**The need for a systematic methodology**
**for the evaluation and optimal design of parallel manipulators**

J-P. Merlet
INRIA Sophia-Antipolis, France

*Abstract*: It has been theoretically and experimentally demonstrated in the past that PKS may reach a very high level of performance in term of accuracy, rigidity, load carrying capabilities etc. But when designing a PKS with strict requirements we are confronted with two main difficulties: finding the right topology for the PKS and finding the right geometry for the chosen topology. Both problems are highly complex: it is difficult to compare different topologies without having to specify the geometry and performances of PKS are highly dependent of the geometry of the mechanism. Hence we believe that it is necessary to develop a design methodology for PKS with two components: topological synthesis and dimensional synthesis. In this paper we will present the necessary requirements for the dimensional synthesis methodology, analyze their complexity and outline a possible generic methodology which will be the result of a collaborative work between mathematicians, researchers in the field of mechanism theory and end-users.

# 1 Introduction

Historically, closed-chain structures have attracted the interest mostly of mathematicians as they offer interesting problems. Some theoretical problems linked to this type of structure were mentioned as early as 1645 by Christopher Wren, then in 1813 by Cauchy [4] and in 1867 by Lebesgue [6]. One of the main theoretical problems in this field, called the *spherical motion problem*, related to what is now called singularity analysis, was the central point of a competition called *Le Prix Vaillant*, that took place in France in the 1900's and was organized by the Académie des Sciences. The prize was won on equal terms by Borel [1] and Bricard [3].

But clearly at this time the technology was not able to deal with any practical applications of this type of structure. Practical application were then considered in the 70's for flight simulator (a very specific area where mostly acceleration are of interest) and in the 80's for robotics application (with an interest in a larger panel of performances).

Starting in the 90's PKS 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-tool. Among these applications, some of them were not as successful as expected. The clearest illustration of this fact is the use of PKS in the machine-tool industry. Although the first presentation of such PKS dates from 1994 with the Variax of Giddings, we have still to see PKS in current use for such application. I see three main reasons for this failure:

1. with very few exceptions there is no interaction with the laboratories having worked in this field for many years and the developers in the industry; hence problems that were familiar for researchers are completely overlooked by the developers, while researchers may miss important points for a specific application.

2. the inherent non-linearity of PKS and its consequences on the design and on the control of such structure is highly unfamiliar to people working in the field of machine-tool, accustomed to a linear world,

3. developers in the industry focus first on the development of the basic mechanical elements of a PKS, such as ball-and-socket joints. Although this work was necessary as these components in the right size were not commercially available, this type of development is only *local*,

missing the point that these elements are part of a *global system* that has to be considered as such.

I must also be noted that these failures have a negative influence on the development of PKS, as they comfort a trend that states that these structures are too complex to work in practice (a trend that completely ignore past success stories in this field).

In the following I will try to emphasize the difficulties with which we are confronted to build efficient PKS and outline a possible approach to solve these difficulties.

## 2 Topology synthesis

Although I will focus on the dimensional synthesis problem, it is easy to show that the topology synthesis problem, i.e. finding the most appropriate mechanical architecture for a given task, is difficult for PKS.

Assume for a simplicity that for a given task we have to design a mechanism with 6 d.o.f. and that the comparison between different mechanical architectures has to be done based only on the volume of the workspace that can be reached by the end-effector. A further assumption is that actuated joints will only be of the prismatic (P) or revolute (R) type.

For classical serial structure, only the first three joints have an influence on the location of the end-effector. All the possible architectures will be obtained by considering all the possible set of three elements, each element being either P or R. For example a Cartesian robot is defined as the set PPP, while the spherical robot is RRR. Now affect a standard length $L$ to each element of the robot: revolute joints will connect two bodies of length $L$, prismatic joint have a retracted length of $L$ and an extended length of $2L$. Under that assumption the workspace volume of a PPP robot is $L^3$ while the RRR workspace volume is roughly $85L^3$, *for any value of* $L$. Hence it is clear that the RRR structure is more appropriate than the PPP.

