2014 IEEE/ASME International Conference on
Advanced Intelligent Mechatronics (AIM)
July 8-11, 2014. Besançon, France

Interference Detection for Cable-Driven Parallel Robots (CDPRs)

Laurent Blanchet[1] and Jean-Pierre Merlet[2]

***Abstract*— The main advantage of CDPRs is large workspaces. However, multiple legs and large workspace are both factors for interference. We consider a CDPR platform within a 6D workspace, and as sources of interference the collisions with the robot's environment and self interference. We present two algorithms and their interval analysis-based applications to handle the different types of interference. Finally, the efficiencies of the algorithms are presented.**

***Index Terms*— Modeling and Design, Cable-Driven Parallel Robots, leg interference, collision, robotics, Interval Analysis.**

I. INTRODUCTION

In the last twenty years or so, a new class of parallel robots, called Cable-Driven Parallel Robots (CDPRs), has been the subject of several research projects [7], [4]. A CDPR is composed of a mobile platform suspended by m cables to a base; the set of parameters involved in the design and their domains will be called the robot's configuration. Cable lengths control the pose of the platform, see figure 1. A CDPR can be considered as a mechanical system of known or encapsulated precision which must satisfy the set of constraints of a given application. As well as practical errors (real position of the cables exit points, of the attachment points on the platform, control errors, real cables versus model, ...), it must also account for numerical round-off errors in the constraints calculation.

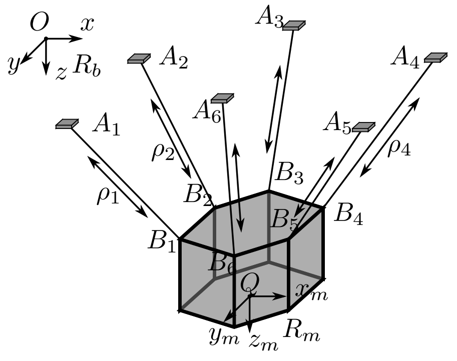

Fig. 1. Schematic of a CDPR with its parametrization.

To account for these errors, interval arithmetic may be used to certify performances, see [6] for a quick introduction, survey results and an extensive bibliography about this arithmetic. Instead of single values for parameters, a range of all possible values for each parameter is used, and constraint satisfaction may be guaranteed in spite of variations of the parameters. Indeed interval arithmetic allows the qualification of the constraints for the interval values of the parameters into three statuses: always valid (satisfied for all values of the parameters range), always invalid (not satisfied by any value of the parameters in their ranges) or undetermined (in the parameters ranges, some values satisfy the constraint, some other do not). This guarantee comes at the price of computation time.

The main motivation for designing a CDPR is its extended workspace. However, self-collisions and collisions in general are the main drawbacks of dealing with large workspaces. This paper addresses this issue, with the provision of a "straight" cables hypothesis, meaning that the CDPR has no deliberately crossing cables, and with the aim of avoiding such interference. In the negative, Wischnitzer *et al.* devised some strategies in [13].

In the following we call *box* $\mathcal{B}$ a n-vector of intervals. Such a box is defined by the values of its boundaries. We will also make use of 2B-consistency. A consistency technique ([8]) removes values from the bounds of the variables domains which are not consistent with the constraint(s), thus without losing solutions. The basic algorithm is a branch and prune scheme as described by Van Hentenryck in [12], with branching occurring when a constraint is evaluated as undetermined over the considered pose box, i.e. the 6-dimensional interval vector representing the domain of definition of the robot end-effector position and orientation. By 'application', we denote the total user-specified workspace, including a list of pose boxes and trajectories in specified coordinate and angle systems, and associated sets of external wrenches. The workspace is called WS in equations, while the sub-set of pose with interference is called WS_I. We will use the terms 'object' for a generic object represented by the facets of its manifold, and 'obstacle' for fixed objects. Finally, $\square a$ designate the domain of the variable a.

A versatile algorithm relying on intersection queries between indexed facets, mainly used for detecting interference with external obstacles (layout of the factory for example) is first described. This algorithm relies on the CGAL library, see [1]. Following is a leg-leg interference algorithm based on the work of Merlet *et al.* in [9]. The first one allows one to handle all types of intersections with relatively simple approaches, but provides rigorous answers only to non-interference cases. The second algorithm is applied here only on the leg/leg collision detection, and provides definitive answers on non- or with-interference cases. The differences between the last algorithm and the reference paper are the extension to CDPRs, a slightly different approach of the geometrical problematic and the 2B reduction presented in III-B ii.) and assessed at the end of that section.

This research has received partial funding from the European Community's Seventh Framework Program under grant agreement NMP2-SL-2011-285404 (CABLEBOT).

[1]Laurent Blanchet is a Ph.D. student with Inria S-A Méditerranée, Team Hephaistos, address below laurent.blanchet@inria.com

[2]Jean-Pierre Merlet is with Inria Sophia-Antipolis Méditerranée, Team Hephaistos, 2004 route des Lucioles - BP 93, 06600 Sophia-Antipolis, France jean-pierre.merlet@inria.com

II. Object interference

Preliminaries: Objects are represented as a list of facets in an STL file. The only constraint on the objects is to respect the STL specification of a closed manifold. The facets of the obstacles are indexed into an Axis-Aligned Bounded Box (AABB) tree, with origin shift and exact similitude transformation. Intersection queries are handled by the 3D Fast Intersection and Distance Computation algorithm [1].

