points are in the workspace but a part of the trajectory is outside the workspace.

The value of the arcs is the distance between the nodes if the line joining the nodes lie 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 [4]) and this path may then be smoothed (figure 6).

![Figure 6: Motion planning](image)
5 Conclusion

We have presented an algorithm enabling to verify if a given trajectory is fully inside the workspace of a parallel manipulator. This workspace is calculated by considering every constraints which can limit the reach of the robot: links lengths range, mechanical limits on the passive joints, links interference. In this algorithm these constraints are expressed as algebraic inequalities which are easily solved. These algebraic inequalities describe exactly the constraints if the orientation of the end-effector is kept constant all along the trajectory and are approximatively exact in the opposite case. By solving these inequalities we can determine if the trajectory is fully inside the workspace or find which part of the trajectory is outside the workspace.

References