Now let us introduce PKS, namely a classical Gough platform and an Hexa robot [12]. A first problem is that for these type of structure the translation ability is not decoupled from the orientation. Then according to our hypothesis we will assume that the radii of the base and platform is identical and equal to $L$. This is clearly a very restrictive assumption, which will have a large effect on the workspace volume. Finally even for a given orientation we do not know what will be the workspace volume of both PKS as a function of $L$. It seems only that for a given geometry of Gough platform the workspace volume is $k\rho^3$ where $\rho$ is the extension of the leg [8], where $k$ is a factor that depend on the geometry of the robot (hence under our assumption the workspace volume will be written as $k(L)L^3$). A similar result has never been established for the Hexa robot but imagine that in that case the workspace may be written as $g(L)L^3$. Comparison of the two PKS in term of workspace volume based on the previous formula may lead only to the conclusion that for some ranges on $L$ the Gough platform has a larger workspace than the Hexa, the opposite being true for other ranges for $L$.

Hence at this time we are only able to compare the generic workspace volume of 2 serial structures but not to compare either a serial and a parallel structure or 2 PKS.

Topology synthesis for PKS is a much more complex problem than for serial structure and probably cannot be completely disconnected from the dimensional synthesis problem.

## 3 Dimensional synthesis

Finding the dimensions of a given mechanism so that it is optimal with respect to some requirements is a very old problem in mechanism theory. Before describing the existing methods let us examine what are the requirements that may be imposed on PKS and what is their complexity.

### 3.1 Requirements

The COPRIN project of INRIA has a lot of practical experience in the optimal design of PKS, which has been gained from the development of our own prototype and from several industrial contracts. We have been dealing especially with:

- fine positioning devices for heavy loads (with the European Synchrotron Radiation Facility, the Institut Laue Langevin, Alcatel),
- machine-tools (with Constructions Mécaniques des Vosges),
- medical robots.

Very early we have established an evaluation form for the design of Gough platform type PKS that both allows the end-user to describe his requirements (either as trends or with numerical values) and enable the designer to get all the necessary information to perform the design study.

The end-user may provide information and requirements that may be classified as:

- *kinematics*: workspace, accuracy, maximal motion of the passive joints, dexterity,
- *statics*: load on the platform, stiffness of the robot,
- *dynamics*: maximal velocity and acceleration of the actuator and of the platform, inertia and center of mass,
- *geometrical*: overall size of the robot, of the mechanical components,
- *technological*: overall information on the actuator, on the sensors and on the passive joints. Indeed the context of the application may impose the use of restricted classes of such components.

Note that most of the time the requirements provided by the end-user will only be subset of the requirements used by the designer. For example the end-user may provide only requirements on the workspace and on the load carrying ability but the designer will also consider singularities. Among the list of requirements, workspace and accuracy are almost always provided and that some of them are *strict requirements* that cannot be relaxed.

The end-user has also to classify his requirements according to their importance: this is crucial as in some case we have to relax some requirements in order to be able to satisfy some other requirements. It must also be mentioned that some requirements may involve a fixed value (e.g. the accuracy of the positioning of the platform for a unit value of the sensor error must not greater than a threshold) and will be called *fixed value requirements*. On the other hand, we may have *a maxima requirements* (e.g. the accuracy of the positioning of the platform for a unit value of the sensor error must be as high as possible).

First of all it must be noticed that for PKS most of these performances are *pose dependent*. For example, the workspace of the end-effector is dependent upon its orientation, while the accuracy is dependent both upon the orientation and the location of the end-effector. This dependency is usually quite complex: for example the accuracy $\Delta\mathbf{X}$ of the positioning of the platform is related to the accuracy $\Delta\rho$ of the sensor by:

$$\Delta\rho = J^{-1}(X)\Delta\mathbf{X}$$

The inverse Jacobian matrix $J^{-1}$ has a relatively simple analytical form, but establishing the positioning accuracy of the platform as a function of the sensor accuracy will require the use of $J$, which is highly complex.

Most of these requirements are of the *worst case* type with respect to the workspace: as the performances are pose dependent, the limits imposed on the requirements have to be considered for the whole workspace. For example an accuracy requirement $\Delta X_d$ indicates that the positioning error must not exceed $\Delta X_d$ *over the whole workspace of the robot*.

