Proceedings of the 2006 IEEE International Conference on Robotics and Automation
Orlando, Florida - May 2006

Legs Interference Checking of Parallel Robots over a Given Workspace or Trajectory

J-P. Merlet
INRIA
BP 93, 06902 Sophia-Antipolis Cedex, France
Jean-Pierre.Merlet@sophia.inria.fr

D. Daney
INRIA
BP 93, 06902 Sophia-Antipolis Cedex, France
David.Daney@sophia.inria.fr

***Abstract*— We are considering a 6 d.o.f. Gough platform that has to move within a given workspace or on a trajectory. The legs of the robot are assumed to be constituted of a set of finite cylindrical elements and we address the problem of determining if any pair of such element will intersect during the robot motion. Collision conditions may be written mathematically according to various equivalent formulations. We show however that these formulations are not numerically equivalent and exhibit an efficient interference checking algorithm based on interval analysis that allows to check 6D workspace or arbitrary time-function trajectories for interference.**

I. INTRODUCTION

One of the drawbacks of parallel robots is their usually limited workspace. The constraints that enforce the limitation on the workspace are:

1) *joints limits* (either for actuated or passive joints): for example the leg lengths of a Gough platform (figure 1) are restricted to lie within some ranges while the motion of the U, S joints that are placed at the end of the legs may also be limited
2) *singularity*: usually parallel robots cannot cross a singularity and consequently singular varieties may split the workspace into different components. The robot motion will be restricted to lie within the components corresponding to the initial assembly mode of the robot
3) *self-collision*: collision between the legs of the robot and eventually with the platform or base may also limit the workspace

Limitations related to point 1 have been extensively studied and efficient algorithms for computing the corresponding workspace are available [3], [6]. Limitations due to point 2 are still an open problem although algorithms are available to check if a given workspace is singularity-free or to find the largest singularity-free cube, sphere or cylinder included in the workspace [7]. Interference between the legs have been considered for planar robot [1], [8], for wire robots [5] (which is a simpler problem as the legs can be assimilated to line segments) or for specific small trajectories within a global motion planner [2]. But to the best of our knowledge for spatial robots workspace determination taking interference into account has been proposed only for a constant orientation of the platform [4]. Still experiences on the Gough platform shows that interference between legs of complex shape is a highly limiting factor for the workspace, especially because of interference between the elements close to the base. As there was up to now no known algorithm to check this interference many designers of prototypes and industrial robots uses a limitation on the stroke of the actuator as a safety measure, thereby limiting the workspace of their robots as this limitation may go up to 60 % of the available stroke of the actuator. Hence interference checking may play an important role to improve the size of the workspace.

In this paper we will consider a Gough platform (figure 1) but any other type of parallel robot may be considered as well. The fixed frame $(O, \mathbf{x}, \mathbf{y}, \mathbf{z})$ will be called the *base frame* while a mobile frame $(C, \mathbf{x_r}, \mathbf{y_r}, \mathbf{z_r})$ attached to the platform will be called the *platform frame*. The pose of the platform will be parametrized by the location of C in the base frame and 3 angles will be used to define the orientation of the mobile frame with respect to the base frame. We will assume that the robot is constituted of finite cylinders with circular section (a common shape for sensors and actuators). The circular sections on top and bottom of the cylinder will be called the *top* and *bottom* sections. Different types of cylinder will be considered, figure 1:

- *base cylinder (BC)*: a cylinder connected to the base with constant length, radius and a fixed axis
- *platform cylinder (PC)*: a cylinder connected to the platform whose axis in the platform frame is known and whose length and radius are known
- *base mobile cylinder (BMC)*: a cylinder of constant radius with the center of its bottom section having a fixed location in the base frame, but with varying axis direction and length, which can be determined being given the pose of the platform
- *platform mobile cylinder (PMC)*: a cylinder of constant radius with the center of its top section having a fixed location in the platform frame, but with varying axis direction and length, which can be determined being given the pose of the platform.

