A FORMAL-NUMERICAL APPROACH TO DETERMINE THE ACCURACY OF A PARALLEL ROBOT IN A 6D WORKSPACE

J-P. Merlet

INRIA

France

Abstract: The positioning error of a parallel robot is conditioned by the measurement errors on the leg lengths, these two quantities being linearly related through the pose-dependent jacobian matrix of the robot. An important design problem is to determine the extremum of the positioning errors over a prescribed 6D workspace. This is a difficult problem as the jacobian matrix has a complex formulation, involving thousands of terms. We present the preliminary result for an algorithm that estimate the positioning error with an arbitrary accuracy.

1 Introduction

Parallel robots have been extensively studied this recent years and are now starting to appear as commercial products for various applications, hence the interest for their optimal design. Among other criterion checking the positioning accuracy is an important part of the design process. Surprisingly this issue has been largely ignored in the literature: Patel [4] has studied the positioning error on a trajectory, Ropponen [5] these errors for a given pose, while Masory [1] uses a specific hardware to evaluate the accuracy of a parallel robot. But to the best of the author knowledge, the problem of determining the maximal errors has not been addressed in the literature. In this paper we consider a Gough-type 6 d.o.f. parallel manipulator (figure 1) constituted of a fixed planar base plate and a planar mobile plate connected by 6 articulated and extensible links. The pose of the platform may be adjusted by changing the length of

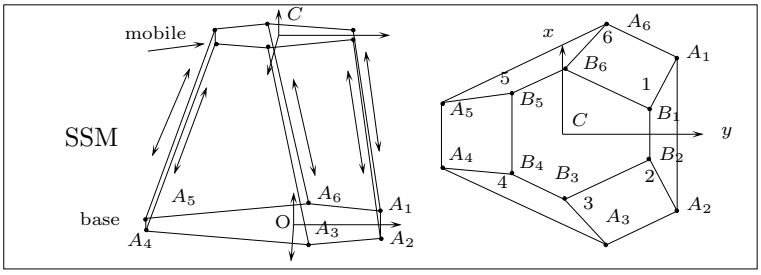

Figure 1: Gough platform

the six legs. A reference frame (O, x, y, z) is attached to the base and a mobile frame (C, x_r, y_r, z_r) is

attached to the moving platform. The leg i is attached to the base with a ball-and-socket joint whose centre is A_i, while it is attached to the moving platform with an universal joint whose centre is B_i. Let ρ_i be the leg lengths (the distance between A_i and B_i), $\boldsymbol{X}$ a 6-dimensional vector defining the pose of the end-effector: the three first components of $\boldsymbol{X}$ are the coordinates of C in the reference frame, while the three last components are three parameters describing the orientation of the end-effector. In this paper we will use the Euler angles ψ, θ, ϕ. The errors on the leg length measurements $\Delta\boldsymbol{\rho}$ and the positioning errors on the end-effector $\Delta\boldsymbol{X}$ are related as follows:

$$\Delta\boldsymbol{\rho} = J^{-1}\Delta\boldsymbol{X} \quad \Delta\boldsymbol{X} = J\Delta\boldsymbol{\rho} \tag{1}$$

where J is the jacobian matrix of the robot and J^{-1} its inverse. In our problem the measurement errors $\Delta\boldsymbol{\rho}$ are known, i.e. we have $-\epsilon_i \leq \Delta\rho_i \leq \epsilon_i$, and we are interested in finding the extremal values of each component of $\Delta\boldsymbol{X}$. Hence the later equation is the one we are interested in. Unfortunately we have a convenient analytical formulation only for J^{-1}; a row of this matrix may be written as:

$$J_i^{-1} = \frac{\boldsymbol{A_i B_i}}{\rho_i} \qquad \boldsymbol{C B_i} \times \frac{\boldsymbol{A_i B_i}}{\rho_i} \tag{2}$$

Although this matrix may be easily expressed in terms of the pose parameters, its inverse has a complex formulation [2]. The maximal value of the positioning error on the i-th pose parameters may be written as:

$$\Delta X_i = \sum_{j=1}^{j=6} |J_{ij}|\epsilon_j$$

Even if we assume that we have been able to compute the jacobian matrix, we will have to determine the maximal value of ΔX_i for all the poses in the prescribed workspace. Clearly this is a difficult optimisation problem.

