Efficient Computation of the extremum of the Articular Velocities of a Parallel Manipulator in a Translation Workspace.

Jean-Pierre MERLET
INRIA Sophia-Antipolis
BP 93, 06902 Sophia-Antipolis, France
E-mail: Jean-Pierre.Merlet@sophia.inria.fr

Abstract

This paper presents an efficient algorithm for computing, with a guaranteed error, the maximal and minimal articular velocities of a parallel manipulator so that whatever is the location of the end-effector in a given volume it may perform a motion at a given cartesian/angular velocity, under the assumption that its orientation is kept constant over the volume. This algorithm is much more faster and safe than the classical discretisation method.

1 Introduction

The design of a parallel manipulators involves various objectives which are either to be optimized or to be reached: positioning workspace, positioning accuracy, maximal articular forces over a given workspace etc.. One of these objectives may be the ability to perform a given cartesian/angular velocity of the end-effector for any of its position in a given workspace. It is then necessary to determine the maximal articular velocities of the robot for reaching this objective. Numerous papers have been devoted to the relations between articular velocities and cartesian/angular velocity of the end-effector [1],[3],[4], [7],[8] but none, to the best of the author knowledge, have addressed the problem of determining the extremum of the articular velocities when the end-effector is moving inside a given workspace.

In this paper we will consider the classical Gough type parallel manipulator [2] illustrated in figure 1. In this robot a base and a platform are connected through 6 extensible legs with ball-and-socket joints at each extremity.

We assume that the user is interested in the maximal and minimal velocities of the linear actuators needed to insure a given cartesian/angular velocity $\boldsymbol{\mathcal{V}}$ of the end-effector, whatever the posture of the mobile platform in a given workspace is. The articular velocity vector $\dot{\boldsymbol{\rho}}$ is related to the cartesian and angular velocity vector $\boldsymbol{\mathcal{V}}$ of the end-effector by:

$$\dot{\boldsymbol{\rho}} = J^{-1}(X)\boldsymbol{\mathcal{V}} \quad (1)$$

where X is the posture of the robot and J^{-1} is its inverse jacobian matrix. For a given cartesian/angular velocity $\boldsymbol{\mathcal{V}}$ the articular velocities are therefore posture dependent. The purpose of this paper is to compute the maximal and minimal values of the components of the vector $\dot{\boldsymbol{\rho}}$ for a given velocity vector $\boldsymbol{\mathcal{V}}$ and for any posture X inside a given domain.

In the sequel C will denote the center of the moving platform, O the origin of the reference frame, A_i, B_i respectively the base and mobile joint centers of leg i. The vector $\boldsymbol{\mathcal{V}}$ can be decomposed into a cartesian ve-

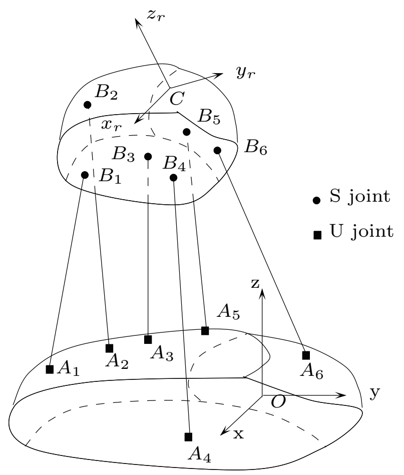

Figure 1: The classical Gough type parallel robot

locity vector $\boldsymbol{V}$ and an angular velocity vector $\boldsymbol{\Omega}$. We will assume that the orientation of the moving platform is constant (hence the vector $\mathbf{CB_i}$ is constant).

2 Extremal velocities for a segment

Let us assume that the point C is moving along a segment defined by two points M_1, M_2. Any position of C on the segment may be written as:

$$\boldsymbol{OC} = \boldsymbol{OM_1} + \lambda \boldsymbol{M_1M_2} \tag{2}$$

where λ is a scalar in the range [0,1]. For a given velocity vector $\boldsymbol{\mathcal{V}}$ the velocity of leg i is written as:

$$\dot{\rho}_i = J_i^{-1}\boldsymbol{\mathcal{V}}$$

where J_i^{-1} is the ith row of the inverse jacobian matrix. It is well known that this row is:

$$\frac{\boldsymbol{A_iB_i}}{||\boldsymbol{A_iB_i}||} \quad , \ \boldsymbol{CB_i} \times \frac{\boldsymbol{A_iB_i}}{||\boldsymbol{A_iB_i}||}$$