But the designer may have also to deal with other cases. Imagine for example that two robots A and B with different geometries have equivalent worst case accuracy. Clearly this does not imply that they are equivalent. Indeed the *average value* of the positioning error over the whole workspace has to be considered or even the *best case* (when some crucial part of the task requires a high accuracy). Note that the best and worst case problems are obtained by solving a constrained optimization problem while the average value problem is somewhat different.

Finally it must be emphasized that all the requirements in the above list are highly sensitive to the geometry of the robot. Such sensitivity is the first reason for the failure of some prototype of PKS which have been designed using a local approach instead of a global one, the second one being that some properties of PKS have been overlooked. For example changing the radius of the platform of a Gough platform by 10% may modify the worst case stiffness by 700%. Clearly such ratio imply that a robot with a poor topology but optimally designed will present largely better performances than a robot with an appropriate topology but poorly designed. Hence **dimensional synthesis is crucial when designing a PKS**.

## 3.2 Workspace requirements

As mentioned previously, the above requirements have to be verified over the workspace of the robot. This workspace may be defined in various terms:

1. a workspace defined with respect to a global reference frame

2. the whole workspace of the robot. For example, for a Gough platform, this workspace may be defined as the set of poses that the robot can reach with the leg lengths $\rho$ satisfying the inequalities $\rho_{min} \leq \rho \leq \rho_{max}$ where $\rho_{min}, \rho_{max}$ are given constants. A general definition will be all the reachable poses such that $n$ inequality constraints $F_i(X, \rho) \leq 0$ $(i = 1, \ldots, n)$ are satisfied.

3. a workspace, where the $z$ component specification is defined relatively to some unknown quantity $z_d$. For example the $z$ motion ability may be specified as $\pm$ 50 mm relatively to some unknown $z_d$.

These three different possibilities may co-exist for a given design problem. For example, the accuracy requirement may be defined for a workspace of type 1, while singularity analysis has to be performed in the type 2 workspace. For the type 3 workspace we have to include $z_d$ as a design parameter.

## 3.3 Design methodology and performance verification

The most well known design methodology is the cost-function approach [5]. To each design requirement $j$ is associated a numerical index $I_j$ that is minimal for the best robot. The cost function $\mathcal{C}$ is defined as:

$$\mathcal{C} = \sum w_j I_j,$$

where the $w_j$ are weight associated to the $I_j$. In some sense, the cost function is an indicator of the global behavior of the mechanism with respect to the requirements. As $\mathcal{C}$ is clearly a function of the set of design parameters $\mathcal{P}$, a numerical procedure is used to find the value of the design parameters that minimize $\mathcal{C}$. This approach has several drawbacks:

- the result is heavily dependent upon the weights that are used in the cost-function, and there is no automatic way to find the right weights,
- defining the index $I$ is not always an easy task, for example if we have constraints on the shape of the workspace. Furthermore, some of these index are even very difficult to estimate; for example, some authors have mentioned the use of a *global conditioning index (CGI)* defined as the average value of the condition number over the workspace of the robot. The condition number itself is obtained as the ratio of the smallest root of a $n$ dimensional polynomial (where $n$ is the number of d.o.f. of the robot) over the largest root of this polynomial. Hence in general the condition number have not an analytical form and consequently computing exactly the CGI is a very difficult task.
- introducing strict requirements in the minimization is difficult, and in any case computer intensive,
- as for any optimization problem, it is difficult to guarantee that the global extremum has been found.
- some of the requirements are antagonistic; for example, it is well known that dexterity is antagonistic with the workspace volume [7]; using both criterion in a weighted sum does not have any physical meaning

But the main difficulty is that the computation of the index for a given geometry must be very efficient as the minimization procedure will use these calculations extensively. Unfortunately, verifying that a given PKS satisfies a single requirement is usually a very complex task. As mentioned previously calculating the index for the worst and best case requirements is equivalent to solving a difficult constrained optimization problem.

### 3.4 Performance verification

In my opinion, any optimal design methodology will use a performance verification module that takes as its input a robot geometry and verify whether this geometry satisfy a list of requirements. Hence **the development of an efficient performance verification module is a key point for the optimal design of PKS**.

