2013 IEEE International Conference on Robotics and Automation (ICRA)
Karlsruhe, Germany, May 6-10, 2013

# Control of a Large Redundantly Actuated Cable-Suspended Parallel Robot

Johann Lamaury and Marc Gouttefarde

*Abstract*— **This paper deals with the control of a 6-DOF cable-suspended parallel robot (CSPR) able to perform tasks such as pick-and-place trajectories over a large workspace. In order to maximize the ratio between the robot workspace and its overall dimensions, actuation redundancy can be used. The control of such a redundantly actuated CSPR turns to be challenging as a realtime embeddable algorithm for distributing the cable tensions should be used, together with a suitable control scheme. This paper proposes a computationally efficient tension distribution algorithm implemented within a dual-space feedforward scheme in order to properly control the moving platform. Experimentations are performed on a large 6-DOF CSPR prototype equipped with 8 actuators.**

## I. INTRODUCTION

Cable-driven parallel robots (CDPR) consist of a mobile platform linked to a fixed base through flexible cables actuated by winches. In order to design an industrial pick-and-place CDPR able to move throughout a plant, a suspended configuration may be adopted (as opposed to a fully-constrained one). Cable-suspended parallel robots (CSPR) are notably suitable for crane-like applications [1], [2] as their cable drawing points are all located above the moving platform. Indeed, the workspace below the platform is then free of cables. This reduces the risks of collisions with the industrial environment, such as machines, goods or workers. In contrast to fully constrained CDPR for which there exists at least one mobile platform pose having force-closure [3], gravity is used to help keeping the cables of CSPR tensed. Furthermore, in order to enlarge the workspace, kinematic redundancy, i.e. the use of more cables than degrees of freedom (DOF), can be used [3], [4]. We illustrated in [5] that using two extra cables allows us to reach large workspaces. These interesting characteristics make redundantly actuated CSPR attractive for large scale pick-and-place tasks.

However, they come with difficult control issues. Cables are flexible links that cannot transmit compressive forces. Their tension vector $\mathbf{t} \in \mathbb{R}^m$, where $m$ is the cable number, is said admissible if all its values remain non-negatives, i.e $0 \leq \mathbf{t}_{min} \leq \mathbf{t}$, and smaller than a maximum value $\mathbf{t} \leq \mathbf{t}_{max}$ which is generally set by the mechanical design. While controlling the mobile platform, if some cables are asked to push, they will slacken and the platform will probably not reach the desired pose. When the number of winches is greater than the number of DOF, an infinite number of cable tension distribution exists for given platform trajectory and external forces. Extensive research on redundancy resolution has been carried out for many different types of robots. In the case of CDPR, fully constrained robots have been mainly considered. Let us note that redundancy resolution turns out to be challenging when the computation has to be done in real time. To the best of our knowledge, the real-time control and cable tension distribution of a 6-DOF redundantly actuated cable-suspended robot has never been reported.

The use of linear programming (LP) has been proposed in [6], [7], [8] in order to have admissible tension distributions with a minimum 1-norm while performing trajectories into the robot workspace. As LP might be subjected to discontinuities, quadratic programming (QP) may be used [9], [10] in order to guarantee continuity. Oh and Agrawal [11] proposed to use either LP and QP in order to minimize the tension vector of a redundant CSPR while keeping it admissible. However, all these methods are using iterative algorithms which may have non-predictable worst-case runtime. Consequently, non-iterative efficient methods have been proposed [12], [13] but, as explained in Section II, they are not suitable for CSPR. Mikelsons *et al.* [14] proposed an algorithm which determines the centroid of the polytope of feasible tension distribution. This algorithm requires the computation of all the polytope vertices and a triangulation phase which makes its direct use in real-time difficult. We thus proposed an efficient way of implementing the centroid approach in [15], as outlined in Section II.

Furthermore, a control scheme able to embed tension distribution algorithms is preferable in order to avoid having a controller output which requires some cables to push on the platform. Two main control scheme frameworks can be distinguished: joint or task space controllers, and non model-based or model-based. In task space coordinates, Gholami *et al.* proposed in [16] a PD controller for tracking a desired trajectory. However, the direct measurement of the platform pose and velocity is difficult [6], [16] and may not be available at all. In joint space coordinates, Kawamura *et al.* employed PD control based on Lyapunov stability analysis and conditions of vector closure [17]. Fang *et al.* used a non-linear feedforward control law based on inverse dynamics. Kino *et al.* applied a PD control in joint space coordinates with an adaptive compensation for a 3-cable fully constrained CDPR in [18]. Let us note that all these previous works deal with fully-constrained CDPR, as opposed to cable-suspended robots. From our experience, to embed a tension distribution algorithm in a joint space controller guaranteeing the non-negativity of the tension vector is not trivial for CSPR. Such

J. Lamaury and M. Gouttefarde are with the Laboratory of Informatics, Robotics and Microelectronics of Montpellier (LIRMM, CNRS-UM2), 161 rue Ada, 34392 Montpellier Cedex 5, France, lamaury@lirmm.fr, gouttefarde@lirmm.fr

978-1-4673-5643-5/13/$31.00 ©2013 IEEE

controllers usually consists of a corrective feedforward term which may not ensure the signs of the cable tensions as it is simply added to the corrector output. An integrated control scheme was simulated in [8] with a PD corrector in both task and joint space, but it still requires direct measurements of the platform pose and velocity. A dual-space feedforward controller has recently been proposed in [19] for rigid-link redundantly actuated parallel manipulators in order to take advantages of both space coordinates. This strategy not only allows the reduction of both the effects of model nonlinearity and the moving platform tracking error, but also enables to easily embed a tension distribution algorithm.

