Appropriate Design of Parallel Manipulators

J.-P. Merlet, D. Daney

1

Appropriate Design of Parallel Manipulators

J.-P. Merlet and D. Daney

INRIA Sophia-Antipolis
2004 Route des Lucioles, 06902 Sophia-Antipolis Cedex, France
Email: Jean-Pierre.Merlet@sophia.inria.fr

Abstract
Although parallel structures have found a niche market in many applications such as machine tools, telescope positioning or food packaging, they are not as successful as expected. The main reason of this relative lack of success is that the study and hardware of parallel structures have clearly not reached the same level of completeness than the one of serial structures. Among the main issues that have to be addressed, the design problem is crucial. Indeed, the performances that can be expected from a parallel robot are heavily dependent upon the choice of the mechanical structure and even more from its dimensioning. In this chapter, we show that classical design methodologies are not appropriate for such closed-loop mechanism and examine what alternatives are possible.

1.1 Introduction

Historically, closed-chain structures have attracted the interest of mathematicians as they offer interesting problems. Some theoretical problems linked to this type of structures were mentioned as early as 1645 by Christopher Wren, then in 1813 by Cauchy [1.1] and in 1867 by Lebesgue [1.2].

But clearly at that time, the technology was not able to deal with any practical applications of this type of structures. The very first application was proposed by Gough for a tire test machine [1.3][1.4], although parallel structures were really put in practice in the 1970's for a flight simulator with the patent of Cappel in 1964 [1.5] and the seminal paper of Stewart [1.6]. Robotics applications were proposed in the early 1980's [1.7][1.8].

Starting in the 1990's, parallel kinematic machines (PKM) have started either to be put in use in various domains such as fine positioning devices or to be considered for potential applications such as machine tools. Although parallel structures have found a niche market in many applications, such as machine tools, telescope positioning or food packaging, they are not as successful as expected.

In our opinion, the main reason of this relative lack of success is that the study and hardware of parallel structures have clearly not reached the same level of completeness than the one of serial structures. In this chapter, we will address the

design issue that is still an open problem for the closed-loop mechanisms. We will assume that the starting point of a design study is a list of wishes about the performances that the robot must satisfy, which is provided by the end-user.

The design process will then proceed along the following steps:

1. translate the end-user wishes into numerical indices
2. choose the mechanism structure
3. choose the dimensioning of the robot

The following sections of this chapter will address these three issues.

1.2 Understanding End-user Wishes and Performance Indices

1.2.1 Establishing the Required Performances

Our experience in design problems is that it is usually very difficult to understand the end-user wishes and to get the numerical values of the performances that he/she is expecting from the robot. To help, we have established a relatively exhaustive requirements list with the following items:

- *number of DOF* (degree of freedom): this data will be important for the choice of the mechanism;
- *workspace description and maximal travel*: most often these data are among the wishes of the end-user but it is important to fully understand them. For example, for a 6 DOF robot the end-user may mention a maximal translation range and a maximal orientation range: should the maxima be reachable simultaneously?
- *geometry and mass of the load*: these data will be important to compute the maximal forces and torques in the joints of the robot;
- *footprint of the robot*: constraints on the overall size of the robot will help us define the upper bound for some design parameters;
- *actuators, joints*: the end-user may have special relations with providers that will impose constraints on the choice of the hardware. The designer should take that into account to provide a design solution that best fits the available hardware of these providers;
- *stiffness*: for some application such as vibration testing, this point is very important;
- *positioning accuracy*: in many applications this is a key point. You will have to ensure that over all the possible poses included in the prescribed workspace, the positioning accuracy is better than some given thresholds;
- *internal sensors*: the positioning accuracy is clearly very dependent upon the accuracy of the sensors that are used to measure the internal state of the robot. Here again the end-user may have a special relation with providers, allowing the use of sensors with only a limited choice in accuracy;
- *passive joints*: motion range, permitted load, maximal velocity and friction will play an important role when choosing the passive joints;
- *velocity, dynamics*: the constraints on the velocities and accelerations of the end-effector will influence the choice of the actuators;

- *cost*: this is clearly an important point for the end-user that may influence the designer.

You will have them to rank the requirements as not all of them may have the same importance for the end-user. Some of them will be *essential* and your design must fulfil them while some of them may be *secondary*, *i.e.* the end-user may relax them if necessary. When this list is completed, you may start a design analysis because

- you have numerical values for the constraints, and
- all the constraints have a well-defined mathematical sense.

Clearly, the requirements established with this list are not exhaustive as the end-user may not be aware of all design aspects. For example, the list does not consider singularity issue (most probably the workspace defined by the end-user must be singularity-free), although this is a crucial design point. Hence, the designer will add requirements to the list that are stemming from his/her expertise. The requirements can be categorised as follows:

- *imperative*: these performance requirements must be satisfied for any design solution;
- *optimal*: a performance index is associated to the requirement and a maximal value of this index is required;
- *primary*: although these performance requirements are specified in the specifications, their values may be modified to some extent if no design solution is found;
- *secondary*: these requirements may not appear in the specifications list, but may be used to choose between design solutions that satisfy imperative, optimal and primary requirements.

It must also be noted that usually you will have to deal with a problem for which multiple criteria are involved, some of which are *antagonistic* (*e.g.* increasing the workspace size will have a negative influence on the positioning accuracy and *vice versa* [1.9]).

Mechanism design is often qualified as *optimal design* although it may be seen that in many cases there will not be performances to be optimised but only those to be guaranteed (*e.g.* the positioning accuracy of the robot over a given workspace is better than a fixed threshold): hence we prefer to use in that case the term *appropriate design*. Even if some performances have to be optimised in most cases, we will still have to satisfy imperative requirements (*e.g.* the workspace volume cannot be lower than a fixed threshold). Hence establishing the design problem as an optimisation one will lead to a constrained optimisation problem, a mathematical problem that is among the most difficult to deal with.

We will see later on that evaluating the maximal performances of a robot is not a trivial issue but having inequality constraints will simplify our calculations. Indeed in that case, it is not necessary to compute *exactly* the performance indices as long as the design methods allow one to determine upper and lower bounds on their values.

Another important issue that is often neglected in design algorithms is the uncertainties that are unavoidable in mechanical design. We will distinguish two different types of uncertainties:

- *mechanical*: most, if not all, of the design parameters have a physical meaning and their physical instances will never be exactly identical to their nominal values. However, the end-user is expecting that the *real* robot (not the theoretical one) will satisfy the requirements. A designer must take these uncertainties into account and must inquire the end-user about them.
- *computational*: the designer will use computer(s) to determine the design solution(s). Numerical round-off errors occur in computer calculations (and more frequently than may be thought, see Section 1.5.4.1). The round-off errors must be taken care of.

1.2.2 Performance Indices

As mentioned previously, a design process requires to associate numerical values to end-user requirements. The translation of the end-user wishes into the requirements list that has been proposed in the previous section leads to performances indices that have a real physical meaning and are well defined in the mathematical sense. However, the proposed requirements are certainly not exhaustive and you may have to introduce others performance indices to deal with specific end-user wishes.

Many performance indices have been defined in the past, especially for serial robots. However, you must be careful when using them. Three issues have to be considered:

- does the performance index translate exactly the end-user wish?
- is the performance index appropriate for closed-loop chains?
- can we compute the performance index with a reasonable accuracy or at least with a bounded error?

These three questions are often neglected and may lead to consider performance index that are not appropriate. A typical index that has to be considered with the most extreme caution is the *condition number* that is very often used for the design of parallel robots.

1.2.2.1 Condition Number

If $\boldsymbol{\Theta}$ and $\mathbf{X}$ denote respectively the joint variables and the pose parameters, it is well known that there is a linear relationship between the variations of these two types of parameters

$$\Delta\boldsymbol{\Theta} = \mathbf{J}^{-1}\Delta\mathbf{X} \tag{1.1}$$

where $\mathbf{J}^{-1}$ is the inverse Jacobian matrix of the robot. The condition number is defined as the amplification factor k between the relative variation of $\boldsymbol{\Theta}$ and the relative variation of $\mathbf{X}$

$$\frac{\|\Delta\mathbf{X}\|}{\|\mathbf{X}\|} \leq k \frac{\|\Delta\boldsymbol{\Theta}\|}{\|\boldsymbol{\Theta}\|} \tag{1.2}$$

It must be noted that the condition number value is dependent upon the choice of the norm and is sensitive to the choice of the length units as long as both translation and orientation are involved in $\mathbf{X}$.

