HAL
open science

Performance analysis and design of parallel kinematic machines using interval analysis

Carlos Viegas, David Daney, Mahmoud Tavakoli, Anibal T. de Almeida

To cite this version:

Carlos Viegas, David Daney, Mahmoud Tavakoli, Anibal T. de Almeida. Performance analysis and design of parallel kinematic machines using interval analysis. Mechanism and Machine Theory, 2017, 115, pp.218 - 236. 10.1016/j.mechmachtheory.2017.05.003 . hal-01669173

HAL Id: hal-01669173

https://inria.hal.science/hal-01669173v1

Submitted on 20 Dec 2017

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

Performance Analysis and Design of Parallel Kinematic Machines using Interval Analysis

Carlos Viegas[1], David Daney[2], Mahmoud Tavakoli[1], Aníbal T. de Almeida[1]
[1]Institute of Systems and Robotics, University of Coimbra, Portugal
[2]INRIA, Bordeaux, France
{carlosviegas;mahmoud;adealmeida}@isr.uc.pt; david.daney@inria.fr

***Abstract*—Some design methodologies for Parallel Kinematic Machines (PKM) have been proposed but with limitations regarding two main problems: how to improve multiple properties of different nature such as accuracy, force or singularity poses, and how to check these properties for all poses inside the PKM workspace. To address these problems, this work proposes to formulate the design problem as a feasibility problem and use a data representation which takes into account the uncertainty or variation of the involved parameters. This method, based on interval analysis, allows to evaluate several performance indexes of a PKM design. For validation purposes, this methodology is applied to a PKM, obtaining a continuous set of possible kinematic parameters values for its architecture which is capable of fulfilling several performance requirements over a desired workspace.**

***Index Terms*—Parallel Manipulator; Interval Analysis; Design Methodology; Workspace Determination;**

I. INTRODUCTION

One setback of parallel machines is the fact that, for a fixed mechanical architecture, their properties or performance metrics are dependent on the dimensional geometry as well as their pose. This means that generally, these machines do not have constant behavior (in terms of accuracy, stiffness, and other properties) in their overall workspace. Another particular aspect of parallel machines is the existence of singular poses inside their workspace, resulting in loss or gain of degrees of freedom (DOF) and consequently loss of control of the machine. This effect can be mitigated within the PKM design process. Through this process of property evaluation and workspace characterization, one determines the values of the PKM kinematic parameters which will improve or certify the properties of the parallel machine.

The first approach consists of an optimization of a weighted criteria depending on the robot parameters, as it chooses the solution which offers the best compromise in terms of performance. Examples of this approach include Atlas approach [1], [2], the cost function approach [3], [4], dual expansion [5], compromise programming methodology [6], physical programming methodology [7] among others [8], [9].

The second approach, on the other hand, defines the performance parameters in terms of constraints and not as subjects of optimization. It addresses the design problem in terms of feasibility, by determining a set of solutions for the kinematic parameters which ensure all performance requirements are met [10]. This approach has several advantages relative to design optimization as it is capable of dealing with manufacturing tolerances and other deviations from the nominal design parameter values. It can also deal with a large number of different properties or design parameters which is something that optimization methods usually struggle with. Optimization methods may also converge to a single solution, which might or might not be global optimal, and depends on the weights given to the performance criteria considered or the compromises made between conflicting criteria.

The proposed design methodology in this work will focus on the second approach. The goal is to design a PKM which fulfills certain desired performance thresholds over its workspace. In other words, one want to obtain the set of kinematic parameters for a PKM with a desired workspace, characterized by its joint range limits, absence of singularities, with a desired motion accuracy and force properties. Other parameters such as the occurrence of link and platform collisions or PKM stiffness can be easily added to the model but are not subject to study in the present work. The workspace is the common variable and serves to unify the properties and certify the set of kinematic parameters.

Interval analysis [11]–[13] is used to evaluate the constraints and Branch-and-Prune to characterize the constraint workspace. Interval arithmetic, proposed by Moore [14], has been used for PKM property analysis, such as accuracy [15], [16], sensitivity [17], force workspace [18], existence of singularities [19], among others. It deals with continuous intervals instead of discrete points, thus allowing a continuous evaluation of the entire workspace of the PKM as well as the entire range of its design parameters. The proposed design method is based on an algorithm which uses some well known interval analysis techniques. Some strategies employed to improve the efficiency of the algorithm are also presented and discussed. Some works have been made on parallel robot property analysis using constraints, Branch-and-Prune and interval analysis. In [20], a certified enclosure of the generalized aspects is computed. It is used to obtain connected sets of non-singular configurations for path planning of planar robots with 2 and 3 DOF, but in theory can be added additional constraints for any parallel robot case. In that work, arm and obstacle collision as well as joint limits constraints were demonstrated. However, few works have been made addressing the design of a PKM with certified performance. In [21] a robot with certified dynamic performance over a workspace is designed. As an example, a range of design parameters

is determined, which ensure that a 2DOF robot with pre-selected actuators can perform a designated task, consisting on following trajectory with a specific velocity and acceleration. In [22], a method is proposed for synthesizing the largest tolerances in the model parameters of a PKM while keeping the pose error below a given limit. A similar work is done in [16]. Most works on PKM design either focus on less than 3-DOF PKM's or on a single property.