The contribution of this paper is the implementation of a dual-space feedforward controller embedding a non-iterative realtime compatible tension distribution algorithm. It allows a proper control of redundantly actuated CSPR since no cable is asked to push on the mobile platform. This controller is applied to and validated on a 6-DOF CSPR prototype, called CoGiRo, equipped with 8 actuators.

This paper is organized as follows. A description of the proposed tension distribution algorithm is given in Section II. Control schemes and experimental results are respectively reported in Section III and Section IV. Finally, conclusions and future works are addressed in the last section.

## II. TENSION DISTRIBUTION

### A. Proposed strategy

In order to avoid the control law asking to some cables to push on the mobile platform, a non-iterative realtime compatible tension distribution algorithm is proposed. The considered CDPR configuration [5] has 8 cables for 6 DOF. Consequently, with a degree of redundancy (DOR) $r = 2$, the $8 \times 6$ Jacobian matrix $\mathbf{J}$ is non square. The *wrench* matrix $\mathbf{W}$, defined by $\mathbf{W} = -\mathbf{J}^T$, maps the cable tension vector $\mathbf{t} \in \mathbb{R}^8$ to the wrench $\mathbf{f} \in \mathbb{R}^6$ applied by the cables to the mobile platform

$$\mathbf{W}\mathbf{t} = \mathbf{f} \tag{1}$$

which is equivalent to the following equation

$$\mathbf{t} = \mathbf{W}^+\mathbf{f} + \mathbf{N}\boldsymbol{\lambda} = \mathbf{t}_p + \mathbf{t}_n \tag{2}$$

where $\mathbf{W}^+$ is the Moore-Penrose generalized inverse of the wrench matrix, $\mathbf{N} = null(\mathbf{W})$ is a full rank $8 \times 2$ matrix and $\boldsymbol{\lambda} = [\lambda_1 \ \ \lambda_2]^T$ is an arbitrary 2-dimensional vector. The two columns of $\mathbf{N}$ form a 2-dimensional basis of the nullspace of $\mathbf{W}$. $\mathbf{t}_p = \mathbf{W}^+\mathbf{f}$ is the particular minimum-norm solution of (1) and $\mathbf{t}_n = \mathbf{N}\boldsymbol{\lambda}$ is a homogeneous solution that maps vector $\boldsymbol{\lambda}$ into the nullspace of $\mathbf{W}$. Let us define $\Sigma \subset \mathbb{R}^8$ the 2-dimensional affine space of the solutions to (1) and $\Omega \in \mathbb{R}^m$ the 8-dimensional hypercube of feasible cable tensions

$$\begin{aligned} \Sigma &= \{\mathbf{t} \mid \mathbf{W}\mathbf{t} = \mathbf{f}\} \\ \Omega &= \{\mathbf{t} \mid t_i \in [t_{min}, t_{max}], \ 1 \leq i \leq 8\} \end{aligned} \tag{3}$$

The intersection $\Lambda = \Omega \cap \Sigma$ of the affine space $\Sigma$ and the hypercube $\Omega$ is a convex polytope representing the set of *feasible* tension distributions $\mathbf{t}$. Under the affine mapping $(\mathbf{N}, \mathbf{t}_p)$, the preimage of $\Lambda$ is a 2-dimensional convex polytope, i.e. a convex polygon as illustrated in Fig.1. According to (2), this polygon is defined by the following set of $2m = 16$ linear inequalities

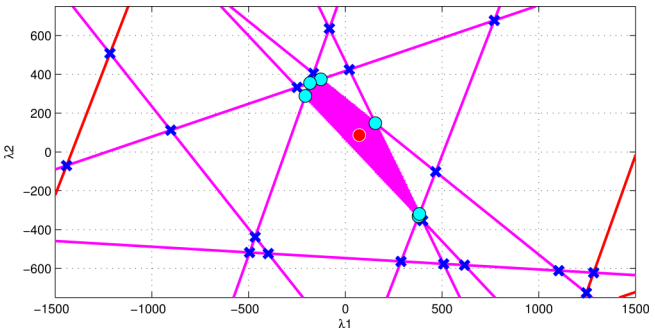


Fig. 1. Preimage of $\Lambda$ in the plane $(\lambda_1, \lambda_2)$ for the CoGiRo CSPR platform pose $[1 \ \ 3 \ \ 2.5 \ \ 15 \ \ 35 \ \ 25]^T$ (units: meters and degrees, XYZ Euler angle convention) with a 40.9 $kg$ platform mass (static equilibrium), $tmin = 10\ N$ and $tmax = 1000\ N$

$$\mathbf{t}_{min} - \mathbf{t}_p \leq \mathbf{N}\boldsymbol{\lambda} \leq \mathbf{t}_{max} - \mathbf{t}_p \tag{4}$$

Here, each inequality defines a halfplane. Intersecting the 2-dimensional set $\Sigma$, they form 16 lines (see Fig.1) corresponding to values of $\boldsymbol{\lambda}$ for which one of the cable tension is equal to $t_{min}$ or to $t_{max}$.