The purpose of this paper is to describe an algorithm that check if any pair of cylindrical elements of the robot may intersect for a prescribed motion of the platform. This prescribed motion may be either

- a trajectory defined by arbitrary time-functions for the pose parameters

0-7803-9505-0/06/$20.00 ©2006 IEEE

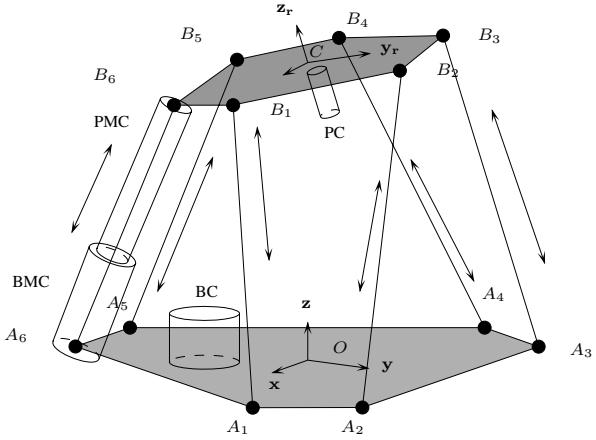

Fig. 1. A Gough platform and its various cylinder elements

- a workspace defined by ranges for the coordinates of C (but we will see that other workspace shape may be defined as well) and ranges for the 3 orientation angles

II. Two cylinders intersection conditions

We consider 2 finite cylinders $\mathcal{C}_1, \mathcal{C}_2$ which are fully defined by the coordinates of the centers of their top and bottom sections (P_1, Q_1), (P_2, Q_2) together with their radii r_1, r_2.

A. First approach

A first approach to determine if $\mathcal{C}_1, \mathcal{C}_2$ intersect is to define a distance d between 2 lines segments $ls_1 = P_1Q_1$, $ls_2 = P_2Q_2$. The projection of a point F_i of a line segment ls_i on the line associated to the line segment ls_j will be denoted F_i^j. The distance d_{12} between the lines associated to ls_1, ls_2 is equal to:

$$d_{12} = \frac{|\mathbf{P_1P_2}.\mathbf{P_1Q_1} \times \mathbf{P_2Q_2}|}{||\mathbf{P_1Q_1} \times \mathbf{P_2Q_2}||} \quad (1)$$

Let M_1, M_2 be the points belonging to the lines associated to ls_1, ls_2 and to their common perpendicular and define l_1, l_2 such that $\mathbf{P_1M_1} = l_1\mathbf{P_1Q_1}$, $\mathbf{P_2M_2} = l_2\mathbf{P_2Q_2}$. We have

$$l_1 = \frac{\mathbf{P_1P_2}.(\mathbf{P_2Q_2} \times (\mathbf{P_1Q_1} \times \mathbf{P_2Q_2}))}{||\mathbf{P_1Q_1} \times \mathbf{P_2Q_2}||^2}$$
$$l_2 = \frac{\mathbf{P_1P_2}.(\mathbf{P_1Q_1} \times (\mathbf{P_1Q_1} \times \mathbf{P_2Q_2}))}{||\mathbf{P_1Q_1} \times \mathbf{P_2Q_2}||^2}$$

We define the set of points $S_1 = \{P_1, Q_1\}$ to which we add P_2^1, Q_2^1 if they belong to ls_1. Similarly we construct the set $S_2 = \{P_2, Q_2\}$ to which may be added P_1^2, Q_1^2 if they belong to ls_2. We define then d as

$$d = \begin{cases} d_{12} \text{ if } \ l_1, l_2 \in \ [0,1] \\ \text{Min}(||\mathbf{Y_1Y_2}||) \ \forall \ Y_1 \in \ S_1, Y_2 \in \ S_2 \end{cases} \quad (2)$$

The following results hold: if $d > r_1 + r_2$, then there is no collision between the cylinders, if $d \leq r_1 + r_2$ and $d = d_{12}$ then the two cylinders intersect. If $d \leq r_1 + r_2$ and $d \neq d_{12}$, then the definition of d does not allow to determine if the cylinders intersect and we have to check the intersection between the top and bottom section of the cylinders.