Ideally, such module should be able to

1. deal with any type of PKS, although optimized version for the most usual PKS may exist,
2. deal with almost any type of requirements, especially worst and best cases,
3. provide *guaranteed* results.

A given requirement usually defines an implicit set of constraints $\mathcal{I}$ that is only dependent upon the topology of the robot. Assume now that we have a generic tool $\mathcal{T}$ that is able to deal with any $\mathcal{I}$ as soon as $\mathcal{I}$ is expressed in a standard form (that we will call the *standard verification form* (SVF)). A generic performance verification module may reach the first two objectives if

1. we first preprocess all the $\mathcal{I}$ for each requirement to put them in the standard form, probably using a symbolic computation software,
2. we use then the generic tool $\mathcal{T}$ to verify all the requirements either in sequence or simultaneously.

In my opinion, many mathematical tools offer the possibility of designing $\mathcal{T}$ (see the Annex for one possibility). But the key point on this issue is **a collaborative work of researchers in the field of mechanism theory and mathematicians**. The first part of this effort is the development of the SVF: the researchers in mechanism theory will provide the description of all the requirements that may be of interest for PKS; the mathematicians will analyze them to obtain a very reduced set of problems to solve. For example, although dealing with very different quantities, determining the worst case accuracy of a robot and the maximal joint forces are strictly equivalent mathematical problems.

The second part of the collaborative work is to solve the reduced set of problems that will lead to $\mathcal{T}$. **A key point on this issue is the problem of the quality of the result**: the result must be *guaranteed*. This has not the same meaning as obtaining *exact* results (or even approximatively exact in the computer science signification of this term). A guarantee on the result means that we are able to determine error bounds on the result, so that a decision based on this result will ensure the satisfaction of the requirements (or in the worst case that the result cannot be calculated in a standard manner on a computer such that we are sure that the requirement is satisfied). It is therefore much less stringent that getting the exact result. Guaranteed results exclude the use of discretization methods that just sample the workspace and verify the requirement only at the sampling points.

Only guarantee is needed, as for many requirements it will not be necessary to obtain the exact result. For example, determining the accuracy of the positioning of the platform is necessary to determine the accuracy of the sensor that must be used. We first determine what will be the accuracy $\Delta X_1$ of the positioning of the platform for a unit value of the sensor error and, as the relationship between these two quantities is linear, we are able to determine what must be the sensor error $\Delta\rho_s$ so that the accuracy of the platform reaches a given value $\Delta X_d$. *In many cases only a restricted set of accuracy for the sensor will be available.* Hence the accuracy of the platform need to be determined only to the extent that it will result in a unique solution for the accuracy of the sensor. For example if the available sensor accuracy are 0.1, 0.2 and 0.5 and if $\mathcal{T}$ is able to compute a range for $\Delta X_1$ such that $\Delta\rho_s$ is in the range [0.3,0.4], we know already that we have to use the sensor with the accuracy 0.2. Hence, although we have not determined what will be the worst case accuracy, we can still guarantee that it will satisfy the requirement.

Clearly $\mathcal{T}$ must be designed so that it only guarantees the result, especially if getting a guaranteed result is less computer intensive than getting the exact result.

## 3.5 Alternative optimal design methodologies

### 3.5.1 Genetic algorithms

Assume now that an efficient performance verification module is available. This open the door to alternative design methodologies such as the use of genetic algorithms (GA). In this type of algorithm individuals have genes that represent values for the design parameters. An initial population of individuals is initially selected as parents and they are crossed-over to generate children, some of them having genes that are obtained as mutation of the genes of their parents. Each individual is evaluated with respect to the design requirements, and selection rules allow to select only the "more promising" children that will constitute the next generation.

GA's are well known optimization procedures that may be used when the function to be optimized are complex. They have been already used in the field of planar PKS [2], although the lack of an efficient performance verification has restricted their use to simple PKS. In my opinion GA may be interesting only if we have only fixed value requirements and cannot be used for a maxima requirements as they give guarantee on the result.

### 3.6 The parameter space approach

Let $m$ be the number of design parameters in $\mathcal{P}$. We define an $m$ dimensional space, the *parameter space* $\mathcal{S}$ , in which each dimension is associated to one design parameter (hence each point in the parameter space defines an unique robot geometry). The purpose of the parameter space approach is to determine the regions of $\mathcal{S}$ that include all the possible solutions of the design problem.