The concept of *swept space* is used in thi paper. To the knowledge of the author, Boyse in [2] was the first to introduce it, in an interference algorithm detecting potential interference between moving parts of a mechanism. The idea is to consider the whole space described by a given entity as a virtual part. Absence of time dependance provides for fast algorithms but determining the swept space is often an problem (see the survey [5]). In the current case, interval analysis allows for fast, simple, but unfortunately overestimated evaluation of the swept spaces.

We define necessary conditions for collision (border, pose box of undetermined status), and sufficient conditions (invalid pose box). With the hypothesis that the list is exhaustive, the negation of all conditions for the whole pose box implies absence of collision (valid pose box).

Necessary conditions for collision: A bisection is needed to reduce the evaluation overestimation if any of the following interference is detected for part of the pose box:

- obstacle against platform (attachment point) swept space: the obstacle intersects part of the platform swept space. This intersection can also be evaluated through the attachment point swept spaces. All swept spaces are obtained through interval evaluation. Example figure 2.
- obstacle against cable sheaths: the obstacle intersects part of a cable swept space. Cable swept spaces are 4 to 8 faces polyhedra and will be called sheaths. Example on fig. 3.

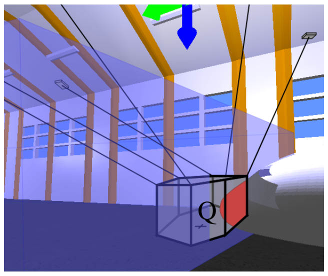

Fig. 2. Example of a mobile platform interference with an object.

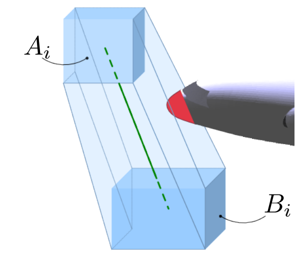

Fig. 3. Depiction of a cable sheath and an interference with an object.

Sufficient conditions for collision: The pose box is declared fully outside of valid workspace if any of the following intersections is detected for part of the pose box:

- object against configuration space: the object intersects part of the configuration space considered;
- object against centre of any cable sheath: if the object intersects the segment from the centre of the anchoring space to the centre of the swept space of the platform attachment point of a given leg, it is considered not caused by overestimation, thick line on figure 3;
- object against application space: the object intersects part of the desired workspace, see figure 4.

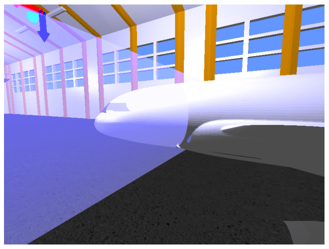
Fig. 4. Example of an application with a self-interfering definition.

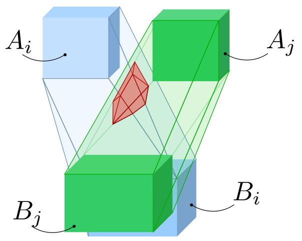

Fig. 5. Depiction of the sheath-sheath interference test; example of an intersection (in red) of the sheaths.

III. Cable/cable interference

Two poses $\mathbf{X_1}$, $\mathbf{X_2}$ of the application may satisfy all constraints, but if getting from one to the other causes the cables to cross, then $\mathbf{X_2}$ won't be reached, at least not with the straight cable control assumed throughout this section. If a trajectory from $\mathbf{X_1}$ to $\mathbf{X_2}$ has cables colliding, then there exists at least one pose $\mathbf{X_C} \subset WS_I$ with crossing cables. Even if $\mathbf{X_1}, \mathbf{X_2}$ are not in the same pose box after bisections, the algorithm will close in on $\mathbf{X_C}$ and all poses of WS_I.

If such scenario occurs then there is an intersection of the interval-evaluated swept spaces. Such a scenario is also equivalent to a small distance between the axes of the two legs. The first implication is implemented as a first, theoretically fast algorithm using swept spaces on the same principle as the object/cable interference test above, and may either qualify a pose box as interference-free or request a split, as it is not possible to differentiate false-positive from overestimation. The second equivalence is implemented as a complete, reliable algorithm, looking for and at the shortest distance(s) between two legs.

A. Swept space based algorithm.

Like in the object versus cable test, leg's swept spaces (sheaths) are built around the anchoring region of a given leg and the space swept by the other end of the leg considering all poses of the platform of a given pose box. The sheaths are then tested for intersection versus each one of the other legs sheaths, see figure 5. As this algorithm loose the correlation between the attachments on the platform, it is very fast when there is no possibility of interference, but leads to numerous splits and often undetermined exits in the rest of the cases.

B. Shortest distances based algorithm.

In [9], Merlet *et al.* describe three forms of an interval-evaluation based leg/leg interference algorithm for parallel robots. We describe here a new algorithm which can be summed up in three to four steps with reference to the schematic 6 and its parameterisation:

i.) Determination of the radius of the cylinder/sheath of the legs by computing the maximum sagging value from the catenary model.
ii.) Determination of sets of values of the model parameters which might lead to the minimal distance between two given legs over a box of poses.
iii.) Determination of the range of values of the potential minimal distances for each of the sets of parameters determined in the previous step.
iv.) In some cases, evaluation of the legs situation with reference to each other.