B. Second approach

Let $\mathbf{n_2}$ be the unit vector of the line associated to ls_2 and two mutually orthogonal vectors $\mathbf{v_2}, \mathbf{w_2}$ that are also perpendicular to $\mathbf{n_2}$. We consider a point M_2 on the axis of the cylinder $\mathcal{C}_2$ so that $\mathbf{P_2M_2} = l_2\mathbf{P_2Q_2}$. A point M on the circular section of the cylinder whose center is M_2 is such that $\mathbf{M_2M} = r_2 \cos\theta \mathbf{v_2} + r_2 \sin\theta \mathbf{w_2}$. Consequently the coordinates of M in the base frame are obtained as:

$$\mathbf{OM} = \mathbf{OP_2} + l_2\mathbf{P_2Q_2} + r_2 \cos\theta \mathbf{v_2} + r_2 \sin\theta \mathbf{w_2} \quad (3)$$

Let M_1 be a point on the axis of the cylinder $\mathcal{C}_1$ such that $\mathbf{P_1M_1} = l_1\mathbf{P_1Q_1}$. The following results hold:

- if $||\mathbf{M_1M}|| > r_1$ for all $l_1, l_2 \in \ [0,1]$ and $\theta \ \in [0, 2\pi]$, then the cylinders do not intersect
- if $||\mathbf{M_1M}|| \leq r_1$ and $\mathbf{P_1M}.\mathbf{P_1Q_1} \geq 0$, $\mathbf{Q_1M}.\mathbf{P_1Q_1} \leq 0$, then the cylinders intersect.

This approach provides a full interference check and is therefore more complete than the first approach.

C. Third approach

A point M belong to the cylinder $\mathcal{C}_1$ if

$$\frac{||\mathbf{P_1M} \times \mathbf{P_1Q_1}||}{||\mathbf{P_1Q_1}||} \leq r_1 \qquad 0 \leq \frac{\mathbf{P_1M}.\mathbf{P_1Q_1}}{||\mathbf{P_1Q_1}||} \leq 1 \quad (4)$$

The first equation indicates that the distance of M to the line associated to ls_1 is less than the cylinder radius while the inequalities implies that the projection of M on this line should belong to ls_1. A similar set of three inequalities may be established to indicate that M belong to $\mathcal{C}_2$. If there is a M for which all 6 inequalities are satisfied, then the cylinders intersect, otherwise they do not intersect. This approach, like the second one, provides a full interference check but involve 6 inequalities instead of the 3 inequalities of the second approach.

III. Cylindrical elements intersection

A. Intersection in a pose

Being given a pose of the robot it is possible to calculate the location of the center of the top and bottom sections of any cylindrical elements i.e. the points P_1, Q_1, P_2, Q_2. Using the above approaches it is then possible to determine if a pair of cylindrical elements intersect using the following procedures:

- for the first approach: compute l_1, l_2. If $l_1, l_2 \in [0,1]$ and $d_{12} \leq (>) r_1 + r_2$, then the cylindrical elements do (not) intersect. If $d > r_1 + r_2$, then the elements do not intersect. If $d \neq d_{12} < r_1 + r_2$ it is necessary to check the intersection of the top and bottom sections of the cylinders
- for the second approach: $||\mathbf{M_1M}||^2$ is a function of l_1, l_2, θ. To respect the constraints on l_1, l_2 we define new variables a_1, a_2 such that $l_i = (1 -$

$\sin(a_i))/2$. To get the extrema of $||\mathbf{M_1M}||^2$ we calculate its derivative with respect to a_1, a_2, θ, which are trigonometric functions of these variables. Using the Weierstrass substitution we transform these 3 equations into 3 polynomial equations in the variables $T_1 = \tan(a_1/2)$, $T_2 = \tan(a_2/2)$, $T = \tan(\theta/2)$. Using the resultant between each pair of equation we first eliminate T_1. It remains then 2 equations in T_2, T whose resultant in T_2 is an univariate 4th order polynomial in T. Solving this polynomial and back-substituting for T_1, T_2 allows one to get the minimum of $||\mathbf{M_1M}||^2$ and the corresponding coordinates for M. If the minimum is lower than r_1 we may then compute $\mathbf{P_1M}.\mathbf{P_1Q_1}$ and $\mathbf{Q_1M}.\mathbf{P_1Q_1}$ to verify if the cylinders intersect.

- for the third approach: we define b_1, b_2 such that $\mathbf{P_iM}.\mathbf{P_iQ_i} = (1 - \sin(b_i))||\mathbf{P_iQ_i}||/2$. If (X, Y, Z) are the coordinates of M, then this two equations are linear in X, Y. After solving this system the distance of M to the line 2 is now a function of X, b_1, b_2. We look at the minimum of this distance by solving the system of derivatives equations. These derivatives are linear in Z. After using one of these equations to determine Z we get a system of 2 equations in b_1, b_2. The first equation may be written as the product of $\cos(b_1)$ by a term which is linear in $\sin(b_1), \sin(b_2)$. The second equation is the product of $\cos(b_2)$ by the same term in $\sin(b_1), \sin(b_2)$ than for the first equation. The cancellation of this term corresponds to the case where M lies on line 2. We may thus affect a value to b_1 (or b_2). The distance from to the line 1 is then a second order polynomial in $\sin(b_2)$. Solving this equation for r_1 allows to determine if the distance between M and line 1 may be lower than r_1.