To reach this goal, the following approach may be used:

1. select a particular requirement $R_j$, or a relaxed version of this requirement (for example if the requirement is that the workspace of the PKS includes a specific Cartesian box we may relax the requirement by verifying only that the workspace includes the 8 corners of the box).
2. determine the region $\mathcal{S}_j$ of $\mathcal{S}$ which include all the robots satisfying $R_j$.
3. repeat the process for another requirement.
4. after completing the 3 first steps of this process we have obtained $m$ regions $\mathcal{S}_j$. If there is a solution to the optimal design problem, then it will lie in the intersection of the regions. At this step we compute this intersection $\mathcal{S}_i$.
5. at this point we have determined all the robots that satisfy a subset of the requirements. A local approach is then used to determine the solutions within $\mathcal{S}_i$ that satisfy all the requirements.

A key issue in this approach is step 2. We must develop a generic method that is able to deal with most common requirements. This method will rely on an extended version of the standard verification form, called the *standard design verification form (SDVF)*, that takes also into account the design parameters and will have basically the same structure than the performance verification module:

- transform the requirements into a SDVF,
- apply a generic tool $\overline{\mathcal{T}}$ to determine the region $\mathcal{S}_j$. Note that the generic tool $\mathcal{T}$ is a special instance of $\overline{\mathcal{T}}$ in which all the design parameters have a fixed value.

Although the problem may seem to be quite complex, we have already obtained some result in this area, especially for the workspace requirement, either by using a geometrical approach [10] or an interval analysis approach [9].

## 4 Conclusion

Optimal design can be divided into two main topics: topology synthesis and dimensional synthesis, although it is unclear if topology synthesis can be separated from dimensional synthesis for PKS. Performances of PKS are highly sensitive to both type of synthesis; hence optimal design is a crucial issue for the development of efficient PKS.

We propose to develop a generic method for the optimal design of PKS, based on the transformation of the requirement into a reduced set of generic problems that may be treated by an universal solver. The development of this generic method is a huge project and can only be the result of a collaborative work between the researchers working in this field, mathematicians interested in this type of problems, and end-users.

A further problem that has to be taken into account is control: there is a crucial need for robot controller that are able to deal efficiently with the inherent non-linearity of PKS and with its

consequence on control, on-line and off-line motion planning, .... In my opinion current controller are not very effective for PKS. But this is another story ...

**Acknowledgment**: Many proposals of this paper are the result of numerous discussions with Arnold Neumaier of Wien University, to which I am deeply indebted.

## **Annex**: Interval analysis

Interval analysis is a powerful method initially proposed by Moore [11]. Let us illustrate this method on a simple example: let $f$ be the function $x^2 - 2x$ and assume that we are looking for the solutions of $f = 0$ when $x$ is in the range $[3, 4]$. Intuitively it is easy to see that if $x$ is in [3,4], then $x^2$ is in [9,16]: this means that if $x$ has a particular value in the range [3,4], then $f(x)$ has a value in the range [9,16] (similarly $-2x$ is in the range [-8,-6]). Now consider the sum of 2 intervals $A = [\underline{a}, \overline{a}]$, $B = [\underline{b}, \overline{b}]$. It may be seen that $A + B = [\underline{a} + \underline{b}, \overline{a} + \overline{b}] = C$, which means that for any value of $x$ in $A$ and $y$ in $B$, then $x + y$ lie in $C$. In our case we will write $f([3, 4]) = [9, 16] + [-8, -6] = [1, 10]$. The resulting interval defines therefore lower and upper bound for the values of $f$: we may guarantee that for any $x$ is [3,4] $1 \leq f(x) \leq 10$. As 0 is not included in the final interval we may state that there is no zero of $f$ for $x$ in the range [3,4]. Note that the bounds provided by interval analysis are overestimated, the true range of $f(x)$ being [3,8]. However, this does not affect the validity of the conclusion.

This method works for all the classical mathematical functions such as $\sin, \cos, \sinh, \ldots$. Furthermore this method may be implemented to take into account numerical round-off errors and is therefore safe from a numerical view point.