In [12], [13], non-iterative efficient methods have been proposed in order to find a feasible tension distribution for fully constrained CDPR. The basic idea is to project a desired point of the admissible tension set $\Omega$ (e.g. its centroid) onto the set of tension solution $\Sigma$. To be feasible, the projection has to lie into $\Omega$. However, for CSPR, the intersection $\Lambda$ is usually small and this projection is most of the time outside the hypercube of feasible tension. This makes these algorithms inefficient for CSPR.

The strategy proposed by Mikelsons *et al.* [14] consists in computing the centroid of the preimage of $\Lambda$ (polygon) in order to use it as vector $\boldsymbol{\lambda}$ into (2). This results in a "safe" and continuous tension distribution strategy. Safe refers here to the fact that the tension distribution thereby provided is far from the feasible polygon boundaries. However, it requires the computation of all vertices of the preimage of $\Lambda$. This can be done by solving all the $2 \times 2$ subsystems of linear equations obtained by setting two of the inequalities in (4) to equalities. Knowing the vertices, the centroid is then determined by means of a triangulation [14]. These steps are time-consuming and thus hardly suitable for real time control. In the following, we propose to adapt this method for realtime applications by strongly decreasing the computational time. The idea is to find a first vertex of the convex polygon (preimage of $\Lambda$) and then *move* along the boundaries of the polygon until a new vertex is reached. This process continues until all the vertices have been found. Finally, the centroid is calculated by means of well-known formulas, here applicable as our algorithm finds the vertices in order (clockwise or, as in Fig. 1, counter-clockwise).

### B. Realtime compatible algorithm

*1) A first vertex:* The first step of the algorithm consists in finding a first vertex of the preimage of $\Lambda$. Such a vertex is the intersection between two lines $L_{i_b}$ and $L_{j_b}$, $\{i,j\} \in \{1,\dots,8\}$, $i \neq j$. $L_{i_b}$ and $L_{j_b}$ are obtained by taking two inequalities in (4) and replacing the inequality signs by equalities. The letter $b$ refers either to $t_{min}$ or to $t_{max}$ depending on which inequalities of (4) are being considered. $L_{i_b}$ and $L_{j_b}$ are thus defined by the following equations

$$\begin{cases} b_i - t_{p_i} = \mathbf{n}_i \boldsymbol{\lambda}_{ij} \\ b_j - t_{p_j} = \mathbf{n}_j \boldsymbol{\lambda}_{ij} \end{cases} \tag{5}$$

where the two-dimensional line vectors $\mathbf{n}_i$ and $\mathbf{n}_j$ denote the lines $i$ and $j$ of $\mathbf{N}$, respectively. Furthermore, the intersection point $\boldsymbol{\lambda}_{ij}$ between $L_{i_b}$ and $L_{j_b}$ is a vertex $\mathbf{v}_{ij}$ of the preimage of $\Lambda$ only if its image is included in $\Omega$, i.e.

$$\mathbf{t}_{min} - \mathbf{t}_p \leq \mathbf{N}\boldsymbol{\lambda}_{ij} \leq \mathbf{t}_{max} - \mathbf{t}_p \tag{6}$$

Let us denote $\mathbf{v}_{init}$ the first vertex to be found. It must meet both conditions (5) and (6). Consequently, in selecting $t_{min}$ or $t_{max}$ as the values of $b_i$ and of $b_j$ into (5), the algorithm searches for a couple $i,j$ of cables among the $C_8^2 = 28$ possible ones which satisfy the corresponding equalities of (6). Although many couples might be tested, in practice a solution is found very quickly. Indeed, the preimage of $\Lambda$ evolves continuously in time, and $\mathbf{v}_{init}$ can thus be easily obtained from any vertices of the preimage of $\Lambda$ associated with the previous point of the trajectory.

*2) Moving along the polygon boundaries:* The second step consists in moving along one of the two lines $L_{i_b}$ or $L_{j_b}$ intersecting at $\mathbf{v}_{init}$ until a new vertex $\mathbf{v}$ is found. Let us arbitrarily choose $L_{i_b}$. The points $\mathbf{p}$ belonging to this line are given by

$$\mathbf{p} = \mathbf{v}_{init} + \alpha \mathbf{n}_{i_\perp}^T \tag{7}$$

where $\alpha$ is a scalar and $\mathbf{n}_i \cdot \mathbf{n}_{i_\perp}^T = 0$. The next vertex is the point $\mathbf{p}$ in (7) corresponding to the maximal value of $\alpha \geq 0$ such that all the inequalities of (4) are verified. Equivalently, this maximal value is equal to the smallest $\alpha \geq 0$ such that one of the inequalities of (4) apart from inequalities $i$ and $j$ becomes an equality. Therefore, let us consider line $k$ of (4), $k \in \{1,\dots,m\}\backslash\{i,j\}$, and let us substitute the point $\mathbf{p}$ of $L_{i_b}$ given by (7) for $\boldsymbol{\lambda}$, i.e.

$$\begin{cases} t_{min} - t_{p_k} \leq \mathbf{n}_k \mathbf{p} \leq t_{max} - t_{p_k} \\ \mathbf{p} = \mathbf{v}_{init} + \alpha \mathbf{n}_{i_\perp}^T \end{cases} \tag{8}$$

which is equivalent to

$$t_{min} - t_{p_k} \leq \mathbf{n}_k \mathbf{v}_{init} + \alpha \mathbf{n}_k \mathbf{n}_{i_\perp}^T \leq t_{max} - t_{p_k} \tag{9}$$

Let us assume that none of the two lines $L_{k_b}$, the two lines bounding the halfplanes defined by the two inequalities of line $k$ of (4), crosses line $L_{i_b}$ at $\mathbf{v}_{init}$. This amounts to assuming that

$$t_{min} - t_{p_k} < \mathbf{n}_k \mathbf{v}_{init} < t_{max} - t_{p_k} \tag{10}$$

The particular case in which three lines (or more) are crossing at the same point (here lines $L_{i_b}$, $L_{j_b}$ and $L_{k_b}$ crossing at the current vertex) is discussed in [15].