Presentation of the algorithm: Using the parameterisation of figure 6, we can write the position vectors of points M_i and M_j in the base frame as the generic equation $\mathbf{OM} = \mathbf{OA} + l\mathbf{AB}$. Using Chasles relation and distributive property of the parameters, those vectors can be re-written as equations (1), functions of the following known quantities:

- $\mathbf{OA_i}$, from the robot's configuration.
- $\mathbf{QB_i}|_{\mathcal{R}_b} = \underline{\underline{R}}\mathbf{QB_i}$, position of the attachment points in the base frame, depending on the robot's configuration and platform orientation matrix within $WSR \subset SO(3)$. $\mathcal{WSR} \subset \mathbb{R}^3$ is the current workspace orientation box, sub box of the specified workspace (a.k.a. the application); it spans the set $WSR \subset SO(3)$ of matrices.
- $\mathbf{OQ}$, end-effector position within $\mathcal{WS} \subset \mathbb{R}^3$, current workspace position box, sub box of the application.

$$\forall \mathbf{OQ} \in \mathcal{WS}, \forall \underline{\underline{R}} \in WSR, \exists (l_i, l_j) \in [0\,;1]^2 :$$

$$\mathbf{OM_i} = \mathbf{OA_i} + (\mathbf{QB_i}|_{\mathcal{R}_b} - \mathbf{OA_i})\, l_i + \mathbf{OQ} l_i \quad (1a)$$

$$\mathbf{OM_j} = \mathbf{OA_j} + (\mathbf{QB_j}|_{\mathcal{R}_b} - \mathbf{OA_j})\, l_j + \mathbf{OQ} l_j \quad (1b)$$

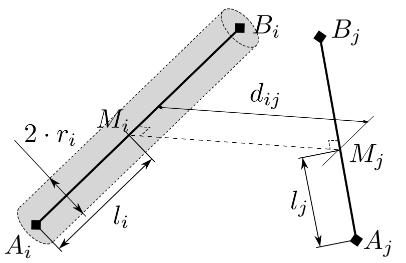

Fig. 6. Schematic of the two segments representing two legs of a parallel robot and parametrization for the minimum gap-based interference test.

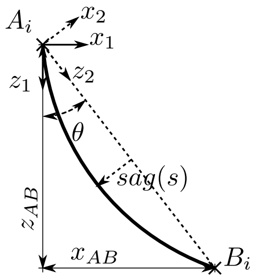

Fig. 7. Schematic of a cable with elasticity and sagging, under the catenary model.

We derive equation (2) from the equations (1).

$$\forall \mathbf{OQ} \in \mathcal{WS}, \forall \underline{\underline{R}} \in WSR, \exists (l_i, l_j) \in [0\,;1]^2 :$$

$$\mathbf{M_jM_i} = \mathbf{A_jA_i} + \left(\underline{\underline{R}}\mathbf{QB_i} - \mathbf{OA_i}\right) l_i - \left(\underline{\underline{R}}\mathbf{QB_j} - \mathbf{OA_j}\right) l_j + \mathbf{OQ}\left(l_i - l_j\right) \quad (2)$$

We will use the distance $d_{ij} = \|\mathbf{M_jM_i}\|$ to evaluate the range of distances from leg i to leg j for given box(es) of parameters $p = (x, y, z, \alpha, \beta, \gamma, l_i, l_j)$. Moreover, a re-arrangement of equation (2) will be used to do a useful 2B-consistency on the domain of either one of the last two parameters, which is particularly effective for CDPRs. The partial derivatives of d_{ij} will also be used in a local branch and prune algorithm to find out those boxes.

Step i.) Computation of a maximum sagging value: In the frame of the CableBot European project, we compared a sagging simulation experimentation by Technalia [11] with simulation results obtained using the catenary model popularized by Irvine in [3]. This qualitative comparison is presented figure 8.

The approach used in [9] is suited for parallel robots with non-deformable legs, not CDPRs with sagging cables. In regards to this qualitative comparison, we make the hypothesis that the cable is encapsulated in a cylinder whose radius is the maximum sagging value obtained from the catenary model over the configuration space, the pose box

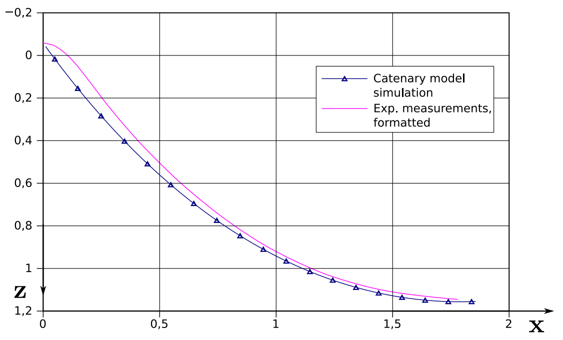

Fig. 8. Depiction of the qualitative comparison of Technalia's sagging experiments versus catenary model based simulation results.

and for the wrench box of the application. This hypothesis is likely correct as long as the cable is taut. As Merlet shows in [10], several poses lead to at least one slack cable, hence an additional hypothesis: when the cable is slack, the tension, reduced to the cable self-weight and the bending rigidity, is supposedly negligible in front of a taut cable tension.

Since the longer the leg is, the more massive is the hanging part of the cable, we first look for the couple (anchor space, workspace box) which leads to the higher uncoiled length.