Clearly, if the condition number is very large, even a small error on the joint control will induce a large positioning error for the platform: such situation should be avoided. But the condition number cannot be used to rank the accuracy of the robot as it bounds only the relative variations while we are only interested in the amplification factor γ that relates the absolute change of the variables

$$\|\Delta \mathbf{X}\| \leq \gamma \|\Delta \boldsymbol{\Theta}\| \tag{1.3}$$

It may easily be found cases where at a given pose robot 1 has a better index $1/k$ than robot 2 while the opposite is true for the γ value [1.10]. In other words, the condition number will provide incorrect information: robot 1 will be ranked more accurate than robot 2 while the converse is true.

1.2.2.2 Global Conditioning Index

Even assuming that the condition number has a significant meaning, it is valid only at a given pose and hence will not provide an overview of the performance of the robot over its workspace. Gosselin [1.11] has introduced the *global conditioning index* (GCI) as the average value of the inverse of the condition number over a given workspace W

$$GCI = \frac{\int_W 1/k \, dW}{\int_W dW}. \tag{1.4}$$

As the condition number usually does not have a closed-form, only an approximation of the GCI can be calculated by sampling the workspace, computing k at each sampled poses and averaging the result. But this method has the drawback that it is impossible to bound the difference between the true GCI and its approximation. As this index will be used to rank different design solutions, an unknown error in the approximation may lead to incorrect result.

1.2.2.3 Necessary Properties of Performance Indices

A large number of performances indices have been proposed in the literature: for singularity analysis [1.12]–[1.14], statics [1.15]–[1.18] and workspace analysis [1.19] to name a few. Depending on the design job, it is necessary to choose performance index S that satisfies the following properties:

- P_1: $S(\mathbf{X})$ must have a clear physical meaning;
- P_2: $S(\mathbf{X})$ should be invariant under a change of units; and
- P_3: $S(\mathbf{X})$ should be able to be calculated with a reasonable accuracy or at least with a bounded error.

The requirements list we have presented in the previous section satisfies the first two properties while, like for all performance indices, property 3 is the most difficult to satisfy.

1.2.3 Indices Calculation

Developing methods to calculate indices properly requires a strong mathematical background in numerical analysis, geometry and optimisation even if numerous software packages are available for these calculations. In many cases, however, you will have to develop customised code for the sake of efficiency as the design process will be computing intensive. An important point is to keep in mind that in many cases it is not necessary to compute *exactly* the indices but only their upper or lower bounds with an arbitrary accuracy in order to be able to compare different structures. For example, we may consider the problem of finding the maximal joint forces of a Gough platform over a given translational workspace. This is a difficult constrained optimisation problem if we try to solve it exactly. But if the purpose is to verify the property that the maximal articular forces are lower than a given threshold T_S, we do not need to compute exactly the maximum. We will use a strategy [1.20] that computes the maximum articular forces up to a pre-defined accuracy ε, which is based on two results

- it is possible to compute exactly the maximal value of the articular forces T_M for any pose lying on a straight line segment D whose end-points lie on the border of the workspace. For the sake of simplicity, we will consider a line whose axis is the x axis of the reference frame;
- if we consider another straight line segment D_1 whose end-points lie on the border of the workspace, which is parallel to D and at a distance d from D, then it is possible to find d such that the maximum articular forces T_M for all poses on D_1 are such that $|T_M - T_m| \leq \varepsilon$ (A).

For computing the maximal articular forces in the desired workspace, we will first sweep the section of the workspace that has minimal z value by straight line segments in that plane, starting by a segment with minimal y and increasing the y coordinate of the segments in such a way that the difference between the maximum articular forces of two segments is never greater than ε. The sweep is stopped when a segment has a y coordinate the largest y value for the desired workspace. We then start a new sweep by changing the z coordinate of the last sweep line in such a way that (A) still hold. This defines a new horizontal plane that is swept until the last line has the minimal possible y value at this height. The process is repeated until the plane with the largest z value has been swept.

This algorithm does not compute exactly the maximal articular forces and its computation time will change according to the value of ε. If the result of the algorithm T_M is such that $T_M + \varepsilon < T_S$, we can guarantee that the property is satisfied. On the other hand if $T_M > T_S$, the property is violated. If none of these two inequalities hold, we will half ε until one inequality become valid.

Such algorithm is typical of lazy algorithms that will compute an (almost) exact result only when required and whose computation time may be very low for large ε. In our opinion, lazy algorithms must be used as much as possible in a design job.

1.3 Structural Synthesis

Traditionally, a design process starts by the choice of the general mechanical arrangement of the structure. This tradition in mechanical engineering was followed for the design of serial robots as there are a relatively limited number of possible arrangements. Furthermore for serial chain, qualitative comparison between structures is sometime possible. For example, the workspace volume of a Cartesian robot using 3 linear actuators of stroke L is roughly L^3 while this volume for a 3R robot whose links has length L is roughly $4\pi(3L)^3/3 \approx 113L^3$: hence, in general, a 3R robot will have a much more larger workspace than a Cartesian robot for a similar dimensioning.

However, for closed-loop chains the number of possible mechanical structure arrangements is much larger. It is difficult to determine exactly how many structures have been proposed in the literature but this number will largely exceed 200. They have been obtained more or less by using systematic approaches that consider the number of DOF that are required for the task at hand and synthesise the corresponding structures. We will call this step of the design process the *structural synthesis*.

Various approaches have been proposed for structural synthesis: based on mobility formula [1.21][1.22], graph theory [1.23], group theory [1.24]–[1.28] and screw theory [1.29]–[1.32].

We will not describe further these approaches but various remarks.

- The performances of parallel structures are dependent upon dimensioning, and consequently structural synthesis cannot usually be dissociated from dimensional synthesis (to be addressed in the next section), although Rao [1.33] has addressed this issue for planar robots. Our conjecture is that a well-dimensioned robot with a reasonable mechanical structure will perform much better than a poorly designed robot whose mechanical structure is a-priori more appropriate for the task at hand.
- An open problem is to propose systematic structural synthesis based on criteria that are not only the number of DOF of the platform.

Recently there has been a large effort on the structural synthesis of robot with less than 6 DOF. The main motivation of these studies is that for many applications, less than 6 DOF may be needed. For example, for milling operation in the machine tool domain, the rotation of the platform around its normal is not needed, as the spindle will manage this DOF: hence only 5 DOF are needed.

It must, however, be noted that the proposed structure have the desired number of DOF only in theory. Indeed in most cases, very strict geometrical constraints must be satisfied to get the right number of DOF (*e.g.* perfect parallelograms for the Delta robot). These constraints will never be satisfied in practice and the real robot will exhibit *parasitic motion*, *i.e.* motion along DOF that were not required. Parasitic motion may be acceptable provided that their amplitudes are limited, but an open problem is to determine the maximal amplitude of these motions, given the manufacturing tolerances and other sources of uncertainties.

Another motivation for the study of robots with less than 6 DOF is that they will be less costly than the usual 6 DOF parallel robots as they have fewer actuators, and

that the control will be simpler. The veracity of this claim is a complex issue. First of all, the cost of the machine is only a part of the operating cost, and various factors may increase the cost of less than 6 DOF robots such as maintenance and fabrication costs that may be higher if the chains of the robot involve different actuators and sensors. In terms of electronic hardware, costs will be almost similar as only additional power amplifiers will be required (usual electronic boards will manage up to 8 axes and the control computer will be the same). Furthermore, the redundancy of a 6 DOF robot may be used to improve the quality of the control, the workspace and to manage singularities [1.34]. All these factors must be considered before going to the use of a robot with less than 6 DOF.