We notice immediately that for a given leg the articular velocity is not dependent upon the other leg velocities. Consequently we will drop the subscript in the sequel. We have:

$$\dot{\rho} = \frac{\boldsymbol{AB}.\boldsymbol{V} + (\boldsymbol{CB} \times \boldsymbol{AB}).\boldsymbol{\Omega}}{||\boldsymbol{AB}||} \tag{3}$$

Note that $\boldsymbol{AB}$ may be written as:

$$\boldsymbol{AB} = \boldsymbol{AO}+\boldsymbol{OC}+\boldsymbol{CB} = \boldsymbol{AO}+\boldsymbol{OM_1}+\boldsymbol{CB}+\lambda\boldsymbol{M_1M_2}$$

In this equation the three first vectors are constant. Consequently we may write it in a simpler form as:

$$\boldsymbol{AB} = \boldsymbol{U} + \lambda \boldsymbol{M_1M_2}$$

The norm of this vector is therefore:

$$||\boldsymbol{AB}|| = \sqrt{\lambda^2||\boldsymbol{M_1M_2}||^2 + 2\lambda\boldsymbol{U}.\boldsymbol{M_1M_2} + ||\boldsymbol{U}||^2}$$

Similarly we have:

$$\boldsymbol{CB} \times \boldsymbol{AB} = \boldsymbol{CB} \times \boldsymbol{U} + \lambda\boldsymbol{CB} \times \boldsymbol{M_1M_2}$$

Using the previous equations (3) can be written in a simplified form:

$$\dot{\rho} = \frac{a_1\lambda + a_2}{\sqrt{a_3\lambda^2 + a_4\lambda + a_5}} \tag{4}$$

with

$$\begin{aligned} a_1 &= \boldsymbol{M_1M_2}.\boldsymbol{V} + (\boldsymbol{CB} \times \boldsymbol{M_1M_2}).\boldsymbol{\Omega} \\ a_2 &= \boldsymbol{U}.\boldsymbol{V} + (\boldsymbol{CB} \times \boldsymbol{U}).\boldsymbol{\Omega} \quad a_3 = ||\boldsymbol{M_1M_2}||^2 \\ a_4 &= 2\,\boldsymbol{U}.\boldsymbol{M_1M_2} \quad a_5 = ||\boldsymbol{U}||^2 \end{aligned}$$

The a_i are therefore constants which depend only upon the geometry of the robot, the imposed cartesian velocity and the segment extremities. Differentiating this expression with respect to λ leads to an expression whose numerator N is simplified to a linear function in λ. Consequently the minimal and maximal values of the articular velocity is obtained either for $\lambda = 0$ or $\lambda = 1$ or for the value which nullify N (if this value lie inside the range [0,1]).

3 Extremal velocities for a rectangle

Let us assume now that the workspace is defined by all the points inside a rectangle. Without loss of generality we may assume that any point in the rectangle is such that its coordinates verify

$$x_1 \leq x \leq x_2 \qquad y_1 \leq y \leq y_2$$

Consequently we want to calculate the extremal values of $\dot{\rho}$ for any position of C verifying the previous constraints. This can be done using classical optimization techniques. We define two new variables α, β by:

$$\begin{aligned} x &= x_1 + \frac{(1+\sin\alpha)(x_2 - x_1)}{2} \\ y &= y_1 + \frac{(1+\sin\beta)(y_2 - y_1)}{2} \end{aligned} \tag{5}$$

$\dot{\rho}$ is now a function of α, β whose extremal values satisfy necessarily the following equations:

$$\frac{\partial\dot{\rho}}{\partial\alpha} = 0 \qquad \frac{\partial\dot{\rho}}{\partial\beta} = 0 \tag{6}$$

It appears that equations (6) may be written as:

$$\cos(\alpha)\ F_1(\alpha,\beta) = 0 \qquad \cos(\beta)\ F_2(\alpha,\beta) = 0 \tag{7}$$

The solutions defined by $\alpha = \pm\frac{\pi}{2}$ and $\beta = \pm\frac{\pi}{2}$ correspond to an extremum on the edges of the rectangle, which can be computed using the previous section. The last remaining solutions are obtained when:

$$F_1(\alpha,\beta) = 0 \quad F_2(\alpha,\beta) = 0$$

which correspond to an extremum for a point inside the rectangle. In these equations the unknowns α, β appear via their sine only. F_1 is linear in $\sin\alpha$ and is solved for this unknown. The result is substituted into F_2 which become a third order polynomial in $\sin\beta$ only. By solving this equation the last remaining set of solutions of equations (6) are determined, which give the extremum inside the rectangle.. In summary the extremum on the edges are computed using the method described in the previous section and are compared to the extremum found for the inside of the rectangle to lead to the extremum for the whole rectangle.