Taking the higher end of the cable as origin of a new local frame R_1 whose $\mathbf{x_1}O\mathbf{z_1}$ plane contains the cable (see figure 7), we integrate the catenary model equations (3) of the normalized horizontal and vertical parametrized coordinates of a cable (noted here $x(s)$ and $z(s)$) using $dL_0 s = L_0 ds$ from their differentiated expressions (4). We note the weight of the uncoiled length of the cable $P_{c_0} = \rho_0 g L_0$ and the difference of the z component of the force applied on the cable at the platform attachment point, by the weight of the cable portion left from abscissa s to lower point B, $\Delta F_z(s) = F_z + P_{c_0}(s-1) = \Delta F_z(0) + sP_{c_0}$.

$$x(s) = \frac{F_x s L_0}{EA_0} + \frac{|F_x| L_0}{P_{c_0}}\left(\operatorname{arcsinh}\left(\frac{\Delta F_z(s)}{F_x}\right) - \operatorname{arcsinh}\left(\frac{\Delta F_z(0)}{F_x}\right)\right) \quad (3a)$$

$$z(s) = \frac{\Delta F_z(0)\, s L_0}{EA_0} + \frac{P_{c_0} L_0}{EA_0}\frac{s^2}{2} + \frac{L_0}{P_{c_0}}\left(\sqrt{F_x^2 + (\Delta F_z(s))^2} - \sqrt{F_x^2 + (\Delta F_z(0))^2}\right) \quad (3b)$$

$$\frac{dx}{ds} = \frac{F_x}{EA_0} + \frac{F_x}{\sqrt{F_x^2 + (\Delta F_z(s))^2}} \quad (4a)$$

$$\frac{dz}{ds} = \frac{\Delta F_z(s)}{EA_0} + \frac{\Delta F_z(s)}{\sqrt{F_x^2 + (\Delta F_z(s))^2}} \quad (4b)$$

We use a second local frame R_2, which is a rotation of angle θ of the current one such that both attachment points of the cable (noted A and B) are on the new $\mathbf{z_2}$ axis. In this frame, the parametrized sagging function is simply the opposite of the $x_2(s)$ parametrized coordinates, issuing equation (5) and its derivative. Then, we solve (6) in order to find the extrema of (5).

$$sag(s) = -\frac{z_{AB}\, x(s)}{\sqrt{x_{AB}^2 + z_{AB}^2}} + \frac{x_{AB}\, z(s)}{\sqrt{x_{AB}^2 + z_{AB}^2}} \quad (5)$$

$$\frac{d\,(sag)}{ds} = 0 \Leftrightarrow -z_{AB}\frac{dx}{ds} + x_{AB}\frac{dz}{ds} = 0 \quad (6)$$

However, (6) is equivalent to (7), whose first factor is always strictly positive, whatever the value of s, ensuing the equivalence (8).

$$\left(\sqrt{F_x^2 + \Delta F_z(s)^2} + EA_0\right)(x_{AB}\Delta F_z(s) - z_{AB}F_x) = 0 \quad (7)$$

$$\frac{d\,(sag)}{ds} = 0 \Leftrightarrow s_{\text{max sag}} = \frac{z_{AB}}{x_{AB}}\frac{F_x}{P_{c_0}} - \frac{\Delta F_z(0)}{P_{c_0}} \quad (8)$$

At this point, the F_x and F_z variables are any values of their respective domains, while x_{AB}, z_{AB}, and P_{c_0} are single values. As all equations are for a normalized parameter, we can safely discard solutions not in the $[0\,;1]$ interval. Then we take the biggest value from the evaluation of the sagging function for all values of $s_{\text{max sag}}$ left.

Step ii.) Determination of the boxes of parameters of potential minimum gap: We need to determine a set $\mathcal{S}$ of boxes of parameters $\mathcal{B}_k$ such that one of them contains the values of the eight parameters $p = (x, y, z, \alpha, \beta, \gamma, l_i, l_j)$ leading to the minimal gap between leg i and leg j. To do that, we use a branch and prune algorithm to solve the equations $\frac{\partial d_{ij}}{\partial p_i} = 0$, for $d_{ij} = \|\mathbf{M_j M_i}\|$. With no loss of generality, we can write that $\exists d \in \mathbb{R}, \exists \mathbf{u} \in [-1\,;1]^3$ such that $\mathbf{M_i M_j} = d\mathbf{u}$.

We begin by a 2B reduction on the domain of either l_i or l_j. Let us choose parameter l_j and proceed to the following substitution: $\lambda = l_j - l_i$. Then, from (2), we can derive (9).

$$\forall \mathbf{OQ} \in \mathcal{WS}, \forall \underline{\underline{R}} \in WSR, \exists l_i \in [0\,;1], \exists \lambda \in [-1\,;1],$$
$$\exists d \in \mathbb{R}, \exists \mathbf{u} \in [-1\,;1]^3 :$$
$$\mathbf{OQ}\lambda = \mathbf{A_j A_i} + \left(\underline{\underline{R}}\mathbf{B_j B_i} - \mathbf{A_j A_i}\right) l_i - \left(\underline{\underline{R}}\mathbf{QB_j} - \mathbf{OA_j}\right)\lambda - d\mathbf{u} \quad (9)$$

Let us first encapsulate the distance vector for interference poses. For some $\mathbf{Q_I}$ and $\underline{\underline{R_I}}$ a pose for which there is a leg/leg interference, as the distance vector $\mathbf{M_i M_j}$ is the shortest distance between the two legs, we have $\|\mathbf{M_i M_j}\| \leq r_i + r_j$. Moreover, $\mathbf{M_i M_j} \in [-\frac{a}{2}\,;\frac{a}{2}]^3$, see figure 9. By choosing a so that the cube (C) is the circumcube of the sphere (S), we have $a = r_i + r_j$.