B. Numerical complexity and sensitivity for a varying pose

For a given set of poses (defined as a workspace or a trajectory) it will usually not be possible to verify analytically the intersection constraints, whatever is the chosen approach. A numerical procedure must thus be used and it is necessary to check the complexity and the numerical conditioning of each intersection approach.

In term of complexity for the intersection test of two cylindrical elements we have

- for the first approach: the unknowns are the 6 poses parameters while up to 19 inequalities must be checked.
- for the second approach: the total number of unknowns is 9 (6 poses parameters, l_1, l_2, θ) with at most 3 inequalities to verify
- for the third approach there is also 9 unknowns (6 pose parameters and the coordinates of M) and a total of 6 inequalities.

In term of numerical sensitivity there is a major problem with the first approach: the calculation of d_{12}, l_1, l_2 involves the ratio of 2 quantities including the term $||\mathbf{P_1Q_1} \times \mathbf{P_2Q_2}||$ which will be 0 or close to it if the lines associated to ls_1, ls_2 are parallel or nearly parallel, a case that occurs frequently for the legs of parallel robots. Hence the first approach will be numerically sensitive and although it involves less unknowns will be discarded.

The second and third approach have the same number of unknowns while the number of inequalities to verify is less for the second approach. Furthermore all unknowns in the second approach are naturally bounded (l_1, l_2 should be in the range [0,1] while θ has to lie in $[0, 2\pi]$). For the third approach we will see that bounds may be found for the coordinates of M but that these bounds will usually be quite large. But the second approach involves also the orthogonal vector basis $\mathbf{n_2}, \mathbf{v_2}, \mathbf{w_2}$ that is varying for a set of poses if the cylindrical element is a PC, a BMC or a PMC. In summary there may some advantage to use the second approach, although the third one may also have to be considered, while the first approach may usually be discarded in view of numerical robustness.

IV. Verification of interference over a workspace

For a set of poses (e.g. defined by a set of ranges for the pose parameters) it appears that there is little hope to be able to determine analytically if interference may occur, whatever is the chosen approach. Hence we will have to rely on a numerical procedure but one which ensure a reliable result. Collision detection may be formulated as a constrained optimization problem in which we minimize a distance between the cylindrical elements although we are not interested in finding the exact minimum but just to show that it is lower or greater than a given value. As in the past interval analysis has been able to solve many difficult robotics problems we have investigated this method. A basic concept of interval analysis is the *interval evaluation* of a multi-variate function $F(x_1, \ldots, x_n)$. Being given ranges $[\underline{x_i}, \overline{x_i}]$ for each unknown x_i the interval evaluation of F is a range $[a, b]$ such that the value of F for any instance x_i^r of the x_i's in their ranges verifies $a \leq F(x_1^r, \ldots, x_n^r) \leq b$. In other words the interval evaluation of a function gives bounds for the minimal and maximal values of the function over the unknowns ranges. A drawback of interval analysis is that a, b are usually only bounds for the minimal and maximal values of the function and are usually overestimating the minimum and maximum. But the level of overestimation is decreasing with the width of the unknowns ranges.

An interval evaluation may be calculated as soon as F is constituted of any classical mathematical function. Furthermore the calculation of an interval evaluation may take into account round-off errors i.e. the result is guaranteed. For example if $a > 0$, then we are sure that for any instance of the x_i F will always be positive.

A. Algorithm

The principle of an interval analysis based algorithm is always the same (although there are many different ways to implement it and to add heuristics that may change drastically the computation time). This principle will be

illustrated on the second approach for the intersection of a base cylinder $\mathcal{C}_2$ and a base mobile cylinder. $\mathcal{C}_1$. In that case the coordinates of the centers P_2, Q_2 of the bottom and top sections of $\mathcal{C}_2$ are fixed in the base frame. For $\mathcal{C}_1$ the center P_1 of the bottom section is fixed in the base frame while the coordinates of the center Q_1 of the top section are functions of the pose parameters.