Then, it is not difficult to infer from the sign $s_k = \text{sgn}(\mathbf{n}_k \mathbf{n}_{i_\perp}^T)$ which one of the inequalities in (9) can become an equality for $\alpha \geq 0$. Moreover, since $\mathbf{v}_{init}$ is the intersection of lines $L_{i_b}$ and $L_{j_b}$, the line $L_{k_b}$ must be found for $k$ belonging to the index set $\{1,\dots,m\}\backslash\{i,j\}$ which leaves $2(m-2)$ possibilities (two possible line equations per line of (4)). By computing $s_k$, the previous analysis allows us to reduce this number to $m-2$ possibilities.

In order to find the next vertex of the polygon, i.e., the second polygon vertex belonging to $L_{i_b}$, $(m-2) = 6$ scalars $\alpha_k$ are thus computed and the one, denoted $\alpha_v$, which determines the next vertex is the smallest one, that is

$$\alpha_v = \min(\alpha_k) = \min_{k, \mathbf{n}_k \mathbf{n}_{i_\perp}^T \neq 0} \left(\frac{b_k - t_{p_k} - \mathbf{n}_k \mathbf{v}_{init}}{\mathbf{n}_k \mathbf{n}_{i_\perp}^T}\right) \tag{11}$$

where $b_k = t_{min}$ or $t_{max}$ depending on the sign of $s_k$. The next vertex $\mathbf{v}$ is thus given by

$$\mathbf{v} = \mathbf{v}_{init} + \alpha_v \mathbf{n}_{i_\perp}^T \tag{12}$$

The index $l$ of the line $L_{l_b}$, which crosses $L_{i_b}$ at $\mathbf{v}$ while supporting the polygon along one of its edge, is defined by

$$l = \underset{k, \mathbf{n}_k \mathbf{n}_{i_\perp}^T \neq 0}{\text{argmin}} \left(\frac{b_k - t_{p_k} - \mathbf{n}_k \mathbf{v}_{init}}{\mathbf{n}_k \mathbf{n}_{i_\perp}^T}\right) \tag{13}$$

Note that $L_{l_b}$ is the set of points $\mathbf{p}$ verifying $b_l - t_{p_l} = \mathbf{n}_l \mathbf{p}$ where $b_l = t_{max}$ if $\mathbf{n}_l \mathbf{n}_{i_\perp}^T > 0$ and $b_l = t_{min}$ if $\mathbf{n}_l \mathbf{n}_{i_\perp}^T < 0$.

In this example, the newly found polygon vertex $\mathbf{v}$, is the intersection point between $L_{i_b}$ and $L_{l_b}$. Moving along line $L_{l_b}$ in the appropriate direction, the next polygon vertex is found in exactly the same way as vertex $\mathbf{v}$ has been found. This process continues until the last found vertex lies on the same line as $\mathbf{v}_{init}$ ($L_{j_b}$ in our example), at which point the research stops. Indeed, at this stage, all the vertices of the convex polygon, the preimage of $\Lambda$, have been determined.

*3) The centroid calculation:* The final step is the calculation of the centroid $\mathbf{v}_c$ of the preimage of $\Lambda$ whose $q$ vertices $\mathbf{v}_p = [v_{p_1} \ v_{p_2}]^T$, $p \in \{1,\dots,q\}$ have been determined in steps II-B.1 and II-B.2. Compared to fully constrained CDPR, it can be observed that the area of the preimage of $\Lambda$ in the case of CSPR is considerably smaller. Consequently, for distributing the cable tensions, the choice of the centroid is relevant for CSPR as this point is far from the polygon boundaries, these latter corresponding to lower ($t_{min}$) or upper ($t_{max}$) admissible values of the cable tensions.

The preimage of $\Lambda$ is a convex polygon which is not self-intersecting. Therefore, its centroid $\mathbf{v}_c = [v_{c_1} \ v_{c_2}]^T$ is given by the following well-known formulas

$$\begin{cases} v_{c_1} = \frac{1}{6A} \sum_{p=1}^{q-1} (v_{p_1} + v_{(p+1)_1})(v_{p_1} v_{(p+1)_2} - v_{(p+1)_1} v_{p_2}) \\ v_{c_2} = \frac{1}{6A} \sum_{p=1}^{q-1} (v_{p_2} + v_{(p+1)_2})(v_{p_1} v_{(p+1)_2} - v_{(p+1)_1} v_{p_2}) \end{cases} \tag{14}$$

where $A$ is the area of the polygon given by

$$A = \frac{1}{2}\sum_{p=1}^{q-1}(v_{p_1}v_{(p+1)_2} - v_{(p+1)_1}v_{p_2}) \tag{15}$$

Note that these formulas are applicable since we have found the polygon vertices in order (clockwise or counter-clockwise). Finally, the polygon centroid $\mathbf{v}_c$ is mapped into $\mathbb{R}^m$ in order to find $\mathbf{t}_c = \mathbf{t}_p + \mathbf{N}\mathbf{v}_c$, the centroid of $\Lambda$ which is the desired feasible cable tension distribution.