1.4 Dimensional Synthesis

We will now assume that the general mechanical structure of the robot has been chosen and will address the issue of finding the appropriate dimensioning of the robot. This is a crucial issue that will have drastic influence on the performances of the robot. A good example for emphasising the importance of dimensional synthesis for parallel structure is to look at the minimal stiffness values of a Gough platform over a given workspace. It may be shown that changing the radius of the platform by only 10% may leads to a change of the minimal stiffness by over 700%.

1.4.1 Choosing Design Parameters

The first task of a designer is to choose the design parameters. Clearly, this number should be the smallest possible. For example, it will be quite difficult to manage the 138 parameters identified by Vischer [1.35] that define the basic geometry of a 6-UPS robot. Furthermore, these parameters are not sufficient to fully describe the robot as technological quantities, such as minimal and maximal leg lengths, limits on the passive joints motion, geometry of the legs, *etc*., must be added.

There are no simple guidelines for reducing the number of design parameters. Symmetry is very often useful as long as the task does not favour specific working directions. Otherwise, there is no clear general rule that allows for neglecting the influence of some parameters.

1.4.2 Design Methods

We will now introduce classical design methodologies in mechanism theory and we will show that their limitations and drawbacks are such that they cannot be usually used for the design of parallel robots.

1.4.2.1 Trial and Error

The first approach to dimensional synthesis is trial and error, which consists of manually modifying the dimensions of the mechanism, and then evaluating the performance of the new mechanism after each modification, with the help of a simulation software, until a mechanism is obtained that is deemed satisfactory. Unfortunately, this approach is almost impossible to use for parallel structures as the

number of design parameters is large while their influence is most of the time antagonistic: it is hence quite difficult to manually figure out an appropriate design. Furthermore, it is often needed to develop a customised simulation software package as classical CAD systems may have difficulties when dealing with the closed-loop mechanism [1.36].

1.4.3 The Atlas Approach

The idea of the atlas approach is first to reduce the number of design parameters to 2 or 3 so that performance indices may be graphically represented as atlases. The designer then uses these atlases to choose the design parameters.

An early use of the atlas approach for the design of parallel manipulators has been mentioned by Clavel [1.37] for the dimensioning of the *Delta* robot. Bhattacharya [1.38] calculates the average value of stiffness related indices over the workspace by using a discretisation method, and draws curves that show the value of the various criteria as functions of the design parameters for a 6-UPS robot. For the 3-UPU robot, Badescu [1.39] plots the workspace volume, the average of the inverse of the condition number and the GCI; Hong [1.40] defines global torque, force and velocity manipulability measures, and plots them as function of 2 design parameters. Liu [1.41] plots the distribution of the shape of the workspace of planar and spatial robots in the design parameter space. Masuda [1.42] plots various manipulability measures as functions of the design parameters in order to choose the best design of a 6-PUS robot.

Clearly, the atlas approach is very limited, and may be used only for a very small set of design parameters.

1.4.4 Cost Function Approach

The design methodology that is mainly used for parallel robots is the cost-function approach [1.43]. As we have already seen, we may define performance indices associated to design requirement and we may assume that each index is positive and decreases when the corresponding robot performance increases. The cost function C is defined as:

$$C = \sum w_j I_j ,$$

where the I_j are the performances indices and w_j are weights associated to the I_j. In some sense, the cost function is viewed as an indicator of the global behaviour of the mechanism with respect to the requirements. As C is clearly a function of the set of design parameters P, a numerical procedure is then used to find the value of the design parameters that minimise C. This approach has several drawbacks [1.44]:

- Defining the index I is not always an easy task, *e.g.* if we have constraints on the shape of the workspace. Furthermore, as mentioned earlier, some of these indices are even very difficult to estimate exactly (for example, the global conditioning index) and their calculation is computer intensive;
- Managing imperative requirements normally requires to solve a constrained minimisation problem, that is usually very difficult;

- Finding the global minimum of a constrained problem is a difficult numerical problem as most optimisation software tools will have problems with local minimum [1.45][1.18]. Error at this level is in jeopardy of the whole design methodology;
- The weights are used to balance between different requirements that do not have the same unit and represent very different physical quantities. If the solution that is found is not satisfactory, it is quite difficult to figure out how to change the weights;
- As seen previously, some of the requirements are antagonistic and hence no classical optimum can be calculated but only Pareto one (a design parameters set P^0 is called a *Pareto optimum* if there is no other set P such that $I_i(P) \leq I_i(P^0)$, $i = 1, \ldots, m$ with strict inequality for at least one i). It can be shown that in general it is impossible to adjust the weights to find all Pareto optimum [1.44];
- This methodology only provides a single solution whose sensitivity to manufacturing uncertainties is not managed.

This methodology has been proven to be relatively effective only for simple cases such as 2 DOF with a limited number of requirements [1.46][1.47].

1.4.5 Other Design Methodologies Based on Optimisation

To override the drawbacks of the cost function approach, several other optimisation approaches have been proposed.

If the design problem has m requirements, we may define an m-dimensional space H whose points have, as coordinates, the values of the m performance indices I_i. In the *Compromise Programming* methodology [1.48], the performances *utopia* point is defined as the point of H whose coordinates are all minimal. The cost function is then defined as a weighted distance between the utopia point and the point that represents the performance indices of a given robot. But here again, we are confronted with difficulties: determining the performances indices, finding the utopia point, managing the weights, and solving the optimisation problem.

To avoid the use of weights, the *Physical Programming* methodology [1.49] proposes to define classes of constraints for the performance indices, and for each class a degree of desirability, from highly desirable to unacceptable, is defined for different ranges for the index value. A cost function taking into account the desirability may then be defined, with imperative requirements being considered as constraints for the optimisation. But this approach basically substitutes the arbitrary weights of the cost function by desirability weights and will hence suffer from the same drawbacks.

1.4.6 Exact Design Methodologies

If the number of design parameters is small, it may be possible to find analytically all design solutions.

Workspace is a requirement that is present in most cases and various works have addressed dimensioning methodologies for a prescribed workspace [1.46][1.50]–[1.53]. For example, Chablat [1.54] was able to determine the dimensioning of a 3

DOF translational robot so that its workspace includes a prescribed workspace, and such that the eigenvalues of $\mathbf{J}_k^{-T}\mathbf{J}_k^{-1}$, where $\mathbf{J}_k^{-1}$ is the inverse kinematic Jacobian, lie within a given range for all poses in the prescribed workspace. Huang [1.55] was able to determine analytically the actuator stroke of a 6-PUS robot so that its workspace includes a prescribed translational workspace with a minimum reachable yaw angle. Arsenault [1.56] was able to find the dimensioning of a planar robot with an optimal singularity-free workspace.

Exact synthesis may also be obtained if it is assumed that the requirements are to be satisfied only at a limited number of poses or on simpler motion varieties (*e.g.* a straight line segment). For example, Simaan [1.57] determines what should be the design parameters to obtain a given stiffness matrix at a given pose, while Jafari [1.58] proposes a method for designing a 6-UPS robot so that $\mathbf{J}_{fk}^{-T}\mathbf{J}_{fk}^{-1}$ is diagonal at a given pose, the purpose being to obtain given maximal translational and angular velocities at this pose.

Assuming that the base and platform radii are the only design parameters of a Gough platform, we were able to find all possible radii so that the robot workspace includes a set of pre-defined line segments [1.59]. The result is obtained as a region in a graph, as shown in Figure 1.1, whose *x* axis represents the base radius while the *y* axis is the platform radius.