In this article, there is a first description of the proposed methodology for performance analysis and design of PKM's. Then, a case study of a spatial PKM design is introduced. The PKM used as a case study is a derivation of the well known Triglide spatial manipulator, proposed by Budde [23], and is shown in Figure 1. This architecture is chosen as it is a part of a larger system for factory automation over a large scale, being developed in the Institute of Systems and Robots of the University of Coimbra. However, the methodology presented here can be applied to any other parallel architecture.

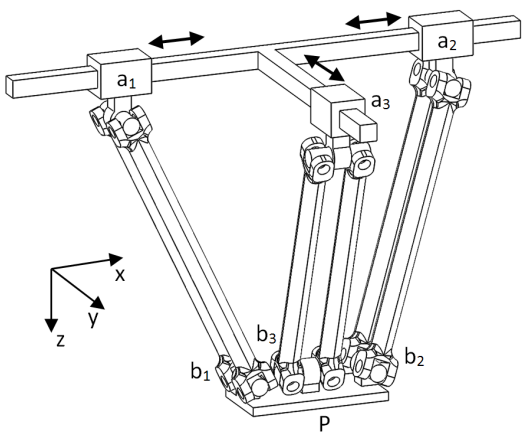

Fig. 1. Arrangement of the 3 P˜UR limbs.

Contribution and focus of this work include an efficient design methodology which addresses the evaluation of several PKM properties of different nature, including singular poses, joint limits, accuracy and force, for an entire workspace, while also taking into account possible variations or uncertainties of the geometrical parameters. Same methodology also enables design of a PKM with multiple DOF, taking into account different performance requirements. It is also the first part of a work on the design and realization of a real PKM with certified performance, so it is a self-contained thorough method which is validated with the physical realization of the PKM, shown in Figure 12.

II. DESIGN METHODOLOGY

In this section, the interval analysis tool is introduced, with a detailed discussion on how it is used to evaluate each performance parameter. Then, the outline of the proposed design algorithm is presented.

A. Interval analysis

The discrete approach to the study of a manipulator's workspace has been widely used for its simplicity. However, being able to evaluate the workspace continually (and not in a discrete way) is important since singularities or particular poses of the robot where its performance in terms of accuracy and forces is not satisfactory may occur between two certified workspace points. One can refine the evaluation by adding more points at the expense of computation velocity and efficiency. For this reason, most robust approach of Interval Analysis (IA) is used for design certification, since it deals with a continuous set of points instead of some particular discrete points. In this sense, IA can certify an entire workspace, since it provides simple tools to evaluate the lower and upper bounds for a function with interval unknowns. It can perform more evaluations on critical areas of the workspace close to singular or boundary regions, while performing much less calculations on large non-critical areas, thus largely improving computation velocity and efficiency.

An application is when some parameters are not known exactly but are bounded, such as the physical realization of the mechanical components of a robot with its manufacturing tolerances [24]. It can also be used to take into account computer round-off errors [16].

For this work the usual interval notation is used:

- Interval real - $[x] \in \mathbb{IR} = [\underline{x}\,,\overline{x}] = \{x \in \mathbb{R} | \underline{x} \leq x \leq \overline{x}\}$;
- Interval vector - $[\mathbf{v}] \in \mathbb{R}^n = [\underline{v}_i\,,\overline{v}_i]\, for\ i = 1,...,n$;
- Interval matrix - $[\mathbf{M}] \in \mathbb{R}^{m\times n} = \begin{pmatrix} [a_{11}] & ... & [a_{1n}] \\ \vdots & \ddots & \vdots \\ [a_{m1}] & ... & [a_{mn}] \end{pmatrix}$;
- Infimum - $\underline{x} = inf([x]) \triangleq inf\{a \in \mathbb{R} | \forall x \in [x], a \leq x\}$;
- Supremum - $\overline{x} = sup([x]) \triangleq sup\{b \in \mathbb{R} | \forall x \in [x], x \leq b\}$;
- Radius - $rad([x]) = \delta \triangleq \frac{\overline{x}-\underline{x}}{2}$;
- Width/Diameter - $wid([x]) = 2\delta \triangleq \overline{x} - \underline{x}$;
- Midpoint/Center - $mid([x]) = \tilde{x} \triangleq \frac{\overline{x}+\underline{x}}{2}$;
- Interval approximation of the solution set - $\square$;
- Inner box - $\blacksquare$.