4 Extremal velocities for a box

Let us assume now that the workspace is defined to be all the points inside a rectangular box. Without loss of generality we may assume that any point in the box is such that its coordinates verify

$$x_1 \leq x \leq x_2 \quad y_1 \leq y \leq y_2 \quad z_1 \leq z \leq z_2 \qquad (8)$$

Consequently we want to calculate the extremal values of $\dot{\rho}$ for any position of C verifying (8).

4.1 The optimization approach

We define three new variables α, β, γ such that:

$$\begin{aligned} x &= x_1 + \frac{1+\sin\alpha(x_2-x_1)}{2} \\ y &= y_1 + \frac{(1+\sin\beta(y_2-y_1)}{2} \\ z &= z_1 + \frac{1+\sin\gamma(z_2-z_1)}{2} \end{aligned} \qquad (9)$$

$\dot{\rho}$ is now a function of α, β, γ whose extremal values satisfy necessarily the following equations:

$$\frac{\partial\dot{\rho}}{\partial\alpha} = 0 \quad \frac{\partial\dot{\rho}}{\partial\beta} = 0 \quad \frac{\partial\dot{\rho}}{\partial\gamma} = 0$$

These three equations are transformed into algebraic equations using the classical half-angle tangent substitution. The resultants of two different pairs of equations lead to the two following equations :

$$\sum_{i,j=0,4} a_{ij} T_1^i T_2^j = 0 \qquad \sum_{i,j=0,4} b_{ij} T_1^i T_2^j = 0$$

with $T_1 = tan(\alpha/2), T_2 = tan(\beta/2)$. Unfortunately we have not been able to solve this system.

At this point various options are possible: we may use a numerical method like continuation or intervals computing to solve numerically this set of equations. In our implementation we have chosen an alternative approach proposed in the next section.

4.2 An alternative approach

4.2.1 Principle

We intend to determine the maximal articular velocities in a box with a guaranteed error $\epsilon > 0$. The idea is to sweep the box by horizontal rectangles at various heights z, the difference of height between two successive rectangles being such that the difference of extremal articular velocities between the two rectangle does not exceed ϵ. Thus starting with the rectangle with the lowest z of the box, we will then determine the next z satisfying the previous constraint. The process will be repeated for the new rectangle until the height of the rectangle is greater or equal to the maximal z of the box. At this point the maximal velocity will have been determined with an error of at most ϵ.

4.3 Finding the increase of height

Let us assume that at the k^{th} step of the algorithm the maximal velocity is $\dot{\rho}_k$ for the plane at height z_k. We want to determine the minimal increase of height z_Δ^2 such that

$$\dot{\rho}(z_k + z_\Delta^2) = \dot{\rho}_k + \epsilon \qquad (10)$$

This may be seen a a classical optimization problem but the previous formulation is difficult to use in practice as $\dot{\rho}$ is a complex expression. But we have:

$$\dot{\rho} = \frac{F(z, \boldsymbol{\mathcal{V}})}{\rho(z)}$$

F and ρ^2 being algebraic functions of z. Thus equation (10) is transformed into:

$$F^2(z_k + z_\Delta^2) - (\dot{\rho}_k + \epsilon)^2 \rho^2(z_k + z_\Delta^2) = 0 \qquad (11)$$

which has now an algebraic form. Now we have to solve the optimization problem of finding the minimal z_Δ^2 such that equation (11) is satisfied with the constraints on x, y defined in equations (9). We define the optimization function H as:

$$H = z_\Delta^2 + \mu(F^2(z_k + z_\Delta^2) - (\dot{\rho}_k + \epsilon)^2 \rho^2(z_k + z_\Delta^2))$$

in which the value of x, y have been substituted by equations (9). The minimum of z_Δ will be obtained by solving the system of equations:

$$\frac{\partial H}{\partial z_\Delta} = 0 \; \frac{\partial H}{\partial \mu} = 0 \; \frac{\partial H}{\partial \alpha} = 0 \; \frac{\partial H}{\partial \beta} = 0$$

For each solution of this system it is necessary to check that the value of z_Δ does not lead to $\dot{\rho} = -\dot{\rho}_k - \epsilon$ which is also solution to equation (10). Due to the lack of space we will not present the details of the calculation but this problem is solved by manipulating a few sets of algebraic equations.