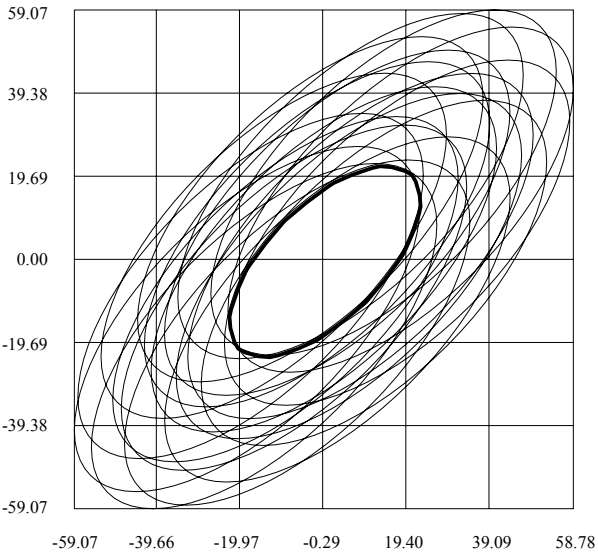

Figure 1.1. The *x*, *y* axes of this graph represent the base and platform radii of a Gough platform, respectively. The region with the border in think line represents all possible values of radii so that the robot workspace will include a set of pre-defined line segments trajectory.

Exact synthesis has a big advantage compared to optimisation: it provides *all* possible design solutions for a given requirement. Unfortunately, exact synthesis is only possible in a very limited number of cases: it is possible to manage only a

limited number of parameters and requirements. Still, the idea of being able to find all design solutions is worth pursuing.

1.5 The Parameter Space Approach

The objectives of the parameter space approach are as follows:

- to propose not one design solution but a set of design solutions that offer various performance compromises for both the primary and the secondary requirements;
- to guarantee that all design solutions satisfy the imperative requirements; and
- to guarantee that all design solutions are robust with respect to manufacturing tolerances, *i.e.* the design parameter values of the real robot may differ from the theoretical design solutions by bounded tolerances, but the real robot will still satisfy the imperative requirements.

1.5.1 Parameter Space

A key point of this approach is the concept of the *parameter space*. Each dimension of this space represents a design parameter, and consequently a point in this space represents a unique geometry for the robot. Searching for an optimal or appropriate robot, therefore, consists in finding the location of the points in the parameter space, such that the requirements are satisfied. Although in theory the parameter space is unbounded, practical considerations on each design parameter generally restricts its value to some range. Hence, the search for an optimal robot will have to be done only within a bounded domain of the parameter space, called the *search space*.

1.5.2 Principle of the Method

In the previous section, we have seen that it was possible in some cases to calculate exactly *all* the possible values of the design parameters, so that the corresponding robots satisfy one set of specific requirements. In terms of parameter space, this calculation amounts to determining a domain of the parameter space that includes all geometries satisfying a given requirement: such domain is called the *allowed region* for the requirement. The design of algorithms for calculating the allowed regions is central in the parameter space approach.

The steps of the method are as follows:

1. define the parameter space;
2. compute the allowed region for each requirement in the requirements list;
3. compute the intersection of all allowed regions: any robot geometry represented by a point in this intersection satisfies all requirements; and
4. determine a set of appropriate robots by sampling the intersection so that different compromises for the primary and secondary requirements are presented in the set.

The advantages of this approach are:

- all design solutions are guaranteed to satisfy the imperative, primary and secondary requirements;
- an optimality requirement may still be satisfied as long as the optimisation procedure allows searching for an optimum within a bounded domain. Indeed looking for the optimum within the intersection of the allowed regions will ensure that the imperative, primary and secondary requirements are still satisfied. Fortunately, we will see later that there exist optimisation methods that use bounded domains;
- manufacturing uncertainties can be taken into account. Indeed, a requirement will not be satisfied due to uncertainty only if the corresponding point in the parameter space is too close to the border of the corresponding allowed region. Hence, by decreasing the allowed region by the uncertainty values, we can obtain a safe design region that is robust with respect to the manufacturing tolerances;
- having a set of design solutions gives some flexibility to manage additional requirements that may appear during the design process.

The difficult point of the methodology is clearly finding the allowed regions.

1.5.3 Finding Allowed Regions

Ideally, an allowed region algorithm should be able to solve the following problem: find all possible design parameters P in the search domain such that some given relations $\mathbf{F}(P, \mathbf{X})$ are satisfied for all poses in a given workspace W. The relation $\mathbf{F}$, called the *requirement constraint*, will involve various types of performance indices involving both the pose of the robot and the design parameters and, therefore, verifying if $\mathbf{F}$ is satisfied is a difficult problem. However, practical and theoretical considerations will play an important role in getting a tractable problem:

- $\boldsymbol{H_1}$, *completeness of the result*: it is not necessary to determine the allowed region *exactly*. Indeed, as the design parameters are submitted to tolerances, point on the border of the allowed region cannot be chosen as nominal value for the design parameters as the physical instance may be *outside* of the allowed region;
- $\boldsymbol{H_2}$, *completeness of the verification of the requirement constraint*: for a requirement constraint involving the verification of an inequality $\mathbf{F}(P, \mathbf{X}) \leq 0$, it is not necessary to calculate the values of $\mathbf{F}$ *exactly* but just to ensure that its maximal value is indeed negative;
- $\boldsymbol{H_3}$, *relaxation of the workspace constraints*: for simplifying the calculation of the allowed region, it is possible to assume that the workspace is reduced to a set of characteristic elements such as poses, or segments between two poses. This imposes only an additional verification step, after completing the design process, in which the design solution performances are checked with respect to the specification list over the entire workspace;
- $\boldsymbol{H_4}$, *distributed implementation*: design is in general computer intensive. But computer science now offers powerful tools for the distribution of heavy calculations over a network of computers. Hence design algorithms that allow for distributed calculation should be favoured.

Another needed feature of the calculation is that the description of the resulting allowed region should be convenient for later intersection with other allowed regions. Clearly, we should seek as much as possible analytic descriptions of the allowed regions, but as seen previously such situation arises only for a limited set of requirements and when the number of design parameters is small. We will present a generic method that has been successful in finding the solutions of complex design problems.

1.5.4 Finding Allowed Regions with Interval Analysis

A generic method that can deal with design problems should allow manipulating arbitrary formulae and bounded domains. These features are typical of interval analysis and will be presented in this section.

1.5.4.1 Interval Analysis

Detailed explanation of interval analysis may be found in [1.60][1.61] and we will provide here only basic principles. Interval analysis relies on *interval arithmetics* whose purpose is to determine guaranteed bounds for the minimum and maximum of a given function f over ranges for the unknowns with a minimal number of calculations. This determination is called an *interval evaluation* of the function and leads to a range $[\underline{F}, \overline{F}]$ that varies according to the ranges for the unknowns. If $\mathbf{X}$ denotes the ranges for the unknowns and $\mathbf{X}_0$ is a particular instance of the values of the unknowns within $\mathbf{X}$, then we have:

$$\underline{F} \leq f(\mathbf{X}_0) \leq \overline{F} \tag{1.5}$$

An interval evaluation may be calculated in different ways. The simplest is called the *natural evaluation*, which consists in using specific interval versions of all mathematical operators used in the function (interval version exists for all classical operators). For example, the addition of two intervals $\mathbf{a} = [\underline{a}, \overline{a}]$, $\mathbf{b} = [\underline{b}, \overline{b}]$ is defined as $\mathbf{a} + \mathbf{b} = [\underline{a} + \underline{b}, \overline{a} + \overline{b}]$. Natural evaluation may be simply illustrated with $f = x^2 - 2x$ when x lies in the range [3, 5]. In that case, we can safely state that for any instance of x in [3, 5], then x^2 lies in [9, 25], $2x$ in [6, 10] and consequently $-2x$ in [–10, –6]. Summing the interval for x^2 and $-2x$ leads to [9, 25] + [–10, –6] = [–1, 19], which constitutes the interval evaluation of f over the range [3, 5].

This example shows that simple operations are required by interval arithmetics, but also one of the drawbacks of the method. Clearly for any x in [3, 5], the value of f lies in [3, 15]: hence interval arithmetics overestimates the values of the minimum and maximum of the function. This occurs because we have multiple occurrences of the same variable in f, which are considered as independent during the calculation. But this overestimation does not always occur and will decrease with the width of the input ranges. Furthermore, there are ways to decrease the size of the overestimation such as, for example, using the monotonicity of the function over the intervals.