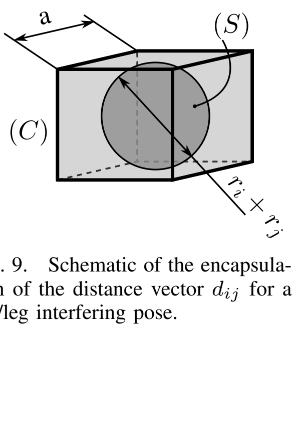

Fig. 9. Schematic of the encapsulation of the distance vector d_{ij} for a leg/leg interfering pose.

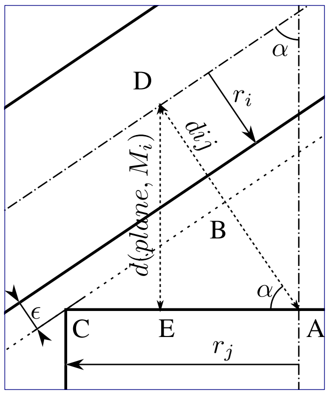

Fig. 10. Schematic of the "cap" situation in case one leg is right above - or below - the other, and parametrization.

Now, for the derivation of the equations for the 2B reduction, using (9) we build the set $WS_I(\lambda)$, equation (10), which represents a linear combination of the set of positions for which there are leg/leg interference(s) for a given value of the parameter λ, and the box $\mathcal{H}(\lambda)$, equation (11), a known box depending on parameter λ, and superset of $WS_I(\lambda)$.

$$WS_I(\lambda) = \big\{\lambda\mathbf{Q_I} \in \lambda.\mathcal{WS} \;\big|\; \exists l_i \in [0\,;1], \exists \underline{\underline{R}} \in WSR,$$
$$\exists d < r_i + r_j, \exists \mathbf{u} \in [-1\,;1]^3, \mathbf{OQ_I}\lambda = \mathbf{A_j A_i} +$$
$$\left(\underline{\underline{R}}\mathbf{B_j B_i} - \mathbf{A_j A_i}\right) l_i - \left(\underline{\underline{R}}\mathbf{QB_j} - \mathbf{OA_j}\right)\lambda - d\mathbf{u}\big\} \quad (10)$$

$$\mathcal{H}(\lambda) = \big\{\mathbf{A_j A_i} + \left(\underline{\underline{R}}\mathbf{B_j B_i} - \mathbf{A_j A_i}\right) l_i$$
$$- \left(\underline{\underline{R}}\mathbf{QB_j} - \mathbf{OA_j}\right)\lambda - d\mathbf{u} \;\big|\; \forall l_i \in [0\,;1],$$
$$\forall \underline{\underline{R}} \in WSR, \forall d < r_i + r_j, \forall \mathbf{u} \in [-1\,;1]^3\big\} \quad (11)$$

This inclusion is equivalent, $\forall \lambda \in [-1\,;1]$, to the two inequalities (12).

$$\text{Inf}(\mathcal{H}(\lambda)) \leq \text{Inf}(WS_I(\lambda)) \quad (12a)$$
$$\text{Sup}(WS_I(\lambda)) \leq \text{Sup}(\mathcal{H}(\lambda)) \quad (12b)$$

Moreover, the definition (10) implies the inequalities (13).

$$\text{Inf}(WS_I(\lambda)) \leq \text{Sup}(\Box\mathbf{OQ}\lambda) \quad (13a)$$
$$\text{Inf}(\Box\mathbf{OQ}\lambda) \leq \text{Sup}(WS_I(\lambda)) \quad (13b)$$

Combining (12) and (13) leads to (14).

$$\begin{aligned} \forall \lambda \in [-1\,;1], &\quad \text{Inf}(\mathcal{H}(\lambda)) \leq \text{Sup}(\Box\mathbf{OQ}\lambda) \\ \forall \lambda \in [-1\,;1], &\quad \text{Inf}(\Box\mathbf{OQ}\lambda) \leq \text{Sup}(\mathcal{H}(\lambda)) \end{aligned} \quad (14)$$

It is important to notice in the inequalities (14) that the parameter λ is a single real number, not a set. This fact, combined with the linearity of the two interval operators $\text{Inf}(\mathbf{X})$ and $\text{Sup}(\mathbf{X})$, justifies the distributivity used to obtain inequalities (15). The two inequalities (14) are actually 6 inequalities, as they apply on all three translational dimensions. Using the notation $\mathbf{e}$ to represent a generic eigen vector of the box $\mathcal{WS}$, inequalities (15) follow.

$$\forall \lambda \in [-1\,;1], \forall \mathbf{e} \in \mathcal{WS},$$
$$\text{Inf}(\mathcal{H}(\lambda).\mathbf{e}) \leq \text{Sup}(\Box\mathbf{OQ}.\mathbf{e})\lambda \quad (15a)$$
$$\text{Inf}(\Box\mathbf{OQ}.\mathbf{e})\lambda \leq \text{Sup}(\mathcal{H}(\lambda).\mathbf{e}) \quad (15b)$$

Finally, with $\mathcal{H} = \{\mathcal{H}(\lambda) \mid \forall \lambda \in [-1;1]\}$, for all $\lambda \in [-1\,;1]$, and with $\mathbf{e}$ such that $0 \notin \Box\mathbf{OQ}.\mathbf{e}$, (15) brings (16).