As for the determination of the minimal velocities a similar method is used with $\epsilon < 0$.

4.3.1 Computation time

Clearly the computation time of the previous algorithm is dependent upon the precision ϵ with which the extremal velocities are to be determined. Figure 2 presents the computation time as a function of the error ϵ, obtained with a SUN Ultra 1 workstation. It

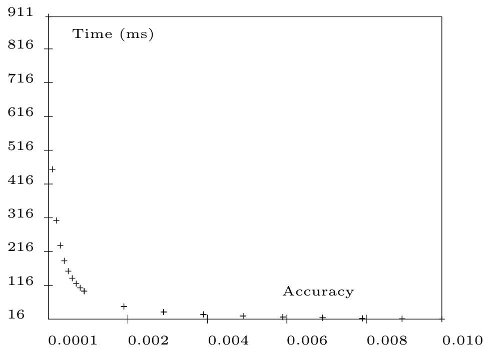

Figure 2: Computation time as a function of the precision on the value of the extremal velocities

may be seen that even with a high precision on the velocities the computation time is quite reasonable. Note that even with an accuracy of 0.01 the computation time of our algorithm is equal to the time needed to compute the articular velocities at less than 400 points in the box i.e. less than 8 points for each dimension: such limited number of points will never insure a correct estimation of the extremal velocities.

5 Maximal velocities in any volume

This method can be extended to more complex workspaces than a box. We will assume here that a workspace is described by a set of polygonal cross-sections. We will compute the extremal velocities for each volume defined between two successive cross-sections, then it will be easy to determine the extremal velocities for the whole workspace. A given volume will be decomposed into as many boxes as necessary until the velocities are determined with the desired accuracy. A list of box $\mathcal{B}$ is maintained during the algorithm: this list is initialized with the bounding box B_0 of the whole volume. Another list $\mathcal{L}$ will contain the current extremum of the articular velocities: this list is initialized by the extremum of the articular velocities computed over the vertices of the workspace boundary. At step k the algorithm perform the following operation:

1. if the box B_k is completely outside the volume we consider the next box in the list
2. if the box B_k is completely inside the volume we compute the extremal articular velocities for this box and update $\mathcal{L}$.
3. if the box B_k is partially inside the volume we compute the extremal articular velocities for this box.
   (a) if these extremum are within the range of $\mathcal{L}$ or the differences between the extremum are lower than ϵ we consider the next box in the list
   (b) otherwise the box is split into eight boxes by dividing each dimension of the box by 2. The resulting boxes are put at the end of the list and we consider the next box in the list.

The algorithm stop if there is no more box in the list. It enables to compute the extremal velocities for any type of workspace in a reasonable amount of time. For example we have considered the workspace defined by three square cross-sections: $z = 50, x \in [-10, 10], y \in [-10, 10]$, $z = 55, x \in [-5, 5], y \in [-5, 5]$, $z = 60, x \in [-10, 10], y \in [-10, 10]$ represented in figure 3.

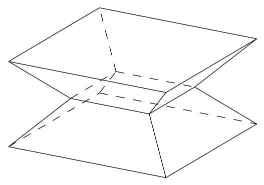

Figure 3: The test volume

The orientation is defined by the three Euler angles with value -60 degree, the desired cartesian/angular velocity is : -10, 0, -10, -10, 0 10. Table 1 indicates the computation time for various values of ϵ. We may

ϵ	1	0.5	0.1	0.05	0.01
Time (ms)	360	430	4260	12220	95500

Table 1: Computation time for a complex workspace as function of the desired accuracy ϵ on the articular velocities

note that the computation time for determining the velocities with an accuracy better than 1% is approximatively 4s.