An interesting property of interval arithmetics is that it can be implemented to take into account round-off errors. We must emphasise that round-off errors, which are often not considered in robotics, should be dealt with for critical applications. Even if they are not so frequent in occurrence, they may still happen in some simple cases. A classical example of this phenomenon, due to Rump, may be observed when computing the floating point value of

$$333.75y^6 + x^2(11x^2y^2 - y^6 - 121y^4 - 2) + 5.5y^8 + \frac{x}{2y}$$

for $x = 77617$, $y = 33096$. Classical scientific software will return the value -10^{23}, interval evaluation computed in $\mathbb{C}$ is $[-0.56610^{23}, 0.55510^{23}]$, while the real value is ≈ -0.8273960599.

Interval analysis is able to solve very different problems such as system solving (finding all solutions of a system of equations within some bounded problem[1]), and optimisation (finding the optimum of a function when the variables are restricted to lie within a bounded domain). We will now illustrate the use of interval analysis on a realistic robotics problem.

1.5.4.2 A Simple Verification Example

A simple example of interval analysis is a simple algorithm whose purpose is to determine if the workspace of a given Gough platform includes a given desired workspace. We assume that the constraints to be satisfied are

$$\rho^2_{\min_j} \le \rho_j(\mathbf{X})^2 \le \rho^2_{\max_j} \quad \forall j \in [1, 6] \tag{1.6}$$

where, $\rho_j(\mathbf{X})^2$ is the length of leg j for the pose $\mathbf{X}$ (that is an analytical function of the components of $\mathbf{X}$) and $\rho_{\min_j}$, $\rho_{\max_j}$ are the minimum and maximum allowed leg lengths for the same leg, which are known.

The desired workspace W will be defined as interval ranges for the components of $\mathbf{X}$ (*i.e.* the centre of the platform is allowed to move within a parallepiped with orientation angles that may have any value within the given ranges). Being given the ranges for $\mathbf{X}$, we may calculate the interval evaluation $[\underline{\rho_j^2}, \overline{\rho_j^2}]$ of each $\rho_j(\mathbf{X})^2$. If $\rho^2_{\min_j} \le \underline{\rho_j^2}$ and $\overline{\rho_j^2} \le \rho^2_{\max_j}$ for all legs, we can guarantee that W is included in the robot workspace. If $\rho^2_{\max_j} < \underline{\rho_j^2}$ or $\overline{\rho_j^2} < \rho^2_{\min_j}$ for at least one leg, W is not included in the workspace as at least one leg has a length that violates the constraints. If $\underline{\rho_j^2} \le \rho^2_{\min_j}$ or $\rho^2_{\max_j} \le \overline{\rho_j^2}$ for at least one leg, we cannot conclude: the constraints are

[1] An on-line system solver is available at http://www-sop.inria.fr/coprin/index_english.html, together with tutorials on the use of interval analysis and our interval analysis library ALIAS.

not satisfied but they may be not violated as the inequalities may be due only to the overestimation of interval analysis. In that case, we will split W into two parts W_1, W_2 whose union is W and apply them to the same process. The splitting will be repeated until either we find a part of the workspace W_n for which the constraints are violated or we determine that the constraints are satisfied for all parts that have been derived from W. Basically, this algorithm may be seen as a lazy optimisation procedure to which an a-priori information on the optimum is given and it examines a domain for possible improvement of the optimum only if needed.

Assume that we have small uncertainties in the geometry of a robot, *e.g.* the coordinates of the anchor points of the legs on the base and platform are known only upon some manufacturing tolerances $[-\varepsilon, \varepsilon]$. As the leg lengths are functions of the coordinates of the anchor points, we may introduce them as intervals: a direct consequence is that the leg lengths at a given pose will have intervals. But an interval evaluation of the leg lengths can still be computed for a sub-part of W and the algorithm may still be used. In other words, we can guarantee that W is included in the workspace of the *real* robot whatever its geometry is.

Such algorithm can be used for any type of parallel robots and may be extended to deal with other workspace constraints (*e.g.* limitation on the passive joints motion) or various geometries for W [1.62]. To further improve the performances of the structure (or to maintain it in the long term), *calibration* should also be considered [1.63]–[1.66].

1.5.4.3 Interval Analysis and Allowed Regions

Assume now that we have designed a checking algorithm $C(\mathbf{X}, P)$ that verifies if some requirements **F** are satisfied for all poses **X** in a workspace W, being the given bound on the design parameters P. This algorithm returns true if all elements in **F** are satisfied, false if at least one element of **F** is violated, and possibly *cannot assert* if the requirements cannot be all stated (for example, the ranges for the design parameters are too large).

A simple generic algorithm for determining the allowed region can be designed based on the branch-and-bound principle: a *box* is a set of ranges for the design parameters, and the algorithm possesses a list of L boxes indexed by the integer i. At the beginning, the algorithm L only has one box for the search domain $i = 1$. A box will be valid only if the width of the range for design parameter s_j is larger than ε_j, $j \in [1, m]$.

1. Verify if the requirements **F** are satisfied by the box i of L, using C;
   - if yes, store the box as an allowed region
   - if no, $i = i + 1$ and return to 1
2. If one of the requirements in **F** cannot be asserted, check the width of each range in the box;
   - if all widths are lower than ε_j, $i = i + 1$ and go to step 1
   - otherwise, select the design parameter that has the largest range in the box, split the box into 2 boxes according to this parameter and store them in L, then $i = i + 1$ and return to 1

The algorithm stops when i is larger than the number of boxes in L, *i.e.* all boxes have been processed.

Such an algorithm will, in general, satisfy property $\boldsymbol{H}_1$. Indeed, the algorithm provides an approximation to the allowed region as a set of boxes, the boxes getting smaller near the boundary of the real allowed region. We usually set ε_j to be twice the tolerance on the design parameter j. As a result, we get boxes with range $[a_j, b_j]$ for the design parameter j, and we may choose any value in $[a_j + \varepsilon_j, b_j - \varepsilon_j]$ as the nominal value for this parameter, so that we can ensure that the real value for this parameter is indeed included in $[a_j, b_j]$. As for property $\boldsymbol{H}_2$, all will depend on the checking algorithm C.

For property $\boldsymbol{H}_3$, we have mentioned in the algorithm description that the domain for $\mathbf{X}$ was W. We may also choose a sub-domain of W, which may be only a set of poses or a collection of small domains around a specific pose in order to decrease the computation time of the calculation of the allowed region. Hence, for the allowed region, the requirements $\mathbf{F}$ will be satisfied only on the chosen sub-domain of W.

An interesting point, however, is that we may directly deduce from the design algorithm a verification algorithm that will check if the requirements $\mathbf{F}$ are satisfied for a given robot, possibly with small uncertainties on its design parameters. Indeed, for the design algorithm, we start with large ranges for the design parameters P and relatively small ranges for $\mathbf{X}$.

On the other hand, for verifying a robot, we will have small ranges for P and large ranges for $\mathbf{X}$ (to cover W). Hence, the verification algorithm is simply obtained from the design algorithm by exchanging the role of P and $\mathbf{X}$: the boxes will be a set of ranges for $\mathbf{X}$ and the bisection process operates on the pose parameters. As for property $\boldsymbol{H}_4$, it is an intrinsic feature of branch-and-bound algorithm.

There is also an additional advantage of the presented algorithm. The calculation of the intersection of the allowed regions for various specifications may be done easily, using two possible approaches:

- The result of the algorithms is a set of boxes and computing the intersection of two such sets is easy;
- Alternatively, we may use an incremental approach. Assume that it is necessary to calculate the allowed regions for a set of n requirements $\{F_1, \ldots, F_n\}$. A possibility for calculating the allowed region of requirement F_k, $k > 1$, is to initialise the list L not with the search domain but with the list of boxes obtained when calculating the allowed region of the requirement S_{k-1}. Hence, we start with the calculation of the allowed region for F_1 with the full search domain. Then, the result is used for the calculation for F_2: the resulting boxes will be the values of the design parameters such that both F_1, F_2 are verified. Consequently, there is no need to compute the intersection.