$$\underbrace{\frac{\text{Inf}(\mathcal{H}.\mathbf{e})}{\text{Sup}(\Box\mathbf{OQ}.\mathbf{e})}}_{\lambda_{min}} \leq \lambda \leq \underbrace{\frac{\text{Sup}(\mathcal{H}.\mathbf{e})}{\text{Inf}(\Box\mathbf{OQ}.\mathbf{e})}}_{\lambda_{max}} \quad (16)$$

The interest of this framing of the λ parameter lies in the fact that for a CDPR, we will often have big values for the interval $\Box\mathbf{OQ}.\mathbf{z} = \mathcal{WS}|_3$ in order to mitigate the tension in the cables through their inclinations with reference to the vertical, ensuing small values over $\mathcal{H}.\mathbf{z}$ in front of the values of $\Box\mathbf{OQ}.\mathbf{z}$. It is even more interesting for almost planar configurations, which induces small values over the interval $\mathcal{H}.\mathbf{z}$. For the case defined in section IV, the size of the resulting λ domain was ranging from 2.52% to 16.34% of the initial domain size; the contraction resulted in no valid domain, eg no interference, for 7.14% of the search tree for a small workspace (SB) and 42.24% for a big one (BB).

The leg/leg interference algorithm continues with the branch and prune algorithm fig. 11 for each of the variables.

```
1) 2B on $\lambda$. If $\lambda_{max} < \lambda_{min}$, then EXIT, no interference for this pose box with the current configuration box.
2) initialize global variables: $i = 8$, $\mathcal{S} = \{\varnothing\}$
3) initialize local variables: $n = 1$, $W$
4) While $n < n_{max}$, do:
  a) $W := \text{eval}\left(\square \frac{\partial d_{ij}}{\partial p_i}\right)$
  b) if $0 \in W$ ...
     (b.i) but no (or not enough) variations on W, change $\square p_i := \text{Mid}(\square p_i)$
     (b.ii) and both $\text{Diam}(W) >$ min allowed for $\frac{\partial d_{ij}}{\partial p_i}$ and $\text{Diam}(\square p_i) >$ minimum allowed for parameter $p_i$, split $\square p_i$, store lower half in to-do list $\mathcal{L}$, and change $\square p_i := [\text{Mid}(\square p_i); \text{Sup}(\square p_i)]$
     (b.iii) but either $W$ or $\square p_i$ is not big enough, store $\square p_i$ in list $\mathcal{S}$, and if to-do list is not empty change $\square p_i := \mathcal{L}(last)$, else exit (to sub-step 5).
  c) if $0 \notin W$ ...
     (c.i) and first evaluation, hence $i == 1$, then $d_{ij}$ is monotonous with respect to parameter $p_i$. If the evaluation $W$ is negative, $d_{ij}(p_i)$ is decreasing; change $\square p_i := \text{Sup}(\square p_i)$. If the evaluation $W$ is positive, $d_{ij}(p_i)$ is increasing; change $\square p_i := \text{Inf}(\square p_i)$. Exit (to sub-step 5).
     (c.ii) and to-do list $\mathcal{L}$ is not empty, discarding current $\square p_i$ and change $\square p_i := \mathcal{L}(last)$
     (c.iii) and to-do list $\mathcal{L}$ is empty, check if there was at least one contribution of this variable domain to the list $\mathcal{S}$ and exit (to sub-step 5), else overestimation of the derivative: go to (c.i).
  d) $n = n + 1$, go to 4
5) if $i == 7$ (thus current parameter is $l_i$), $\square l_i = (\square l_i \cap \square (l_j - \lambda))$. If $\square l_i = \{\varnothing\}$, EXIT, no interference for this pose box with the current configuration box.
6) if $i > 1$, decrement i for next variable, go to 3
```

Fig. 11. Branch and prune solving algorithm for all parameters of the leg/leg interference tests.

Step iii.) Computation of all potential minimal distances between the legs i and j: Once determined all boxes of potential interference $\mathcal{S} = \{\mathcal{B}_k \mid \forall k \in [\![0; r]\!]\}$, the algorithm evaluate the distance interval d_{ij} between the legs i and j for parameters in $\mathcal{B}_k$, and compare it to the radii of the legs, see figure 12. Those are intervals; there is no interference if all distances from all the boxes (the lowest values of d_{ij} for all $\mathcal{B}_k$) test above the sum of the two radii; and there is interference if all distances from one of the boxes (the highest value of d_{ij} for any $\mathcal{B}_k$) test below the i-th leg radius. If all distances are above the i-th leg radius, but not above the sum of the two radii, there are several possibilities, see *Step iv.)*.

Step iv.) evaluation of the legs situation with reference to the other leg: If all distances are above the i-th leg radius, but not above the sum of the radii, it could be that the legs actually are interfering, the respective cylinders just crossing, but not as far as crossing each other's axes, or it could be that leg i is right above or below leg j, with no intersection.