Now we must examine why we must compute the maximum of the positioning errors. Two main reasons may justify this computation:

- prove that a given robot design has a maximal error less than a given threshold ϵ
- compare the positioning accuracy of two robots with different dimensions

It must be note that in both cases it is not necessary to determine the *exact* value of the maximal positioning error ΔX. This may be shown easily in the former case: assume that we are able to determine a number G such that we can guarantee that $G - \mu|\Delta X| \leq G + \mu$, where μ is an arbitrary error margin. Now suppose that $G + \mu < \epsilon$ or $G - \mu > \epsilon$: then we have determined that in the first case the robot verify the desired accuracy property while in the second case that it fails. Otherwise we cannot answer the question and we will have to decrease μ and start again. Note however that if we are free to fix the value of μ at will, then we may get a very good approximation of the exact value of the maximal positioning error.

Thus our main goal will not to compute the error *exactly* but up to the given accuracy μ. A direct result is that the computation time will be quite low for relatively large μ: in the two problems presented

above we will start with large μ and reduce its value only if the result is indeterminate. This approach will be, in general, quite efficient.

2 Positioning errors and algorithm principle

Let S_i be the matrix obtained from J^{-1} by substituting the i-th column of J^{-1} by the vector $\Delta\boldsymbol{\rho}$. By solving the linear system (1) we get the positioning error on X_i as:

$$\Delta X_i = \frac{|S_i|}{|J^{-1}|}$$

Let S_i^j be the minor of S_i obtained by removing the i-th column and j-th row of this matrix. We have:

$$\Delta X_i = \frac{\sum_{j=1}^{j=6}(-1)^{i+j}\Delta\rho_j|S_i^j|}{|J^{-1}|} \tag{3}$$

We define now the *semi inverse jacobian matrix* M by its i-th row:

$$M_i = (\boldsymbol{A_iB_i} \qquad \boldsymbol{CB_i} \times \boldsymbol{A_iB_i}) \tag{4}$$

A direct consequence of this definition is that:

$$|J^{-1}| = \frac{|M|}{\prod_{k=1}^{k=6}\rho_k} \quad |S_i^j| = \frac{|M_i^j|}{\prod_{k=1,6}^{k\neq j}\rho_k} \tag{5}$$

It must be noted that contrary to $|J^{-1}|, |S_i^j|$ the expressions $|M|, |M_i^j|$ have no denominator. Reporting these results in equation (3) we get:

$$\Delta X_i = \frac{\sum_{j=1}^{j=6}(-1)^{i+j}\Delta\rho_j|M_i^j|\rho_j}{|M|} \tag{6}$$

Consider now that the pose parameters should lie in some workspace $\mathcal{W}$ and let U_{ij}^M be the maximal absolute value of $(-1)^{i+j}|M_i^j|\rho_j/|M|$ for all the possible poses in the workspace. A consequence of equation (6) is that:

$$T_{ij}^M = \sum_{j=1}^{j=6} -\epsilon_j U_{ij}^M \leq \Delta X_i \leq T_{ij}^m = \sum_{j=1}^{j=6} \epsilon_j U_{ij}^M \tag{7}$$

Assume now that we want to determine the extreme values $\Delta^m X_i, \Delta^M X_i$ of ΔX_i with a given accuracy μ, i.e. we want to find 2 quantities V_i^m, V_i^M such that:

$$V_i^m - \mu \leq \Delta^m X_i \qquad \Delta^M X_i \leq V_i^M + \mu$$

Initial values for V_i^m, V_i^M may be obtained by taking random poses in the workspace. If we are able to compute bounds on T_{ij}^m, T_{ij}^M for all poses in the workspace and update V_i^m, V_i^M in such way that

$$V_i^m - \mu \leq T_{ij}^m \qquad T_{ij}^M \leq V_i^M + \mu \tag{8}$$

then V_i^m, V_i^M may be considered as extremal values of the positioning error, up to the accuracy μ. So the basic principle of our algorithm for computing the extremal values of ΔX_i is