We have implemented such algorithms for managing the workspace, accuracy, and statics requirements for 6-UPS, 6-PUS robots [1.67] with 6 design parameters: base and platform radii, angles between adjacent anchor points on the base and platform, minimal and maximal leg length (for the 6-UPS), length of the leg and stroke of the actuators (for the 6-PUS robot).

- 6-UPS: R_1, r_1, α, β, the lowest leg length and the stroke (6 design parameters)
- 6-PUS: R_1, r_1, α, β, the fixed length of the leg and the stroke (6 design parameters)

Figure 1.2 shows a partial view of the result of a design example (as the parameter space is of dimension 6, we cannot fully present the result in a drawing). This figure shows the possible values of the base and platform radii and of the angle between adjacent anchor points on the base. The management of accuracy analysis is interesting as it does not involve a closed-form of the requirement. Indeed, the problem is to determine the design parameters so that the positioning errors $\Delta\mathbf{X}$ are lower than a given threshold. Among the design parameters, we have $\Delta\boldsymbol{\Theta}$ that are the sensor errors. They are related by Equation (1.1) but unfortunately it is not possible to express directly $\Delta\mathbf{X}$ as a function of $\Delta\boldsymbol{\Theta}$ and the other design parameters. But Equation (1.1) may be considered as a linear interval system (*i.e.* a linear system that involves an interval matrix and interval vectors) and solving such type of system (*i.e.* finding bounds for the solution in $\Delta\mathbf{X}$) is a classical issue in interval analysis [1.68]. Hence, although we do not have an explicit form for the requirement but only an implicit one, it does not forbid designing a checking algorithm *C*.

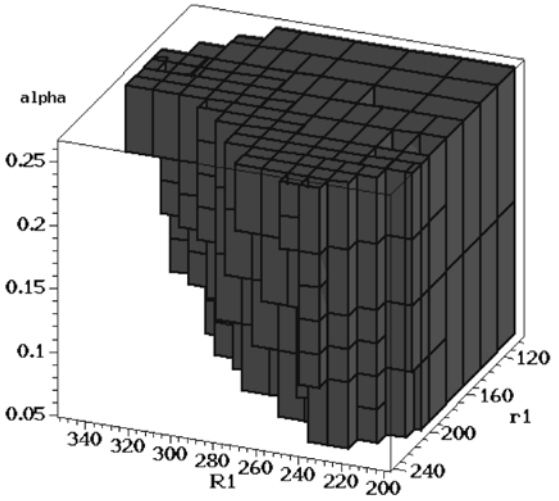

Figure 1.2. Possible values of the base and platform radii and the angle between adjacent anchor points on the base for a workspace requirement

It must be mentioned that although the principle of interval analysis may seem to be quite simple, numerous heuristics and mathematical works should be used to get an efficient algorithm. Hence, the help of interval analysis experts is necessary before implementing the design algorithms. After running the design algorithms for the various requirements and having computed the intersection of the allowed regions we will get a good approximation of all possible design parameter values. But such result cannot be presented as is to the end-user.

1.5.5 Search for Appropriate Robots

A designer cannot propose to the end-user an infinite set of design solutions, and it is, therefore, necessary to select a limited number of solutions that will be presented to the end-user. Furthermore, some allowed regions may have been computed with relaxed versions of the constraints because of the complexity of the constraints. Selecting design solutions (with possibly small uncertainty on their geometry) will allow for checking whether they satisfy the full constraints.

The chosen design solutions will first satisfy the imperative requirements and, ideally, should provide the largest possible panel of compromise between the requirements. For example, we may have imperative requirements on the workspace (the robot must include a given domain W) and on the accuracy (the positioning errors should be lower than the thresholds being given to the sensor errors). Typical compromises are to present design solutions satisfying the imperative requirements, one having the largest workspace volume and the other one presenting the lowest maximal positioning error for each component of $\mathbf{X}$.

When more requirements have to be considered, it is more difficult to find all possible compromise and we just sample at regular intervals the intersection of the allowed regions, each node of the sampling representing a unique robot geometry.

If some allowed regions have been obtained with a relaxation of the constraints, we will check that the design solution obtained for a node will satisfy the full constraints. Whenever possible, this verification will be performed by assigning a range for the design parameters, whose width will be the corresponding tolerances; if a node is validated as a design solution then the real robot obtained for the node, with the stated tolerances, will also satisfy the specifications.

Primary and secondary requirements are also calculated at each node. After verifying all nodes and retaining the solutions that satisfy the imperative, primary and secondary requirements, we will manually select the solutions representing the most different compromises. For example, if the stiffness k_x and k_y are secondary requirements, we will select the one with the largest k_x, the one with the largest k_y, and the one having a mean value for k_x and k_y, as design solutions.

1.5.6 Design Examples

The proposed approach has been used to manage entirely or partially the design of complex machines. Figure 1.3 presents one of the Gough platforms that have been studied for the European Synchrotron Radiation Facility (ESRF) in Grenoble. Imperative requirements were the load (up to 2.5 tons) and the accuracy (absolute accuracy should be better than 1 μm).

Figure 1.3 shows the milling machine designed for Constructions Mécanique des Vosges, who use it as a milling head for the high-speed manufacturing of huge aeronautical parts. Like any machine tools, imperative requirements were accuracy, stiffness and working load.

We also use this design methodology for robot with less than 6 DOF. For example, it was used for the design of our micro-robot for endoscopy MIPS, shown in Figure 1.4, which has 3 DOF. Imperative requirements here were overall size (the

robot is located at the end of a 1 cm diameter endoscope), accuracy and sustainable forces/torques.

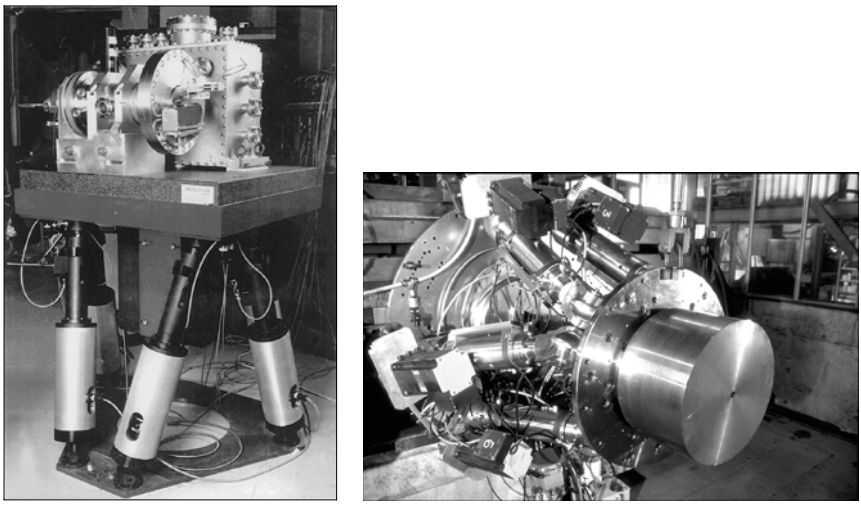

Figure 1.3. A robot designed for the European Synchrotron Radiation Facility (left), and a milling head designed for Constructions Mécanique des Vosges (right)

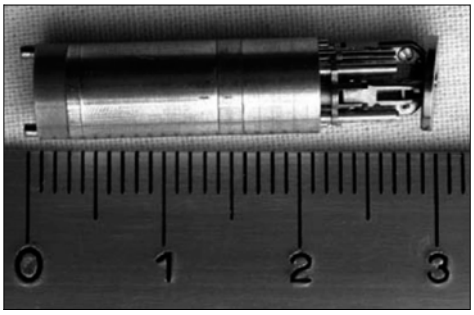

Figure 1.4. The MIPS micro-robot design for endoscopic surgery

1.6 Other Design Approaches

Until now we have only considered performance as the main issue for the design problem. However in some cases, other criteria may be very important although performance will always be considered.