To differentiate those two situations we begin by testing on the first possibility, see algorithm figure 13. From the vector

```
7) for all boxes in $\mathcal{S}$, $\text{eval}(d_{ij}(x, y, z, \alpha, \beta, \gamma, l_i, l_j))$.
  a) if it exists at least one box $\mathcal{B}_k$ in list $\mathcal{S}$ such that $\text{Sup}(d_{ij}(\mathcal{B}_k)) \leq r_i$, EXIT, there is a leg/leg interference in this pose box with the current configuration box.
  b) if it exists at least one box $\mathcal{B}_k$ in list $\mathcal{S}$ such that $r_i + r_j \in d_{ij}(\mathcal{B}_k)$, EXIT, need bisection of pose box.
  c) if all boxes $\mathcal{B}_k$ in list $\mathcal{S}$ satisfy $\text{Inf}(d_{ij}(\mathcal{B}_k)) > r_i + r_j$, EXIT, no interference for this pose box with the current configuration box.
  d) else (if $r_i < \text{Sup}(d_{ij}(\mathcal{B}_k)) \leq r_i + r_j$) there might be interference, if leg $j$ is not right above or below leg $i$. Calling Step iv.).
```

Fig. 12. For all sets of potential leg/leg interference, computation of leg/leg distance and test for interference.

$A_j B_j$ of leg j, we build the generic cartesian equation of a perpendicular plane (or actually, a set of planes, as the normal vector of the plane is a box). Then, we check if the box of the potential positions of M_i contains a point that could be on the planes at either one the leg's end points. If so, we need a bisection to reduce overestimation and clear up the situation. Otherwise, checking the signs of the distances between M_i and the planes at each end points of the leg j allows to determine either if:

- M_i is in-between the cylinder caps planes, hence the two legs are on the same level, and their cylinders interfere;
- M_i is outside of the space delimited by the planes in which case we need to check that the angle α between the two legs is perpendicular enough for the legs not to interfere.

From the parametrization figure 10, and considering successively triangles BAC and DAE, we find the two relations (17) on α, from which ensues (18).

$$\cos(\alpha) = \frac{d_{ij} - \epsilon - r_i}{r_j} \quad ; \quad \sin(\alpha) = \frac{d(plane, M_i)}{d_{ij}} \tag{17}$$

$$d_{ij} - \epsilon = r_i + r_j \cos\left(\arcsin\left(\frac{d(plane, M_i)}{d_{ij}}\right)\right) \tag{18}$$

We derive the two conditions (20) from equation (18) and ask for a bisection if neither conditions are met. For $\mathcal{D}_1 = \square d_{ij} \cap [0; r_i + r_j]$, $\mathcal{D}_2 = d(plane, M_i)$, we get the distances $\mathcal{D}_3$ as (19) and the tests as equations (20).

$$\mathcal{D}_3 = r_i + r_j \sqrt{1 - \left(\frac{\mathcal{D}_2}{\mathcal{D}_1}\right)^2} \tag{19}$$

$$\begin{array}{rl} \text{No intersection if:} & \text{Inf}(\mathcal{D}_1) > \text{Sup}(\mathcal{D}_3) \\ \text{Intersection if:} & \text{Sup}(\mathcal{D}_1) \leq \text{Inf}(\mathcal{D}_3) \end{array} \tag{20}$$

The whole algorithm (from 1) to 8) is applied to each distinct pair of legs. Hence, for m the number of legs of the robot, it is run $\begin{pmatrix} m \\ 2 \end{pmatrix} = \frac{m(m-1)}{2}$ times at each box.

IV. Computation Time

Computation times were obtained using the C library 'Time' for a single core of an Intel®Xeon®CPU E5520 at 2.27GHz. The robot considered is a 6-legged suspended CDPR with planar configuration and cubic platform with uncertainties on cable exit points and platform attachment points. The nominal positions of the cable exit points

8) if for any box $\mathcal{B}_k$ in $\mathcal{S}$, d_{ij} evaluate such that $\text{Sup}\,(d_{ij}) \in \,]r_i\,;r_i+r_j]$ and all other boxes $\mathcal{B}_l$ of $\mathcal{S}$ either evaluate to the same case, or satisfy $\text{Inf}\,(d_{ij}\,(\mathcal{B}_l)) > r_i+r_j$ (test of non-interference 7c), then do:

- a) build planes at A_j and B_j, evaluate for boxes M_i in each $\mathcal{B}_k$ of $\mathcal{S}$,
- b) if any contains 0, EXIT, need bisection of pose box.
- c) if the projection of M_i on plane at B_j is negative, and the projection of M_i on plane at A_j is positive, then EXIT, there is a leg/leg interference in this pose box with the current configuration box.
- d) if not, cap situation: for $\mathcal{D}_3$ defined by (19),
  - (d.i) if $\text{Inf}\,(\Box d_{ij}) > \text{Sup}\,(\mathcal{D}_3)$, then EXIT, no interference for this pose box with the current configuration box.
  - (d.ii) if $\text{Sup}\,(\Box d_{ij}) \leq \text{Inf}\,(\mathcal{D}_3)$, then EXIT, there is a leg/leg interference in this pose box with the current configuration box.
  - (d.iii) else, EXIT, need bisection of pose box.