Let $\mathbf{n_2} = (n_2^x, n_2^y, n_2^z)$ be the known unit vector of the axis of $\mathcal{C}_2$. Let $\mathbf{v_2}$ be the unit vector with components $(0, n_2^z/u, -n_2^y/u)$ with $u = \sqrt{n_2^{y2} + n_2^{z2}}$ and $\mathbf{w_2} = \mathbf{n_2} \times \mathbf{v_2}$. Clearly $\mathbf{v_2}, \mathbf{w_2}, \mathbf{n_2}$ is an orthonormal vector basis. Using equation (3) it may be seen that $\mathbf{OM}$ is a function of the unknown l_2, θ. Let M_1 be a point on the axis of $\mathcal{C}_1$ such that $\mathbf{OM_1} = \mathbf{OP_1} + \mathbf{P_1M_1}$ with $\mathbf{P_1M_1} = l_1\mathbf{P_1Q_1}$. The components of $\mathbf{OM_1}$ are functions of the pose parameters (through the coordinates of Q_1) and of l_1. Consequently the components of the vector $\mathbf{M_1M}$ are functions of the following unknowns $\mathcal{X}$: the pose parameters, l_1, l_2, θ.

Let us now assume that all the unknowns in $\mathcal{X}$ are constrained to lie in some range. A set of ranges for the 9 unknowns define a 9-dimensional *box* and we will use the term box for a set of ranges for the 9 unknowns. We will here first assume that the range for l_1, l_2 is [0,1] and $[0, 2\pi]$ for θ. The set of range for the pose parameters are obtained from the workspace $\mathcal{W}$ definition. Hence we are considering a box B_1 and by using interval analysis we are able to calculate an interval evaluation of the components of $\mathbf{M_1M}$ and then an interval evaluation $[a, b]$ of $||\mathbf{M_1M}||$. Three different cases may occur:

1) if $a > r_1$, then $\mathcal{C}_1$, $\mathcal{C}_2$ never intersect for any pose in $\mathcal{W}$
2) if $b \leq r_1$, then $\mathcal{C}_1$, $\mathcal{C}_2$ may intersect, provided that the conditions on $\mathbf{P_1M}.\mathbf{P_1Q_1}$, $\mathbf{Q_1M}.\mathbf{P_1Q_1}$ are satisfied
3) if $a \leq r_1, b \geq r_1$, then we are not able to determine the position of $||\mathbf{M_1M}||$ with respect to r_1.

In case 3 we will choose one variable x_i, bisect its range $I_i = [\underline{x_i}, \overline{x_i}]$ and create 2 new boxes B_2, B_3 by keeping the same ranges than in B_1 for all variables except for the variable x_i for which the range will be $[\underline{x_i}, (\underline{x_i} + \overline{x_i})/2]$ for B_2 and $[(\underline{x_i} + \overline{x_i})/2, \overline{x_i}]$ for B_3. These 2 boxes are stored in a list $\mathcal{L}$. The algorithm will then process all the boxes in the list, starting with B_2. As soon as a box has been considered the algorithm will process the next box in $\mathcal{L}$.

For case 2 we will compute the interval evaluation $[p_1, q_1]$, $[r_1, s_1]$ of $\mathbf{P_1M}.\mathbf{P_1Q_1}$, $\mathbf{Q_1M}.\mathbf{P_1Q_1}$. If $p_1 > 0$, $s_1 < 0$ then the cylinders intersect, if $q_1 < 0$ or $r_1 > 0$, then the cylinders will not intersect for this set of ranges. If $p_1, r_1 < 0$ and $s_1, q_1 > 0$, then we will proceed as in case 3 by creating 2 new boxes that will be added to the list $\mathcal{L}$.

The algorithm will stop either when it has been determined that the two cylinders intersect or when all the boxes in the list have been processed, in which case the cylinders do not intersect.