As an example, Fig. 1 shows $\mathbf{v}_c$ as the (red) dot into the polygon. Crosses correspond to some intersection points between lines bounding the halfplanes defined in (4). The light dots are the vertices of the preimage of $\Lambda$. The average task execution time (TET) of the whole algorithm, including the SVD and nullspace calculation is about $0.87\ ms$ in MATLAB, of which $0.36\ ms$ are dedicated to the vertex search and the centroid calculation. As a comparison, our implementation of the vertex search and the triangulation of [14] take $2.52\ ms$ on average. The proposed algorithm is then about 7 times faster thanks to an important decrease of the number of operations.

A possible implementation into a control scheme is proposed in the next section.

## III. CONTROL

### A. Description of the prototype

The tension distribution method presented in Section II has been implemented on a CSPR prototype shown in Fig. 2. This 6-DOF robot called CoGiRo has the following characteristics:

- $15\ m \times 11\ m \times 6\ m\ (L \times l \times h)$ overall dimensions with a potential workspace of $677\ m^3$;
- cubic mobile platform of side length $1\ m$ in suspended configuration weighting $M = 40,9\ kg$;
- 6-DOF with a Cartesian coordinates vector $\mathbf{x} = \{x, y, z, \phi, \theta, \psi\}$ (XYZ Euler angle convention);
- 8 actuators with joint coordinates (motor angular position) denoted $\mathbf{q} = \{q_1, ..., q_8\}$;
- at least $300\ kg$ payload capability over all the workspace and $500\ kg$ close to its center;
- $\mathbf{t}_{min} = 0\ N$ in order to respect the non-negativity constraint on cable tensions and $\mathbf{t}_{max} = 5000\ N$ according to the maximum capabilities of the force sensors.

The cable length vector $\mathbf{l} \in \mathbb{R}^m$ $(m)$ and the cable linear speed $\dot{\mathbf{l}} \in \mathbb{R}^m$ $(m.s^{-1})$ are indirectly measured by means of incremental encoders. The winches are located on the ground and the cable drawing points near the top of the 6-meter high base frame. The relationship between the *i*th motor angle and the corresponding cable length is written as follows:

$$l_i = \pm r_i q_i \tag{16}$$

where $l_i$ is the unwound length of the *i*th cable, $r_i$ accounts for the transmission reduction ratio $r_{Ri}$ and the drum pitch radius $d_{PRi}$ of the *i*th motor and $q_i$ is the motor angle, with $i \in 1, \ldots, m$. In (16), the effect on the unwound cable length of the cable displacement along the drum has been considered negligible. For this prototype, $r_{Ri} = 3$ and $d_{PRi} = 67.5\ mm$ are identical for all winches. Furthermore, the threaded drum pitch is equal to $5\ mm$ and then

$$r_i = \frac{\sqrt{4\pi^2 d_{PRi}^2 + 0.005^2}}{6\pi} = 0.0225\ m.rd^{-1} \tag{17}$$

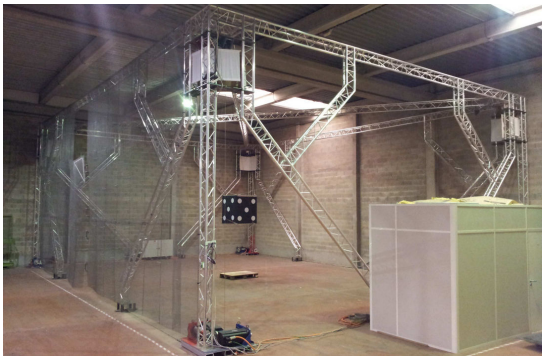

Fig. 2. The LIRMM/Tecnalia COGIRO CSPR prototype

### B. Dynamics

Using Newton-Euler's laws, the general form of the mobile platform equations of motion can be obtained in Cartesian space as:

$$\mathbf{D}(\mathbf{x})\ddot{\mathbf{x}} + \mathbf{C}(\dot{\mathbf{x}}, \mathbf{x})\dot{\mathbf{x}} + \mathbf{g}(\mathbf{x}) = \mathbf{W}\mathbf{t} \tag{18}$$

where

$$\mathbf{g} = \begin{bmatrix} 0 \\ 0 \\ Mg \\ \mathbf{0}_3 \end{bmatrix}; \quad \mathbf{D}(\mathbf{x}) = \begin{bmatrix} M\mathbf{I}_3 & 0 \\ 0 & \mathbf{Q}\mathbf{I}\mathbf{Q}^T\mathbf{S} \end{bmatrix};$$

$$\mathbf{C}(\dot{\mathbf{x}}, \mathbf{x})\dot{\mathbf{x}} = \begin{bmatrix} \mathbf{0}_3 \\ \mathbf{S}\dot{\boldsymbol{\phi}} \times \mathbf{Q}\mathbf{I}\mathbf{Q}^T\mathbf{S}\dot{\boldsymbol{\phi}} \end{bmatrix}$$

with $M$ and $\mathbf{I}$ the moving platform mass and moment of inertia, expressed at the platform center of gravity. $\mathbf{Q}$ is a $3 \times 3$ orthogonal matrix which defines the orientation of the moving platform in the base frame. Let us denote $\mathbf{S}$ the $3 \times 3$ matrix which maps the time derivative of the vector of Euler angles into the mobile platform angular velocity vector $\boldsymbol{\omega}$

$$\boldsymbol{\omega} = \begin{bmatrix} \omega_1 \\ \omega_2 \\ \omega_3 \end{bmatrix} = \begin{bmatrix} 1 & 0 & s\theta \\ 0 & c\phi & -s\phi c\theta \\ 0 & s\phi & c\phi c\theta \end{bmatrix} \begin{bmatrix} \dot{\phi} \\ \dot{\theta} \\ \dot{\psi} \end{bmatrix} = \mathbf{S}\dot{\boldsymbol{\phi}} \tag{19}$$