1.6.1 Design for Reliability

Parallel robots have numerous joints and may be used in applications for which reliability is critical (*e.g.* space and medical tasks). Design for reliability is for the

purpose to reduce the influence of components failure on the robot performances. Although a few works have addressed this problem [1.69]–[1.71], this is a very complex issue that will require large theoretical and experimental studies.

1.6.2 Design for Control

In the design for control approach, the purpose is to determine the design of a system to simplify its control. This is an approach that may be used for parallel robots as well [1.72][1.73]. For example, it may be thought that an appropriate design may help to simplify the dynamic modelling, reducing its computation time and consequently allowing faster operating velocities. This issue is still an open problem for parallel robots.

1.7 Conclusions

Design of parallel robots is a crucial issue as their performances are very sensitive to the choice of the mechanical architecture and of the dimensioning. Design is also a complex problem as the number of design variables is relatively large while the closure constraints that are specific to closed-loop mechanisms complicate the modelling and optimisation. Research studies on this topic are only beginning and are far from having reached the level of accomplishment that has been reached for serial structures.

References

[1.1] Cauchy, A., 1813, “Deuxième mémoire sur les polygones et les polyèdres,” *Journal de l'École Polytechnique*, pp. 87–98.

[1.2] Lebesgue, H., 1967, “Octaèdre articulé de Bricard,” *L'enseignement mathématique*, (13), pp. 150–160.

[1.3] Gough, V.E., 1956–1957, “Contribution to discussion of papers on research in automobile stability, control and tire performance,” *Proceedings of Automobile Division of IMechE*.

[1.4] Gough, V.E. and Whitehall, S.G., 1962, “Universal tire test machine,” In *Proceedings of the 9th International Technical Congress F.I.S.I.T.A.*, London, **117**, pp. 117–135.

[1.5] Cappel, K.L., 1967, *Motion Simulator*, United States Patent n° 3,295,224, The Franklin Institute.

[1.6] Stewart, D., 1965, “A platform with 6 degrees of freedom,” *Proceedings of the Institution of Mechanical Engineers*, **180**(Part 1, 15), pp. 371–386.

[1.7] McCallion, H. and Pham, D.T., 1979, “The analysis of a six degrees of freedom work station for mechanized assembly,” In *5th IFToMM World Congress on the Theory of Machines and Mechanisms*, Montréal, Canada, pp. 611–616.

[1.8] Reboulet, C. and Robert, A., 1985, “Hybrid control of a manipulator with an active compliant wrist,” In *3rd ISRR*, Gouvieux, France, pp. 76–80.

[1.9] Ma, O. and Angeles, J., 1991, “Optimum architecture design of platform manipulator,” In *ICAR*, pp. 1131–1135.

[1.10] Merlet, J.-P., 2006, “Jacobian, manipulability, condition number, and accuracy of parallel robots,” *ASME Journal of Mechanical Design*, **128**(1), pp. 199–206.

[1.11] Gosselin, C., 1988, *Kinematic Analysis Optimization and Programming of Parallel Robotic Manipulators*, PhD Thesis, McGill University, Montréal, Canada.

[1.12] Nawratil, G., 2006, "The control number as index for Stewart-Gough platforms," In *ARK*, Ljubljana, Slovenia, pp. 15–22.

[1.13] Voglewede, P.A. and Ebert-Uphoff, I., 2004, "Measuring "closeness" to singularities for parallel manipulators," In *IEEE International Conference on Robotics and Automation*, New Orleans, USA, pp. 4539–4544.

[1.14] Wolf, A. and Shoham, M., 2003, "Investigation of parallel manipulators using linear complex approximation," *ASME Journal of Mechanical Design*, **125**(3), pp. 564–572.

[1.15] Chang, W.-T., Lin, C.-C. and Lee, J.-J., 2003, "Force transmissibility performance of parallel manipulators," *Journal of Robotic Systems*, **20**(11), pp. 659–670.

[1.16] Funabashi, H. and Takeda, Y., 1995, "Determination of singular points and their vicinity in parallel manipulators based on the transmission index," In *9th IFToMM World Congress on the Theory of Machines and Mechanisms*, Milan, Italy, pp. 1977–1981.

[1.17] Krut, S., Company, O. and Pierrot, F., 2004, "Force performance indexes for parallel mechanisms with actuation redundancy, especially for parallel wire-driven manipulators," In *IEEE International Conference on Intelligent Robots and Systems (IROS)*, Sendai, Japan.

[1.18] Zhang, D., Xu, Z., Mechefske, C.M. and Xi, F., 2004, "Optimum design of parallel kinematic toolheads with genetic algorithm," *Robotica*, **22**(1), pp. 77–84.

[1.19] Carretero, J.A. and Pond, G.T., 2006, "Quantitative dexterous workspace comparison," In *ARK*, Ljubljana, Slovenia, pp. 297–306.

[1.20] Merlet, J.-P., 1998, "Efficient estimation of the extremal articular forces of a parallel manipulator in a translation workspace," In IEEE International Conference on Robotics and Automation, Louvain, Belgium, pp. 1982–1987.

[1.21] Alizade, R.I. and Bayram, C., 2004, "Structural synthesis of parallel manipulators," *Mechanism and Machine Theory*, **39**(8), pp. 857–870.

[1.22] Jin, Q. and Yang, T.-L., 2004, "Theory for topology synthesis of parallel manipulators and its application to three-dimension-translation parallel manipulators," *ASME Journal of Mechanical Design*, **126**(1), pp. 625–639.

[1.23] Earl, C.F. and Rooney, J., 1983, "Some kinematics structures for robot manipulator designs," *Journal of Mechanisms, Transmissions and Automation in Design*, **105**(1), pp. 15–22.

[1.24] Angeles, J., 2005, "The degree of freedom of parallel robots: a group-theoretic approach," In *IEEE International Conference on Robotics and Automation*, Barcelona, Spain, pp. 1017–1024.

[1.25] Hervé, J.M., 2004, "Parallel mechanisms with pseudo-planar motion generators," In *ARK*, pp. 431–440.

[1.26] Gogu, G., 2005, "Mobility of mechanisms: a critical review," *Mechanism and Machine Theory*, **40**(10), pp. 1068–1097.

[1.27] Karouia, M. and Hervé, J.M., 2002, "A family of novel orientational 3-dof parallel robots," In *14th RoManSy*, Udine, Italy, pp. 359–368.

[1.28] Carricato, M., 2005, "Fully isotropic four-degrees-of-freedom parallel mechanisms for Schoenflies motion," *International Journal of Robotics Research*, **24**(5), pp. 397–414.

[1.29] Fang, Y. and Tsai, L.-W., 2002, "Structure synthesis of a class of 4-dof and 5-dof parallel manipulators with identical limb structures," *International Journal of Robotics Research*, **21**(9), pp. 799–810.

[1.30] Frisoli, A. and others, 2000, "Synthesis by screw algebra of translating in-parallel actuated mechanisms," In *ARK*, Piran, Slovenia.

[1.31] Gao, F., Li, W., Zhao, X., Jin, Z. and Zhao, H., 2002, "New kinematic structures for 2-, 3-, 4- and 5-dof parallel manipulator designs," *Mechanism and Machine Theory*, **37**(11), pp. 1395–1411.

[1.32] Kong, X. and Gosselin, C.M., 2007, *Type Synthesis of Parallel Mechanisms*, Springer Tracts in Advanced Robotics, Heidelberg, Germany.

[1.33] Rao, A.C., 1997, "Platform-type planar robots: topology-based selection for rigidity and workspace," *Journal of Robotic Systems*, **14**(5), pp. 355–364.

[1.34] Merlet, J.-P., Perng, M.-W. and Daney, D., 2000, "Optimal trajectory planning of a 5-axis machine tool based on a 6-axis parallel manipulator," In *ARK*, Piran, Slovenia, pp. 315–322.

[1.35] Vischer, P., 1996, *Improving the accuracy of parallel robots*, PhD Thesis, EPFL, Lausanne, Switzerland.