1. initialize V_i^m, V_i^M by computing the positioning error at poses randomly selected in $\mathcal{W}$
2. compute bounds on T_{ij}^m, T_{ij}^M for all poses in $\mathcal{W}$
3. if these bounds satisfy the inequalities (8) return V_i^m, V_i^M as extremal values of the positioning error
4. otherwise split $\mathcal{W}$ in smaller subset. In each subset compute the positioning error at some randomly selected poses and update V_i^m, V_i^M if necessary. Then repeat the process for each subset and return the largest extremal values obtained for all the subsets.

We have now to explain how to perform step 2 of this algorithm and how to split $\mathcal{W}$ in smaller subsets.

3 Finding bounds on the positioning errors

We have to determine bounds on the extremal values of a sum of terms, each of them having the generic form $\rho|M_m|/|M|$, where M_m is a minor of M. This sum has a very complex formulation, so we will consider independently each term of the sum, find bounds on the extremal values of each term and consider that the bounds on the sum is the sum of the bounds on each term. Each term is still a complex function of the pose parameters and of the geometry of the robot, as defined by the location of the joints of the robot. So we have decided to use the symbolic computation software MAPLE to compute a generic form for the determinants $|M_m|, |M|$. According to the structure of the elements of the matrix J^{-1} these determinants may be written generically as:

$$|..| = \sum_{a=1}^{a=n} F_a$$
$$F_a = C_a x^i y^j z^k \cos^l(\psi) \sin^m(\psi) \cos^n(\theta) \sin^o(\theta) \cos^p(\phi) \sin^q(\phi)$$

where the C_js are numerical constants. When running our algorithm a MAPLE program will calculate each determinant and determine all its basic components F_a, that it will then write in a result file, each line representing one F_a as a set of float number (for C_a) and 9 integers representing $i, j, k, l, m, n, o, p, q$.

We will then use a method called *interval analysis* [3] to evaluate lower and upper bounds for the F_a. Interval analysis is similar to real analysis except that real variables are replaced by intervals and that specific rules are used for each basic arithmetic operations.

More generally if $X_i = [\underline{x_i}, \overline{x_i}]$ with $\underline{x_i} \leq \overline{x_i}$ denotes an interval it is possible to define all the arithmetic operators (and more complex functions) for intervals. For example the "+" operator for intervals will be defined as:

$$X_1 + X_2 = [\underline{x_1} + \underline{x_2}, \overline{x_1} + \overline{x_2}]$$

With this method we are able to compute lower and upper bounds on any function for given ranges for the input parameters. For example being given 6 ranges for the variables $x, y, z, \psi, \theta, \phi$ we are able to compute two real number F_a^-, F_a^+ such that we can guarantee that for any value of the variable in their

ranges we have $F_a^- \leq F_a(x, y, z, \psi, theta, \phi) \leq F_a^+$ (note that some implementation of interval analysis operators take into account round-off errors to get *guaranteed* bounds).

A drawback of interval analysis is that the resulting bounds may be largely over estimated (e.g. the bounds for the function x^2+x when $x \in [-1, 0]$ are computed as [-1,1], while the real bounds are [-1/4,0]), but they will get closer to the exact values as soon as the size of the input ranges decrease. Thus for given ranges on the pose parameters we are able to compute bounds on ρ, $|M_m|$ and $|M|$ and consequently bounds on $\rho|M_m|/|M|$ which in turn enable to compute bounds on the positioning errors.