This dynamic modeling is only valid for $\mathbf{t} \geq 0$ which should be ensured by the algorithm of Section II. Furthermore, the dynamics of the winches is given by

$$\boldsymbol{\Gamma} = \mathbf{I}_m\ddot{\mathbf{q}} + \mathbf{F}_v\dot{\mathbf{q}} + \mathbf{F}_s\,\mathrm{sgn}(\dot{\mathbf{q}}) + \mathbf{R}\mathbf{t} \tag{20}$$

where $\mathbf{I}_m \in \mathbb{R}^{m \times m}$ denotes the inertia of the motors, drums and other rotating parts, $\mathbf{F}_v \in \mathbb{R}^{m \times m}$ and $\mathbf{F}_s \in \mathbb{R}^{m \times m}$ are the diagonal matrices of viscous and dry friction coefficients, respectively, $\mathbf{R} = \mathrm{diag}(r_1, \cdots, r_8) \in \mathbb{R}^{m \times m}$ and $\boldsymbol{\Gamma} \in \mathbb{R}^m$ is the motor torque vector.

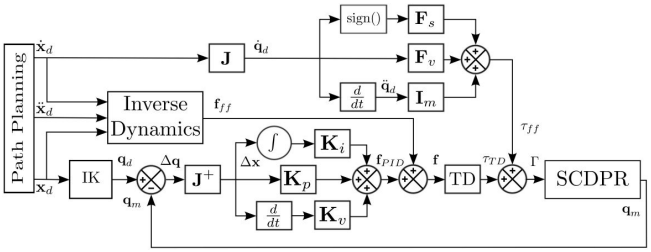


Fig. 3. Dual-space feedforward controller

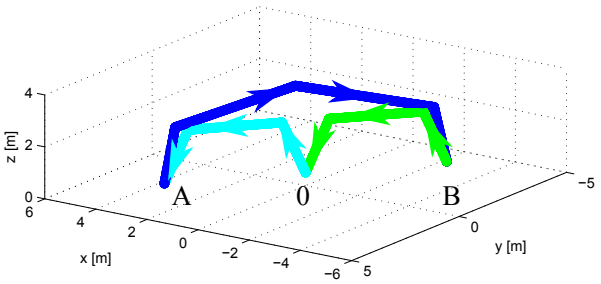


Fig. 4. Trajectory of the present case study

### C. Dual-space feedforward controller

In order to deal with CoGiRo actuation redundancy and dynamic effects in the Cartesian space, the control law (18) may be implemented with a correction in the Cartesian space [19]. However, a direct measurement of the platform pose across a large workspace is difficult. Consequently, we simply use the motor angular positions in the feedback loop. In this section, we propose to apply a dual-space feedforward controller to CSPR as illustrated in the block diagram of Fig.3. Combining (18) and (20) and using the tension distribution algorithm of Section II, the control law is given by

$$\mathbf{\Gamma} = \mathbf{I}_m\ddot{\mathbf{q}} + \mathbf{F}_v\dot{\mathbf{q}} + \mathbf{F}_s\,\mathrm{sgn}(\dot{\mathbf{q}}) + \boldsymbol{\tau}_{TD} \tag{21}$$

where $\boldsymbol{\tau}_{TD}$ is the vector of torques corresponding to the cable tension distribution. $\boldsymbol{\tau}_{TD}$ was obtained from

$$\mathbf{f} = \mathbf{f}_{PID} + \mathbf{D}(\mathbf{x})\ddot{\mathbf{x}} + \mathbf{C}(\dot{\mathbf{x}},\mathbf{x})\dot{\mathbf{x}} + \mathbf{g}(\mathbf{x}) \tag{22}$$

where $\mathbf{f}_{PID} = \mathbf{K}_p\Delta x(t) + \mathbf{K}_i\int\Delta x(t)dt + \mathbf{K}_d\frac{d\Delta x(t)}{dt}$, $\mathbf{K}_p$, $\mathbf{K}_i$, $\mathbf{K}_d$, being diagonal positive-definite $6\times6$ gain matrices. This controller consists in converting the desired platform trajectory $\mathbf{x}_d$ and velocity $\dot{\mathbf{x}}_d$ into their associated joint positions $\mathbf{q}_d$ and velocity $\dot{\mathbf{q}}_d$ using the inverse kinematics (IK). The joint tracking error $\Delta q$ is then converted into $\Delta x$ by means of the pseudo-inverse of the Jacobian matrix in order to apply the PID corrector in the Cartesian space. Indeed, assuming that $\Delta q$ is small ($\Delta t$ being equal to $0.5\ ms$), we can write $\Delta q/\Delta t \approx dq/dt$, and we have $d\mathbf{x}/dt = \mathbf{J}^+(d\mathbf{q}/dt)$. The resulting Cartesian error is then corrected by a PID, the gains of which are $\mathbf{K}_p$, $\mathbf{K}_i$ and $\mathbf{K}_d$. The output $\mathbf{f}_{PID}$ is enhanced with a feedforward term $\mathbf{f}_{ff}$ yielded by the inverse dynamics (18). $\mathbf{f} = \mathbf{f}_{PID} + \mathbf{f}_{ff}$ is then used as an input to the tension distribution algorithm of Section II ("TD" block in Fig. 3). Finally, the motor dynamics is compensated using a feedforward torque $\boldsymbol{\tau}_{ff}$ which is added to $\boldsymbol{\tau}_{TD}$ to yield the motor torque vector $\mathbf{\Gamma}$. The different gains were tuned experimentally to reach good performances. The PID was tuned specifically for each of the three Cartesian axis both for translations and rotations making this operation more complex than for a PID in joint space.