B. Possible workspace definition

As for the workspace in which is constrained to lie the platform, the algorithm is highly flexible and allows to deal with many cases:

- the workspace may be a trajectory with the pose parameters being almost arbitrary functions of the time T, that we may suppose to be in the range [0,1]. Note that in that case a small change in the algorithm allows to determine the lowest time at which an interference occurs. It is also possible to add bounded perturbations on the trajectory representing uncertainties, for example due to control error
- the algorithm is able to deal with 6D workspace, described as a set of ranges for the pose parameters (a *box* in the 6D space). But we may also deal with more complex workspace for the location of C as soon as we are able to find a bounding box of the workspace and design a test to determine if a given box is fully inside or outside the workspace. For example if the workspace is a sphere we use the box algorithm, initialized with the bounding box of the sphere, and we use the test before processing a box. If the test indicates that a box is outside the sphere we just skip this box. If the box is only partly inside the sphere, then we check the intersection of the elements but if there is an intersection we still bisect the box. We may also introduce additional constraints to restrict the workspace. For example mechanical limits on the motion of the passive joints may be added. Assume that the S joints located on the base may rotate by at most an angle μ around a vector $\mathbf{r}$. Hence at a given pose the constraint will be satisfied if $\mathbf{AB}.\mathbf{r}/||\mathbf{AB}|| \geq \cos(\mu)$, an inequality that may be easily incorporated in the algorithm.
  Note also that symmetry in the robot and in the workspace may be used to decrease the computation time.

C. Intersection cases

We will now summarize how the previous algorithm will be used at best for each pair of basic cylindrical elements. Indeed an appropriate choice of which is cylinder 1 and 2 of section II-B will play an important role in the computation time. Using the notation of section II-B we will indicate by a superscript which elements are affected to $\mathcal{C}_1$ and $\mathcal{C}_2$. For example 2^1 will indicate that the cylinder 2 of section II-B will be $\mathcal{C}_1$. If no superscript is present, then we may use indifferently $\mathcal{C}_1$ or $\mathcal{C}_2$ in the calculation of section II-B. We will use the notation $1(\mathbf{X})$ to indicate that the elements of cylinder 1 are a function of the pose parameters. A $\times$ symbol will be used to indicate that for this pair interference does not depend upon the pose. The best combination are indicated in Table I.

V. Implementation and test

A. Implementation

To test the interference algorithm we have used the C++ interval analysis library `ALIAS` which is interfaced

$\mathcal{C}_2$	BC	PC	BMC	PMC
$\mathcal{C}_1$=BC	$\times$	2^1, $1^2(\ )$	2^1, $1^2(\ )$	2^1, $1^2(\ )$
$\mathcal{C}_1$=PC	2^2, $1^1(\ )$	$\times$	$2^1(\ )$, $1^2(\ )$	$2^1(\ )$, $1^2(\ )$
$\mathcal{C}_1$=BMC	2^2, $1^1(\ )$	$1(\ )$ $2(\ )$	$1(\ )$ $2(\ )$	$1(\ )$ $2(\ )$
$\mathcal{C}_1$=PMC	2^2, $1^1(\ )$	$1(\ )$ $2(\ )$	$1(\ )$ $2(\ )$	$1(\ )$ $2(\ )$

TABLE I

BEST CHOICE OF CYLINDERS NUMBERING FOR THE POSSIBLE COMBINATION OF BASIC CYLINDRICAL ELEMENTS.

with the symbolic software `Maple`. The inequalities that must be verified for the interference check are obtained through `Maple`: this allow to modify at will the robot geometry or even its mechanical structure. As soon as the inequalities have been established a specific `Maple` procedure is called that allows to use a C++ solving method of `ALIAS` to determine if there is at least one solution to the inequalities test (in most cases the solution will be a box and for any value of the unknowns in this box there will be interference). This `Maple` procedure generates a C++ code that is run and whose result is returned to `Maple`. It must be understood that the algorithm proposed in section IV-A describes only the basic of the method. To be efficient the implementation requires some expertise in interval analysis. For example in the test implementation we use:

- a recursive interval evaluation of the inequalities based on the interval evaluation of its derivatives. As soon as one of these interval evaluations has a constant sign, then the calculation of the lower and upper bound of the interval evaluation may be done with a fixed value of one unknown, leading to a sharper interval evaluation
- a simplification procedure such that the interval evaluation of terms that appear multiple time in the inequalities are interval evaluated only once (this is especially useful for interval evaluation of trigonometric functions that are relatively computer intensive)
- filtering strategies that allow either to determine that a box cannot satisfy the inequalities or reduce the size of the box.

A drawback of this approach is that in its current version `ALIAS` is able to deal only with one given set of inequalities at a time. Hence a program has to be generated for each check of the intersection of each pair of elements, while it will have been more efficient to design a single program that check the intersection of each pair in a single step.