4 The bisection process

Suppose that the workspace $\mathcal{W}$ is defined as a set of ranges, one for each pose parameters. We will call this type of workspace an *extended box* or EB for short. To create the subsets of $\mathcal{W}$ that are necessary in our algorithm we will bisect each range of the workspace: if the initial range is $[x_1, x_2]$, the bisection process will lead to the two ranges $[x_1, (x_1+x_2)/2]$, $[(x_1+x_2)/2, x_2]$. Thus, if none of the pose parameters has a constant value, we will get $2^6 = 64$ new EBs as a result of the bisection process. These EBs will be stored in a list and the algorithm will be used for each EB in the list until all the EBs in the list have been processed.

5 Dealing with other types of workspace

We have described in the previous section an algorithm to compute the extremal values of the positioning errors if the workspace is an EB. This algorithm allows us to deal with other type of workspace. Assume for example that the workspace is defined as three ranges for the orientation parameters and a geometric object (e.g. a sphere) as the possible location of the origin of the end-effector. We will assume that we are able to define an EB which include the workspace $\mathcal{W}$ so defined and that we may define a test $\mathcal{T}$ for an EB which return 0 if the EB is outside $\mathcal{W}$, 1 if the EB is fully inside $\mathcal{W}$ and 2 if only a part of the EB is inside $\mathcal{W}$. The previous algorithm will be modified at the bisection level where the test $\mathcal{T}$ will be applied on each EB resulting from the bisection. If the test returns 0 we just discard the EB, if it returns 1 we compute the positioning errors for this EB and update if necessary the current extremal values of the positioning errors. If the test returns 2 and the positioning errors for this EB may be larger than the current extremal values, then we store the EB in the list without updating the current extremal errors. A similar approach enable to deal with a workspace defined in terms of bounds on the articular coordinates i.e. we are able to compute the extremal positioning errors *for all* reachable poses of the end-effector.

6 Experimental results

In this section we will assume that the error ϵ on the measurement of the articular coordinates is 1/100. We will assume that the accuracy μ with which we want to calculate the extremal values of the positioning

errors are $\mu_{\Delta x} = 0.0291$, $\mu_{\Delta y} = 0.1329$, $\mu_{\Delta z} = 0.0189$, $\mu_{\Delta\theta_x} = 0.0018rd$, $\mu_{\Delta\theta_y} = 0.00086rd$, $\mu_{\Delta\theta_z} = 0.005rd$ (these values have been selected as 10 % of the extreme positioning errors computed for a few random poses in the workspace). The computation time has been established on a SUN Ultra 1 workstation.

The origin of the end-effector is located in a workspace defined by $x, y \in [-5, 5]$, $z \in [45, 50]$.

First we assume that the orientation of the end-effector is constant and defined by $\psi = \theta = \phi = 0$. We get the maximal positioning errors as $\Delta x = \pm 0.145762$, $\Delta y = \pm 0.664774$, $\Delta z = \pm 0.0946431$, $\Delta\theta_x = \pm 0.0092133rd$, $\Delta\theta_y = \pm 0.004327rd$, $\Delta\theta_z = \pm 0.025138rd$ in a computation time of 0.1 s. If we decrease by half the value of the μs or divide them by 10 we get the same result for the positioning errors in 0.16s and 8.95s respectively. We assume now that the angle ψ may vary in the range [-5,5] degree. The computation time with the same accuracy than above becomes 31.6 seconds.

Then we assume that both angles ψ, θ may vary in the range [-5,5] degrees. The computation time drastically increase and become about 11h.

7 Improving the efficiency

As may be noted in the previous section the efficiency of the algorithm is pretty good up to a 4D workspace, but drastically decrease as soon as we move to a 5D workspace. We will now suggest three possibilities to improve the efficiency.

7.1 Improving the bounds on the determinants

7.1.1 Using the nested form

We have seen that we expand all the determinant involved in the calculation as an expression involving a sum of generic terms F_a. But it is well known in the field of interval arithmetics that the Horner (or "nested" form) of an expression leads, in general, to better bounds. For example we have considered previously the expression $x^2 + x$ with x in the range $[-1, 0]$ and we haven that the corresponding interval evaluation was [-1,1]. If we consider now the nested form $x(x + 1)$ of this expression we get the interval evaluation $[-1, 0]$ and we have reduced by the two the width of the interval evaluation.