## IV. EXPERIMENTAL RESULTS

In this section, experimental results from the implementation of the control scheme of Section III into the CoGiRo prototype control system are briefly discussed. Our experimental testbed is composed of a host computer and an industrial embedded computer which performs the real time tasks at a $2\ kHz$ sampling frequency. The controller of Fig. 3 is coded in ANSI C. For the interest of the experiment, the controller was also implemented in Matlab code on another (smaller) testbed using xPc target at the same sampling frequency.

### A. Scenario

The overall objective of this study is to evaluate the capability of the proposed controller to move the platform while keeping it stable along the trajectory. The desired trajectory given in Table I is chosen such that the moving platform reaches some poses quite close to the workspace corners. At such locations, special care should be taken to avoid a controller asking the cables to push on the mobile platform. Fig. 4 illustrates the sequence of movements to be executed by the moving platform in the present case study. In Fig. 4, both points $A$ and $B$ correspond to potential pick-and-place locations. The given poses are stated in task space coordinates, with $[m]$ and $[deg]$ units. A video, provided with this paper, shows the CoGiRo moving platform motions using the trajectory described here as an input to the controller.

| Points | Coordinates | Points | Coordinates |
|---|---|---|---|
| 1 | [0;0;0;0;0;0] | 6 | [2.5;-2.5;2;0;0;0] |
| 2 | [0.5;0.5;2;0;0;0] | 7 | [2.5;-3;2;0;0;0] |
| 3 | [2.5;2.5;2;0;0;0] | 8 | [-3;-3;0;0;0;0] |
| 4 | [3;3;0;0;0;0] | 9 | [-2.5;-2.5;2;0;0;0] |
| 5 | [3;2.5;2;0;0;0] | 10 | [-0.5;-0.5;2;0;0;0] |

TABLE I

FIRST, INTERMEDIATE AND FINAL POINTS OF THE TRAJECTORY

This trajectory is relevant in our case study as all its poses are contained into the CoGiRo static workspace. But, in order to reach all of them, the controller should exclusively yield non-negative cable tension instructions.

### B. Results

Preliminary simulation results of the tension distribution are shown in Fig. 5 along the proposed trajectory. The tension distribution algorithm returns strictly non-negative cable tension distributions. Fig. 6 shows the outputs of the control law, converted motor torques to cable tensions.

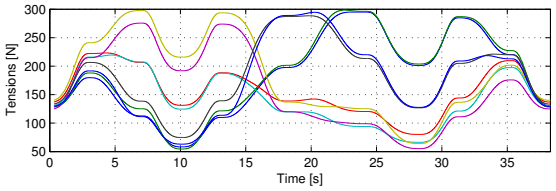


Fig. 5. Control law in tension [N] of the proposed control scheme obtained in simulation

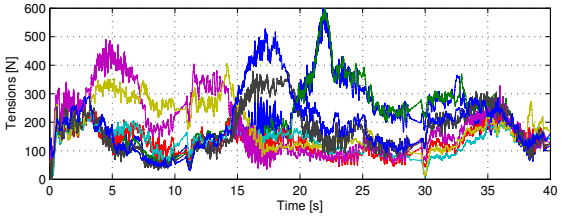


Fig. 6. Control law in tension [N] of the proposed control scheme - Experimental results (CoGiRo prototype

These experimental results show that the proposed controller provides a non-negative and continuous tension distribution along the whole trajectory. These experiments validate the use of our tension distribution algorithm. Furthermore, the algorithm proved to be real-time compatible with an average TET of about $0.25\ ms$. It is important to note that, during our experimentations, it was observed that neglecting dry friction in the winches leads to a large tracking error. Consequently, large forces are generated by the *PID* corrector which may be impossible to balance ($\Lambda = \Omega \cap \Sigma = void$) so that the tension distribution algorithm fails and the control law may ask some cables to push on the moving platform. For the same reasons, once converted into platform wrenches, the joint space feedforward torque $\boldsymbol{\tau}_{ff}$ cannot be included in the input to the tension distribution block ("TD" in Fig. 3).

Finally, large dry friction torques were identified in the powerful winches actuating the CoGiRo prototype. Dry friction is thus the most influencing parameter among the joint space feedforward terms. Besides, the most influencing parameter of the Cartesian space feedforward is the weight of the platform.