For using this procedure it is necessary to indicate range for each unknown. This is not a problem for the first and second approach as all the unknowns for these approaches are naturally bounded. For the third approach we write $\mathbf{OM} = \mathbf{OP_1} + \alpha\mathbf{P_1Q_1}$, where α lie in the range [0,1]. For a given box we can compute an interval evaluation of the coordinates of $\mathbf{OM}$ and thus get the initial box.

B. Test

To test the algorithm we have considered a wire robot with a base radius of 100 mm and a platform radius of 70 mm. The wires may be considered as a set of BMC $\{\mathcal{W}_1, \ldots, \mathcal{W}_6\}$ with a diameter of 2 mm. Furthermore the platform has a PC $\mathcal{P}_1$ centered at (0,0,0) with height 52 mm and radius 46 mm. On the base we have a BC $\mathcal{B}_1$ also centered at (0,0,0) with height 52 mm and radius 46 mm. The cylindrical elements $\mathcal{P}_1$ and $\mathcal{B}_1$ are connected through a BMC $\mathcal{F}_1$. This robot has hence a total of 9 cylindrical elements that may intersect. We have to perform a total of 33 interference check: 15 for the intersection between the $\mathcal{W}_i$ and 6 for the intersection between the $\mathcal{W}_i$ and $\mathcal{P}_1, \mathcal{B}_1, \mathcal{F}_1$ (we assume no intersection between $\mathcal{P}_1, \mathcal{B}_1$).

We have considered that the platform has to move in 2 different types of 6D workspace:

1) *workspace* G_1: a 6D box defined by the ranges $x, y \in [-40, 40]$, $z \in [130, 210]$ and a range [-10,10] degrees for the yaw, pitch, roll angles
2) *workspace* G_2: a sphere centered at (0,0,170) and of radius 40 for C and a range [-10,10] degrees for the yaw, pitch, roll angles
3) *trajectory* T_1: a circular trajectory in the $x - y$ plane centered at (0,0,170) with radius 20 (i.e. $x = 20\sin(2\pi T)$, $y = 20\cos(2\pi T)$ with the Euler angles equal to 0
4) *trajectory* T_2: the same circular trajectory but with $\psi = 2\pi T$, $\theta = 5$ degree and $\phi = -\psi$. This correspond to the case where the normal of the platform is oriented toward the center of the circle with a constant 5 degree tilt
5) *trajectory* T_3: the same circular trajectory but with $\psi = 2\pi T$, $\theta = 5$ degree and $\phi = 0$

The computation for the various interference checks, using the second approach, are presented in Table II. The lower computation time for $\mathcal{W}_i \cap \mathcal{P}_1$, $\mathcal{W}_i \cap \mathcal{F}_1$ for the workspace G_1 occurs because an interference is detected (see figure 2) while there is no interference for G_2, T_1, T_2. Note also that the times obtained for $\mathcal{W}_i \cap \mathcal{F}_1$ have been obtained by using a distributed implementation with 16 computers. It is important to emphasize that if the interference check was performed in a single program the computation time will be slightly reduced. At each step of the bisection process we will check individually each set of inequalities. But a remember flag allows to avoid evaluating inequalities which have already been verified

	$\mathcal{W}_i \cap \mathcal{W}_j$	$\mathcal{W}_i \cap \mathcal{B}_1$	$\mathcal{W}_i \cap \mathcal{P}_1$	$\mathcal{W}_i \cap \mathcal{F}_1$	Total
G_1	19.9	15.6	498	2833	3366.5
G_2	21	16.4	528	1389	1954.4
T_1	5.3	1.9	4.1	3.1	14.4
T_2	7.25	4.5	9.8	6	27.55

TABLE II

COMPUTATION TIME IN SECOND FOR THE INTERFERENCE CHECK ON A DELL D400 LAPTOP, FOR VARIOUS TYPES OF WORKSPACE.

for the box from which is issued the current box. Hence the computation time will be almost reduced to the pair that has the worst computation time. For example for G_1 the worst pair is $\mathcal{W}_5 \cap \mathcal{F}_1$ with a computation time of 2988s. Consequently the whole check of G_1 will have a computation time of about 3000 seconds.