6 Special case

In this special case we will assume that the angular velocity is equal to 0. Consequently the articular velocity is given by:

$$\dot{\rho} = \boldsymbol{V}.\frac{\boldsymbol{AB}}{||\boldsymbol{AB}||} \tag{12}$$

Let D_1 be the line going through A with vector $\boldsymbol{V}$ and D_2 be the line associated to the leg. If μ is the angle between these two lines we have:

$$\dot{\rho} = ||\boldsymbol{V}|| \cos \mu \tag{13}$$

Hence if the line D_1 crosses the volume at a point M, then the maximal articular velocity will be $||\boldsymbol{V}||$ if $\boldsymbol{AM}.\boldsymbol{V}$ is positive or the minimal articular velocity will be $-||\boldsymbol{V}||$ if $\boldsymbol{AM}.\boldsymbol{V}$ is negative. In the sequel we will assume that $\boldsymbol{AM}.\boldsymbol{V}$ is positive (if this quantity is negative, then maximal has to be changed to minimal). If D_1 does not cross the volume the maximal velocity will be obtained for a location of C on one edge of the volume. Similarly the minimal articular velocity is also obtained for a location of C on one edge of the volume. Therefore for computing the maximal articular velocity we have to check if D_1 crosses the volume. We will consider here without loss of generality the volume between two cross-sections. D_1 may cross this volume either on a horizontal facet or on a side facet. An horizontal facet is defined by a polygon and an altitude z_i. For the two horizontal cross sections we compute the intersection of the line with the plane $z = z_i$ and check if the intersection point belongs to the polygon, in which case the maximal articular velocity is $||\boldsymbol{V}||$.

A side facet is defined by two horizontal segments: the coordinates of the extremities of the first segment will be denoted $(x_1, y_1, z_1), (x_2, y_2, z_1)$ and $(x_3, y_3, z_3), (x_4, y_4, z_3)$ will denote the coordinates of the extremities of the second segment.

At a given altitude z the side facet is constituted of a segment whose start and end point X_k, X_{k+1} have the coordinates:

$$\begin{aligned}
x_k &= x_1 + (x_3 - x_1)(z - z_1)(z_3 - z_1) && (14)\\
y_k &= y_1 + (y_3 - y_1)(z - z_1)(z_3 - z_1) && (15)\\
z_{k+1} &= z_k = z && (16)\\
x_{k+1} &= x_2 + (x_4 - x_2)(z - z_1)(z_3 - z_1) && (17)\\
y_{k+1} &= y_2 + (y_4 - y_2)(z - z_1)(z_3 - z_1) && (18)
\end{aligned}$$

A point $M(x, y, z)$ which belongs to this segment has the coordinates:

$$\begin{aligned}
x &= x_k + \lambda(x_{k+1} - x_k) && (19)\\
y &= y_k + \lambda(y_{k+1} - y_k) && (20)
\end{aligned}$$

with λ in the range [0,1]. If this point belongs to D_1 then:

$$\boldsymbol{AM} \times \boldsymbol{V} = \boldsymbol{0} \tag{21}$$

This relation leads to 2 equations in the unknowns z, λ. If the vertical component of the velocity is equal to 0 then the z coordinate is the z coordinates of the point A and λ is obtained linearly from equation (21): if λ lie in the range [0,1], then we have intersection between D_1 and the volume.

If the vertical component of the velocity is not equal to 0 the resultant of two equations of (21) leads to a second order equation in z. This equation is solved and if it has a solution in the range $[z_1, z_2]$ the value of λ is computed and if λ lie in the range [0,1], then we have intersection between D_1 and the volume. With this method we are able to check if D_1 crosses the volume. If this is not the case and in order to compute the minimal articular velocities we have now to compute the extremal velocities for each edge of the volume. This is done by using the algorithm described in section 2.

Clearly this algorithm is very fast (less than 13 ms for the example presented above) and leads to the exact determination of the extremal articular velocities. Note that the algorithm can be trivially extended to workspace volumes described by spheres.

7 Articular workspace

Let us assume that the leg length have a minimal and a maximal values ρ_{min}, ρ_{max}. It may be of interest to compute the extremum of the articular velocities in the workspace defined by a constant orientation and any position of the platform which fulfill the constraints on the leg lengths: this workspace will be called the *articular workspace*. A simple adaptation of the previous algorithm enable to perform this task. Note first that a trivial algorithm enable to determine what are the extremum of each leg lengths while C moves in a given box: we will denote this algorithm $Max_\rho(B)$ where B is a box. Then notice that it is easy to determine a box which contain all the possible locations of the platform being given the extremal values of the leg lengths. We start the previous algorithm with this box. Then we have to change the inclusion test in the previous algorithm: a box will lie within the workspace if all the ranges given by $Max_\rho(B)$ lie within $[\rho_{min}, \rho_{max}]$ while a box will be completely outside the workspace if one of the ranges is outside the range $[\rho_{min}, \rho_{max}]$. In any other cases the box we assume that the box is partially within the workspace.