Fig. 13. Last part of the leg/leg interference algorithm, in case of potential crossing of leg's cylinders or one above/below the other.

form a $20m \times 15m$ rectangle, the platform has $1m$ long edges. The application is either a cartesian workspace of $6m \times 1.5m \times 6m \times 17\,\text{deg} \times 22\,\text{deg} \times 12\,\text{deg}$ centred on $(0, -1.25, 21, 7.5, 0, 5)$ for the big box case (BB); or a cartesian workspace of $10mm$ on translations and $2\,\text{deg}$ on rotations centred on $(-5, 5, 15, -10, 20, 0)$ for the small box case (SB). A given obstacle (8.8 MiB STL file comprising 33087 facets) is placed at several positions to get close-to- and collision configurations. The former is achieved by pulling away the obstacle of a collision configuration. Except for the obstacle tree building and overhead times, all other measurements were done on a loop of 10^6 occurrences of the evaluation and the statistics are obtained for 10^3 runs or more. We use the following notations for tables I, II and III:

- OBT: OBstacle Tree computation time. The obstacle collision algorithms use a tree of traits to search for intersection. As the object is fixed, the tree of the obstacle(s) is computed once and for all before the analysis.
- OH: OverHead computation time, mainly the OBT.
- STB: Sheaths Tree Building computation time: the object collision algorithm has the object features tree, but need a leg sheaths tree, which must be rebuilt for all legs, at each pose-box. Also used for LLCS. The times are per leg.
- OC: Object Collision computation time: all five or six tests, for all 6 legs are embedded in this time.
- LLCS: Leg/Leg Collision computation time, Sheaths-based algorithm, for all $6(6-1)/2 = 15$ tests.
- LLCG: Leg/Leg Collision computation time, minimum Gap-based algorithm, for all $6(6-1)/2 = 15$ tests.

Finally, the performance of the collision algorithms is assessed by the rate of undetermined-validity boxes, hence needing a bisection and re-evaluation (the lower the better).

V. Conclusions

While LLCS and OC use leg sheaths and all leg sheaths must be built beforehand, the LLCS is performing well in terms of computation time, under $10ms$, as expected. The

	OBT $[s]$	OH $[s]$	STB $[\mu s]$
range	$[0.33\,;0.49]$	$[0.33\,;0.50]$	$[18.42\,;22.68]$
mean	0.347	0.348	19.792
std dev	0.010	0.010	0.883

Table I. Static computation times. OBT=OBstacle Tree, OH=OverHead, STB=Sheath Tree Build.

	OC $[ms]$	LLCS $[ms]$	LLCG $[ms]$
range	$[0.26\,;90.76]$	$[7.55\,;8.31]$	$[0.94\,;1.01]$
mean	40.00	7.83	0.99
std dev	40.03	0.17	0.003
split rate	14.30%	0%	0%

Table II. Interference algorithm computation times for small boxes. OC=Obstacle Collision, LLCS=Leg/Leg Collision - Swept space based, LLCG=Leg/Leg Collision - min Gap based.

	OC $[ms]$	LLCS $[ms]$	LLCG $[ms]$
range	$[0.02\,;543.96]$	$[0.66\,;9.28]$	$[0.74\,;1.20]$
mean	44.42	4.65	0.96
std dev	110.95	3.55	0.17
split rate	18.96%	50.1%	0%

Table III. Interference algorithm computation times for big boxes.

obstacle collision (OC) scheme suffer from the sheath-based leg/object test in delicate cases. Summed computation times of all tests in the OC save this one range in $[0.02\,;0.18]ms$. The sheath-based tests (LLCS and OC) suffer from the lack of sure-intersection test, especially troublesome with the fail-first strategy common to effective branching algorithms.

The LLCG is surprisingly effective, as it is both box-size independent and very fast: sheath-free hence no additional time, and less than $1.2ms$ even for delicate situations to state on the interference of all legs with each other. Those algorithms were developed for offline analysis of a pair {CDPR configuration, application}, but the figures - consistent over several cases - open the door for real-time applications.

References

[1] Computational Geometry Algorithms Library. http://www.cgal.org.

[2] Boyse John W. Interference detection among solids and surfaces. *Commun. ACM*, 22(1):3–9January 1979.

[3] Irvine Max. *Cable Structures. Cambridge, MA: MIT Press*, 1981.

[4] J.-P. Merlet, D. Danney . *A portable, modular parallel wire crane for rescue operations. 2010 IEEE International Conference on Robotics and Automation (ICRA)*, pages 2834 – 2839May 2010.

[5] Jiménez P., Thomas F., and Torras C. 3d collision detection: a survey. *Computers & Graphics*, 25(2):269 – 285, 2001.

[6] Kearfott R.Baker. *Applications of Interval Computations*, volume 3 of *Applied Optimization*, chapter A Review of Techniques in the Verified Solution of Constrained Global Optimization Problems, pages 23–59. Springer US, 1996.

[7] Landsberger S.E. and Shanmugasundram A.P. Workspace of parallel link crane. In *IMACS/SICE Int. Symp. on Robotics, Mechatronics, and Manufacturing Systems*, pages 479–486, Kobe, September, 16-20, 1992.

[8] Lhomme Olivier. Consistency techniques for numeric csps. pages 232–238, 1993.

[9] Merlet J. P and Daney D. Legs interference checking of parallel robots over a given workspace or trajectory. In *Robotics and Automation, 2006. ICRA 2006. Proceedings 2006 IEEE International Conference on*, pages 757–762May 2006.

[10] Merlet Jean-Pierre. The forward kinematics of cable-driven parallel robots with sagging cables. Personnal communicationSeptember 2014.

[11] Micaël Michelin Baptiste Seguin, Jean-Baptiste Izard. [wp2.1] cable characterization experiments. Research report, Technalia, for Cable-BOT project, 2012.

[12] Van Hentenryck P., McAllester D., and Kapur D. Solving polynomial systems using a branch and prune approach. *SIAM Journal on Numerical Analysis*, 34(2):797–827, 1997.

[13] Wischnitzer Yonatan, Shvalb Nir, and Shoham Moshe. Wire-driven parallel robot: Permitting collisions between wires. *The International Journal of Robotics Research*, 27(9):1007–1026, 2008.