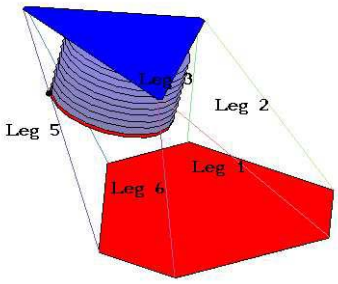

Fig. 2. An interference case that is detected for workspace G_1. Leg 5 collides with a PC.

For trajectory T_3 collision between cables are detected on the trajectory for all pairs of cables, in a computation time of 7.3s (roughly 0.5s if all the test were performed in a single step). Figure 3 presents examples of collision between the legs. It can be established that the first collision between the legs occurs at time 0.3808 between leg 2 and 6.

Our test have shown that the first and third approach are usually much less efficient than the second approach (except for $\mathcal{W}_i \cap \mathcal{W}_j$ for which the third approach is approximately 10% faster than the second approach).

VI. Conclusion

We have shown in this paper that the difficult problem of interference between the bodies of a parallel robot over a 6D workspace or a trajectory may be solved for cylindrical shape of the bodies. Although different formulations are available to test interference not all of them are equivalent in term of efficiency. However it is necessary to extend the possible shape of the bodies to other cases, such as spheres, parallelepiped, which may be more difficult to deal with.

Other difficult problems will be the maximal workspace and appropriate design one. The proposed algorithm allows to detect an interference but we may also be interested to determine the maximal interference-free workspace or to be able to determine the design parameters so that a given workspace will be interference free. The proposed interference algorithm may be extended to deal with the maximal workspace problem but the design problem is much more difficult.

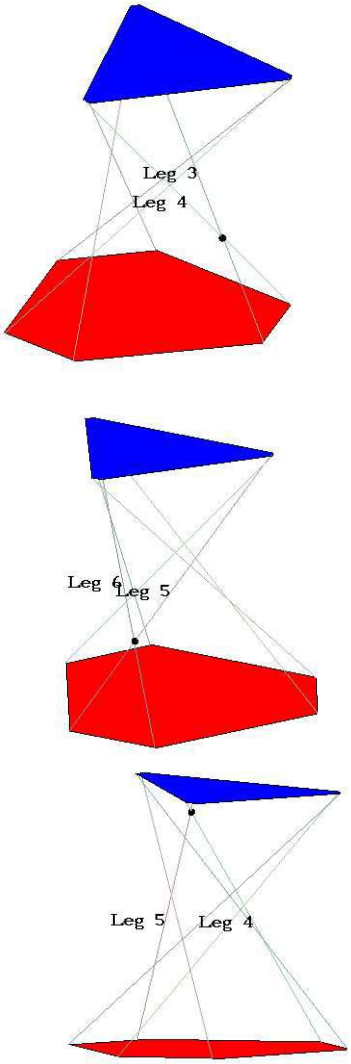

Fig. 3. Examples of cables interference for trajectory T_3

References

[1] Chablat D. and Wenger P. Moveability and collision analysis for fully-parallel manipulators. In *12th RoManSy*, pages 61–68, Paris, July, 6-9, 1998.

[2] Cortés J. and Siméon T. Probabilistic motion planning for parallel mechanisms. In *IEEE Int. Conf. on Robotics and Automation*, pages 4354–4359, Taipei, September, 14-19, 2003.

[3] Gosselin C. Determination of the workspace of 6-dof parallel manipulators. *ASME J. of Mechanical Design*, 112(3):331–336, September 1990.

[4] Merlet J-P. Geometrical determination of the workspace of a constrained parallel manipulator. In *ARK*, pages 326–329, Ferrare, September, 7-9, 1992.

[5] Merlet J-P. Analysis of the influence of wire interference on the workspace of wire robots. In *ARK*, pages 211–218, Sestri-Levante, June 28- July 1, 2004.

[6] Merlet J-P. Determination of 6D workspaces of Gough-type parallel manipulator and comparison between different geometries. *Int. J. of Robotics Research*, 18(9):902–916, October 1999.

[7] Merlet J-P. and Daney D. A formal-numerical approach to determine the presence of singularity within the workspace of a parallel robot. In F.C. Park C.C. Iurascu, editor, *Computational Kinematics*, pages 167–176. EJCK, Seoul, May, 20-22, 2001.

[8] Wenger P. and Chablat D. Workspace and assembly modes in fully parallel manipulators: a descriptive study. In *ARK*, pages 117–126, Strobl, June 29- July 4, 1998.