[1.36] Zhang, D., Wang, L. and Lang, S.Y.T., 2005, "Parallel kinematic machines: design, analysis and simulation in an integrated virtual environment," *ASME Journal of Mechanical Design*, **127**(4), pp. 580–588.

[1.37] Clavel, R., 1991, *Conception d'un Robot Parallèle Rapide à 4 Degrés de Liberté*, PhD Thesis, EPFL, Lausanne, Switzerland, n° 925.

[1.38] Bhattacharya, S., Hatwal, H. and Ghosh, A., 1995, "On the optimum design of a Stewart platform type parallel manipulators," *Robotica*, **13**(2), pp. 133–140.

[1.39] Badescu, M. and Mavroidis, C., 2004, "Workspace optimization of 3-legged UPU and UPS parallel platforms with joint constraints," *ASME Journal of Mechanical Design*, **126**(2), pp. 291–300.

[1.40] Hong, K.-S., 2003, "Kinematic optimal design of a new parallel-type rolling mill: paramill," *Advanced Robotics*, **17**(9), pp. 837–862.

[1.41] Liu, X.-J., Wang, J. and Zheng, H., 2003, "Workspace atlases for the computer aided design of the Delta robot," *Proceedings of IMechE, Part C: Journal of Mechanical Engineering Science*, **217**(8), pp. 861–869.

[1.42] Masuda, T. and others, 2002, "Mechanism configuration evaluation of a linear-actuated parallel mechanism using manipulability," In *IEEE International Conference on Robotics and Automation*, Washington D.C., USA, pp. 489–495.

[1.43] Erdman, A.G., 1993, *Modern Kinematics*, John Wiley & Sons, Inc., New York, USA.

[1.44] Das, I. and Dennis, J.E., 1997, "A closer look at drawbacks of minimizing weighted sums of objectives for Pareto set generation in multicriteria optimization problem," *Structural Optimization*, **14**, pp. 63–69.

[1.45] Du Plessis, L.J. and Snyman, J.A., 2006, "Determination of optimum geometries for a planar reconfigurable machining platform using the LFOPC optimization algorithm," *Mechanism and Machine Theory*, **41**(3), pp. 307–333.

[1.46] Huang, T., Li, M., Li, Z., Chetwynd, D.G. and Whitehouse, D.J., 2004, "Optimal kinematic design of 2-dof parallel manipulators with well-shaped workspace bounded by a specific conditioning index," *IEEE Transactions on Robotics and Automation*, **20**(3), pp. 538–543.

[1.47] Liu, X.-J., Wang, J. and Pritschow, G., 2006, "On the optimal kinematic design of the PRRRP 2-dof parallel mechanism," *Mechanism and Machine Theory*, **41**(9), pp. 1111–1130.

[1.48] Chen, W. and others, 1999, "Quality utility – a compromise programming approach to robust design," *ASME Journal of Mechanical Design*, **121**(2), pp. 179–187.

[1.49] Chen, W., Sahai, A., Messac, A. and Sundararaj, G.J., 2000, "Exploration of the effectiveness of physical programming in robust design," *ASME Journal of Mechanical Design*, **122**(2), pp. 155–163.

[1.50] Affi, Z., Romdhane, L. and Maalej, A., 2004, "Dimensional synthesis of a 3-translational-dof in-parallel manipulator for a desired workspace," *European Journal of Mechanics A/Solids*, **23**(2), pp. 311–324.

[1.51] Gao, F., Liu, X.-J. and Chen, X., 2001, "The relation ships between the shapes of the workspaces and the link lengths of 3-DOF symmetrical planar parallel manipulators," *Mechanism and Machine Theory*, **36**(2), pp. 205–220.

[1.52] Kosinska, A., Galicki, M. and Kedzior, K., 2003, "Design of parameters of parallel manipulators for a specified workspace," *Robotica*, **21**(5), pp. 575–579.

[1.53] Miller, K., 2002, "Maximization of workspace volume of 3-DOF spatial parallel manipulators," *ASME Journal of Mechanical Design*, **124**(2), pp. 347–350.

[1.54] Chablat, D. and Wenger, P., 2003, "Architecture optimization of a 3-dof translational parallel mechanism for machining applications, the Orthoglide," *IEEE Transactions on Robotics and Automation*, **19**(3), pp. 403–410.

[1.55] Huang, T., Jiang, B. and Whitehouse, D.J., 2000, "Determination of the carriage stroke of 6-PSS parallel manipulators having the specific orientation capability in a prescribed workspace," In *IEEE International Conference on Robotics and Automation*, San Francisco, USA, pp. 2382–2385.

[1.56] Arsenault, M. and Boudreau, R., 2004, "The synthesis of three-degree-of-freedom planar parallel mechanisms with revolute joints (3-RRR) for an optimal singularity-free workspace," *Journal of Robotic Systems*, **21**(5), pp. 259–274.

[1.57] Simaan, N. and Shoham, M., 2003, "Stiffness synthesis of a variable geometry six-degree-of-freedom double planar parallel robot," *International Journal of Robotics Research*, **22**(9), pp. 757–775.

[1.58] Jafari, F. and McInroy, J.E., 2003, "Orthogonal Gough-Stewart platforms for micromanipulation," *IEEE Transactions on Robotics and Automation*, **19**(4), pp. 595–603.

[1.59] Merlet, J-P., 1997, "Designing a parallel manipulator for a specific workspace," *International Journal of Robotics Research*, **16**(4), pp. 545–556.

[1.60] Hansen, E., 2004, *Global Optimization Using Interval Analysis*, Marcel Dekker.

[1.61] Jaulin, L., Kieffer, M., Didrit, O. and Walter, E., 2001, *Applied Interval Analysis*, Springer-Verlag.

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

[1.63] Chiu, Y.J. and Perng, M.-H., 2004, "Self-calibration of a general hexapod manipulator with enhanced precision in 5-dof motions," *Mechanism and Machine Theory*, **39**(1), pp. 1–23.

[1.64] Daney, D., Andreff, N., Chabert, G. and Papegay, Y., 2006, "Interval method for calibration of parallel robots: a vision-based experimentation," *Mechanism and Machine Theory*, **41**(8), pp. 929–944.

[1.65] Daney, D., Papegay, Y. and Madeline, B., 2005, "Choosing measurement poses for robot calibration with the local convergence method and Tabu search," *International Journal of Robotics Research*, **24**(6), pp. 501–518.

[1.66] Khalil, W. and Besnard, S., 2001, "Identificable parameters for the geometric calibration of parallel robots," *Archive of Control Sciences*, **11**(3–4), pp. 263–277.

[1.67] Fang, H. and Merlet, J.-P., 2005, "Multi-criteria optimal design of parallel manipulators based on interval analysis," *Mechanism and Machine Theory*, **40**(2), pp. 151–171.

[1.68] Merlet, J.-P. and Daney, D., 2005, "Dimensional synthesis of parallel robots with a guaranteed given accuracy over a specific workspace," In *IEEE International Conference on Robotics and Automation*, Barcelona, Spain.

[1.69] Hassan, M. and Notash, L., 2005, "Design modification of parallel manipulators for optimum fault tolerance to joint jam," *Mechanism and Machine Theory*, **40**(5), pp. 559–577.

[1.70] Notash, L. and Huang, L., 2003, "On the design of fault tolerant parallel manipulators," *Mechanism and Machine Theory*, **38**(1), pp. 85–101.

[1.71] Ukidve, C.S., McInroy, J.E. and Jafari, F., 2006, "Orthogonal Gough-Stewart platforms with optimal fault tolerant manipulability," In *IEEE International Conference on Robotics and Automation*, Orlando, USA, pp. 3801–3806.

[1.72] Li, Q., 2006, "Experimental validation on the integrated design and control of a parallel robot," *Robotica*, **24**(2), pp. 173–181.

[1.73] Ouyang, P.R., Zhang, W.J. and Wu, F.X., 2002, "Nonlinear PD control for trajectory tracking with consideration of the design for control methodology," In IEEE International Conference on Robotics and Automation, Washington D.C., USA, pp. 4126–4131.