## V. CONCLUSIONS AND FUTURE WORKS

In this paper, a dual-space feedforward control scheme implementing a real-time tension distribution algorithm were presented and applied to redundantly actuated cable-suspended parallel robots. Experimental results were also reported. The resulting controller provides good tracking performances and ensures that none of cables is required to push on the mobile platform. However, the Cartesian controller gains are difficult to tune and part of our future works will be dedicated to a dual-space control scheme with a joint space PID.

Large-scale CSPR have a large workspace and may have a wide range of possible payload weights. An issue is then the variability in the moving platform locations and in the payload weights which should be reflected in the control parameters. Consequently, in order to obtain better performances over the whole workspace and for a range of possible payloads, an augmented adaptive control scheme will be developed as part of our future works.

## VI. ACKNOWLEDGMENTS

The financial support of the ANR (grant 2009 SEGI 018 01) and of the Région Languedoc-Roussillon (grants 115217 and 120218) are greatly acknowledged.

## REFERENCES

[1] J. Albus, R. Bostelman, and N. Dagalakis, "The NIST Robocrane," *Journal of Robotic Systems*, vol. 10, no. 2, pp. 709–724, 1993.

[2] W.-J. Shiang, D. Cannon, and J. Gorman, "Dynamic Analysis of Cable Array Robotic Crane," in *IEEE Int. Conf. on Robotics and Automation*, 1999.

[3] S. Kawamura and K. Ito, "A New Type of Master Robot for Teleoperation Usign a Radial Wire Drive System," in *IEEE/RSJ Int. Conf. on Intelligent Robots and Systems*, 1993.

[4] M. Hiller, S. Fang, S. Mielczarek, R. Verhoeven, and D. Franitza, "Design, analysis and realization of tendon-based parallel manipulators," *Mechanism and Machine Theory*, vol. 40, pp. 429–445, 2004.

[5] J. Lamaury, M. Gouttefarde, and O. Tempier, "Design and Control Strategies of a Redundant Suspended Cable-Driven Parallel Robots," in *Advances in Robot Kinematics*, pp. 1–9, 2012.

[6] S. Fang, D. Franitza, M. Torlo, F. Bekes, and M. Hiller, "Motion Control of a Tendon-Based Parallel Manipulator Using Optimal Tension Distribution," *IEEE/ASME Trans. on Mechatronics*, vol. 9, pp. 561–568, 2004.

[7] P. H. Borgstrom, B. L. Jordan, G. S. Sukhatme, M. A. Batalin, and W. J. Kaiser, "Rapid Computation of Optimally Safe Tension Distributions for Parallel Cable-Driven Robots," *IEEE Trans. on Robotics*, vol. 25, pp. 1271–1281, Dec. 2009.

[8] A. Vafaei, M. M. Aref, and H. D. Taghirad, "Integrated Controller For An Over Constrained Cable Driven Parallel Manipulator : KNTU CDRPM," in *IEEE Int. Conf. on Robotics and Automation*, 2010.

[9] T. Bruckmann, A. Pott, M. Hiller, and D. Franitza, "A Modular Controller for Redundantly Actuated Tendon-Based Stewart Platforms," in *EuCoMes, The First Conference on Mechanism Science*, 2006.

[10] M. Agahi and L. Notash, "Redundancy Resolution of Wire-Actuated Parallel Manipulators," *special edition of the Trans. Can. Soc. Mech. Eng.*, vol. 33, no. 4, pp. 561–573, 2009.

[11] S.-R. Oh and S. K. Agrawal, "Cable Suspended Planar Robots With Redundant Cables : Controllers with Positive Tensions," *IEEE Trans. on Robotics*, vol. 21, no. 3, pp. 457–465, 2005.

[12] A. Pott, T. Bruckmann, and L. Mikelsons, "Closed-form Force Distribution for Parallel Wire Robots," in *Computational Kinematics*, pp. 1–10, 2009.

[13] P. Lafourcade, *Etude des manipulateurs parallèles à câbles, conception dune suspension active pour soufflerie*. PhD thesis, Thèse de Docteur Ingénieur de l'ENSAE, 2004.

[14] L. Mikelsons, T. Bruckmann, M. Hiller, and D. Schramm, "A Real-Time Capable Force Calculation Algorithm for Redundant Tendon-Based Parallel Manipulators," in *IEEE Int. Conf. on Robotics and Automation*, May 2008.

[15] J. Lamaury and M. Gouttefarde, "A Tension Distribution Method with Improved Computational Efficiency," in *First Int. Conf. on Cable-Driven Parallel Robots* (T. Bruckmann and A. Pott, eds.), pp. 71–85, Springer, 2012.

[16] P. Gholami, M. Aref, and H. D. Taghirad, "On The Control of the KNTU CDRPM: A Cable Driven Redundant Parallel Manipulator," in *IEEE 2008 International Conference on Intelligent Robots and Systems*, 2008.

[17] S. Kawamura, H. Kino, and C. Won, "High-speed manipulation by using parallel wire-driven robots," *Robotica*, vol. 18, pp. 13–21, 2000.

[18] H. Kino, T. Yahiro, T. Fumiaki, and T. Morizono, "Robust PD Control Using Adaptative Compenstaion for Completely Restrained Parallel-Wire Driven Robots," in *IEEE Transactions on Robotics*, vol. 23, pp. 803–812, 2007.

[19] G. S. Natal, A. Chemori, and F. Pierrot, "Dual-Space Adaptive Control of Redundantly Actuated Parallel Manipulators for Extremely Fast Operations With Load Changes," in *IEEE 2012 Int. Conf. on Robotics and Automation*.