Let us apply this method for a classical problem for PKS. Assume that we want to verify that a particular Cartesian box $\mathcal{B}_0$ is included in the workspace of a Gough platform, the orientation of the platform being constant. If the leg lengths $\rho$ of the robot are restricted to lie in the interval $[\rho_{min}, \rho_{max}]$ we have to verify that for any $X$ in $\mathcal{B}_0$ we have $\rho_{min} \leq \rho(X) \leq \rho_{max}$. As we know an analytical form for $\rho(X)$ we may determine by using interval arithmetics a lower and an upper bound $\underline{\rho(X)}, \overline{\rho(X)}$ for $\rho(X)$ if $X$ lie in a given Cartesian box. The algorithm uses a list of Cartesian box $\mathcal{L}$ which is initialized to be $\mathcal{L} = \{\mathcal{B}_0\}$ at the start and $\mathcal{L}_i$ will denote the $i$-th box in $\mathcal{L}$. The algorithm is then, starting with $i = 1$:

1. compute $\left[\underline{\rho(\mathcal{L}_i)}, \overline{\rho(\mathcal{L}_i)}\right]$ using interval arithmetics.
2. if $\underline{\rho(\mathcal{L}_i)} > \rho_{max}$ or $\overline{\rho(\mathcal{L}_i)} < \rho_{min}$, then $\mathcal{B}_0$ is not included in the workspace, as every point of $\mathcal{L}_i$, which is included in $\mathcal{B}_0$, is outside the workspace. Send the message "BOX IS OUT".
3. if $\underline{\rho(\mathcal{L}_i)} \geq \rho_{min}$ and $\overline{\rho(\mathcal{L}_i)} \leq \rho_{max}$, then $\mathcal{L}_i$ is included in the workspace, as for any point in this box the leg lengths are within the limits. Restart at 1 with the $i = i + 1$.
4. otherwise bisect $\mathcal{L}_i$ along one of its dimension (either $x, y$ or $z$) to create two new Cartesian boxes that will be stored at the end of $\mathcal{L}$. Restart at 1 with the $i = i + 1$.

The algorithm either exits at step 2, in which case part of $\mathcal{B}_0$ is outside the workspace, or it stops when all the boxes of $\mathcal{L}$ have been processed, in which case $\mathcal{B}_0$ is fully included in the workspace. Note that the previous algorithm is just an outline of what can be done, and may be improved in many different aspects.

# References


[1] Borel E. Mémoire sur les déplacements à trajectoire sphériques. *Mémoire présentés par divers savants*, 33(1):1–128, 1908.

[2] Boudreau R. and Gosselin C.M. The synthesis of planar parallel manipulators with a genetic algorithm. *ASME J. of Mechanical Design*, 121(4):533–537, December 1999.

[3] Bricard R. Mémoire sur les déplacements à trajectoire sphériques. *Journal de l'École Polytechnique*, 11(2):1–96, 1906.

[4] Cauchy A. Deuxième mémoire sur les polygones et les polyèdres. *Journal de l'École Polytechnique*, pages 87–98, May 1813.

[5] Erdman A.G. *Modern Kinematics*. Wiley, New-York, 1993.

[6] Lebesgue H. Octaèdre articulé de Bricard. *L'enseignement mathématique*, (13):150–160, 1967.

[7] Ma O. and Angeles J. Optimum architecture design of platform manipulator. In *ICAR*, pages 1131–1135, Pise, June, 19-22, 1991.

[8] Masory O. and Wang J. Workspace evaluation of Stewart platforms. *Advanced Robotics*, 9(4):443–461, 1995.

[9] Merlet J-P. An improved design algorithm based on interval analysis for parallel manipulator with specified workspace. In *IEEE Int. Conf. on Robotics and Automation*, pages 1289–1294, Seoul, May, 23-25, 2001.

[10] Merlet J-P. Designing a parallel manipulator for a specific workspace. *Int. J. of Robotics Research*, 16(4):545–556, August 1997.

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

[12] Pierrot F., Dauchez P., and Fournier A. Hexa: a fast six-dof fully parallel robot. In *ICAR*, pages 1159–1163, Pise, June, 19-22, 1991.