Strictly speaking this may be false: as $Max_\rho(B)$ gives the extremum of the leg lengths *independently* it may occur that there is no posture of the platform where *all* the leg lengths lie in the correct range at the same time, but as this type of box will be divided in smaller box during the process our assumption lie on the safe side.

Note also that another approach will be to use the algorithm described in [5] which enable to compute exact cross-sections of the workspace for a constant orientation and then use the algorithm described for the polyhedric workspace.

For an accuracy of 0.1 (0.01) the computation time is 29800 ms (67650 ms) and it is reduced to 1270 ms (1380 ms) if the angular velocities are equal to 0.

8 Another utility of the algorithm

Let us assume that the angular velocities are set to 0 and that the cartesian velocity is defined as a unit vector $\boldsymbol{V}$. The algorithm will therefore compute the minimum and maximum of the quantity $\boldsymbol{AB}.\boldsymbol{V}/||\boldsymbol{AB}||$ which is the cosine of the angle between the link direction and the vector $\boldsymbol{V}$. Consequently we will get the minimum and maximum values of the angle of the passive joints with any fixed direction.

9 Extension to other types of parallel robots

The algorithm has been presented for the Gough-type parallel robot but may be extended for other types of parallel robots. Indeed it is well known that most of parallel robots have an inverse jacobian matrix of the same form as the one of the Gough-type robot. Therefore the principle of the algorithm will be similar. Consider for example the parallel robots with fixed leg lengths but whose A_i points moves on a line with unit vector $\boldsymbol{u_i}$. The velocity $\dot{\gamma_i}$ of point A_i is related to the cartesian and angular velocities by [5]:

$$\dot{\gamma_i} = \frac{\boldsymbol{A_iB_i}.\boldsymbol{V}}{\boldsymbol{u_i}.\boldsymbol{A_iB_i}} + \frac{\boldsymbol{A_iB_i} \times \boldsymbol{\Omega}}{\boldsymbol{u_i}.\boldsymbol{A_iB_i}} \tag{22}$$

If C moves on a segment the derivative of the articular velocity with respect to λ is constant. Hence the minimal and maximal articular velocities will be obtained either for $\lambda = 0$ or $\lambda = 1$. If C moves into a horizontal rectangle we use equations (5) and the derivative of the articular velocity with respect to α, β have a similar form to (7).

10 Conclusion

An algorithm for computing the extremal articular velocities of a parallel robot whose end-effector must be able to perform a given translation/angular velocity over a whole workspace has been presented. It compute the extremal values of the articular velocity with a guaranteed error (without any error if the angular velocity is equal to zero) and is therefore safer than classical method relying on discretisation. Furthermore it is in general faster than the classical method. This algorithm can be used for the optimal design of parallel robots.

References

[1] Gosselin C. *Kinematic analysis optimization and programming of parallel robotic manipulators.* Ph.D. Thesis, McGill University, Montréal, June, 15, 1988.

[2] Gough V.E. and Whitehall S.G. Universal tire test machine. In *Proceedings 9th Int. Technical Congress F.I.S.I.T.A.*, volume 117, pages 117–135, May 1962.

[3] Ling S-H. and Huang M.Z. Kinestatic analysis of general parallel manipulators. In *ASME Mechanisms Design Conf.*, Minneapolis, September, 14-16, 1994.

[4] Martinez J.M.R. and Duffy J. A simple method for the velocity and acceleration analysis of in-parallel platforms. In *9th World Congress on the Theory of Machines and Mechanisms*, pages 842–846, Milan, August 30- September 2, 1995.

[5] Merlet J-P. *Les Robots parallèles.* Hermès, Paris, 1997.

[6] Merlet J-P. Détermination de l'espace de travail d'un robot parallèle pour une orientation constante. *Mechanism and Machine Theory*, 29(8):1099–1113, November 1994.

[7] Sorli M. and others . Mechanics of Turin parallel robot. In *9th World Congress on the Theory of Machines and Mechanisms*, pages 1880–1885, Milan, August 30- September 2, 1995.

[8] Zanganeh K.E. and Angeles J. Instantaneous kinematics and design of a novel redundant parallel manipulator. In *IEEE Int. Conf. on Robotics and Automation*, pages 3043–3048, San Diego, May, 8-13, 1994.