The nested form of the determinants can be computed by MAPLE but the main difficulty is to use it in a program. To deal with this problem we will use a parser that has been developed in our laboratory. This parser is able to read the analytical expression of a function written in a file and, being given ranges for the unknowns, will compute the interval evaluation of the function. Using this parser is somewhat slower than a direction evaluation of the expression but it is expected that the decrease in the width of the interval evaluation will drastically increase the efficiency of the algorithm.

7.1.2 Evaluation o the trigonometric expressions

In the determinants appear the sine and cosine of the orientation angles. These quantities are clearly not independent but are considered as such in term of interval analysis. For example if at some point appears the quantity $\sin\psi + \cos\psi$, the interval evaluation of this quantity for ψ in $[0, 2\pi]$ will be [-2,2], while the exact value is $[-\sqrt{2}, \sqrt{2}]$. We have developed a program that is able to detect such expression and substitute it by a procedure that will compute better bounds.

7.2 Decreasing the number of EB

For a 6D workspace the bisection process will lead to up to 64 new EB's each of which in turn may lead to 64 new EB's. Consequently the number of EB stored in the list is quickly growing. To avoid this effect we may use a well known method in the field of interval analysis: instead of bisecting the EB in all its 6 dimensions, we may bisect only one of them. This may have a very large impact on the efficiency. Indeed the bisection of an EB along one dimension leads to 2 new EB's. Imagine now that the algorithm is able to calculate that the maximal positioning error cannot be obtained for these 2 EB's, and consequently that they have not to be added to the list. We will evidently get a similar result by using the algorithm described in section 4 but at the cost of examining 64 EB's, instead of only 2. As bisecting along one dimension will lead in the worst case to the same number of new EB's as bisecting along all the dimensions, it may be seen that bisecting along one dimension may be a better approach.

It remains to determine which of the 6 dimensions should be selected as the bisected dimension. A classical approach for this problem is to use the "smear" function. Let $G(X)$ be a function of $X = \{x_1, x_2, \ldots, x_n\}$ and we define s_j as

$$s_j = \text{Max}\{|\frac{\partial G(X)}{\partial x_k \partial x_j}|(\overline{x_j} - \underline{x_j})\}$$

where the jacobian $\partial G(X)/\partial x_k \partial x_j$ is evaluated using interval analysis. In some sense s_j is a measure of the influence of the variable x_j on the value of G, which is weighted by the width of range on x_j. As a consequence the direction which will be bisected is the one which has the larger s_j.

8 Conclusion

We have presented here a preliminary approach to deal with one of the difficult problems related to parallel robots: determining the extreme values for the positioning errors for given ranges on the errors on the measurements of the articular coordinates. The experimental results show that our algorithm perform pretty well for workspace of dimension up to 4, while being quite slow for larger dimension. We have suggested ways to improve the efficiency of the algorithm, which are currently being implemented.

References

[1] Masory O. and Jihua Y. Measurement of pose repetability of Stewart platforms. *J. of Robotic Systems*, 12(12):821–832, 1995.

[2] Mayer St-Onge B. and Gosselin C. Singularity analysis and representation of spatial six-dof parallel manipulators. In J. Lenarčič V. Parenti-Castelli, editor, *Recent Advances in Robot Kinematics*, pages 389–398. Kluwer, 1996.

[3] Moore R.E. *Methods and Applications of Interval Analysis*. SIAM Studies in Applied Mathematics, 1979.

[4] Patel A.J. and Ehmann K.F. Volumetric error analysis of a Stewart platform based machine tool. *Annals of the CIRP*, 46/1/1997:287–290, 1997.

[5] Ropponen T. and Arai T. Accuracy analysis of a modified Stewart platform manipulator. In *IEEE Int. Conf. on Robotics and Automation*, pages 521–525, Nagoya, May, 25-27, 1995.