As, for example, in the real interval $[-3,5]$ the infimum is -3, the supremum is 5, the radius is 4, diameter is 8 and the midpoint is 1.

1) Interval arithmetic and extensions: Interval arithmetic allows to implement basic operators $(+,-,\times,/,^n,\sqrt{},\exp,\sin,\cos,etc\ldots)$, such that [14]:

$$[x] \circ [y] \subseteq [\{x \circ y \mid x \in [x], y \in [y]\}] \tag{1}$$

Then, it is possible to provide an interval extension or inclusion, noted $[f]$ to real function f as:

$$\forall [x] \in \mathbb{IR},\ \ [f]([x]) \supseteq \{f(x); \forall x \in [x]\} \tag{2}$$

There exist several types of interval extension, such as *Natural Inclusion*, where every classical operator is replaced by its interval counterpart, or *Taylor Extension*.

The main problem with interval analysis is the overestimation of the resulting interval extension bounds, which may lead to pessimist evaluations of interval arithmetic. Indeed,

in interval arithmetic, the several occurrences of variables are treated as independent and are not correlated:

$$[x]-[x]=\{x-y|x\in[x],y\in[y]\}\supset\{x-x|x\in[x]\} \quad (3)$$

This, in most cases, leads to loss of properties and to overestimation, where the upper (or lower) bound of $[F]$ is not exactly max (or min) $f(x)$ for $x\in[x]$ [14].

B. Interval extensions of robot properties

The proposed design methodology can be divided into two steps: *Verification* and *Design*.

1) Verification: The workspace W of the robot is the common denominator in this study. One will analyze the robot's static configurations and determine the workspace for each of the performance thresholds. The workspace W_i for the i performance property, characterized by m inequality constraints noted C_i, formulated as conjunctions and/or disjunctions of inequalities, for a real or interval set of n kinematic parameters, $\mathbf{p}$ or $[\mathbf{p}]$ respectively,can be defined as:

$$W_i([\mathbf{p}_t])=\{\mathbf{x}|\forall\mathbf{p}\in[\mathbf{p}_t],C_i(\mathbf{x},\mathbf{p})\} \quad (4)$$

The exact description of the workspace is difficult to obtain formally. In fact, one is interested in the inner approximation of these sets described by interval boxes. Continuous intervals, depending on their dimensions, can be graphically represented by boxes (lines for 1D, rectangles for 2D, parallelepipeds for 3D, and hypercubes for superior dimensions) with their sides parallel to the reference axis of the chosen parameters. These boxes are tested and labeled as internal or external according to whether or not they are part of the solution set, defined by its constraints. A box (ex : $[\mathbf{x_t}]$) is internal to W_i, if the condition $\forall\mathbf{x}\in[\mathbf{x_t}]$, $C_i(\mathbf{x},\mathbf{p})$ is met. This set of inner boxes are denoted $\blacksquare W_i([\mathbf{p}_t])$ and such as $\blacksquare W_i([\mathbf{p}_t])\subset W_i$. This allows to insure that one is fully inside the workspace. External boxes are boxes where the constraint evaluation deems results which lie completely outside the solution set.

An undetermined box is characterized by having parts which belong to the solution set while others lie outside of it. In this case one cannot be sure if the effect of the interval overestimation might be pessimist, so the box is bisected in two and the resulting boxes are re-evaluated.

Then an inner approximation of the final workspace is obtained:

$$W_{all}([\mathbf{p}_t])=\{\mathbf{x}|\forall i=[1,\ldots,m],\ \forall\mathbf{p}\in[\mathbf{p}_t],\ C_i(\mathbf{x},\mathbf{p})\} \quad (5)$$

by intersecting all performance workspaces such that:

$$\blacksquare W_{all}([\mathbf{p}_t])=\bigcap_{i=1}^{m}\blacksquare W_i \quad (6)$$

For W_{all}, the condition for an interior box is $\forall\mathbf{x}\in[\mathbf{x_t}]$, $\forall i\in[1,...]$, $C_i(\mathbf{x},\mathbf{p})$.

2) Design: In *Design*, one finds all possible values of the n kinematic parameters $\mathbf{p}$ vectors for a family of PKM whose performance is certified and complies with all k desired performance parameters, characterized by m inequality constraints noted C_i in a given interval workspace box noted $[\mathbf{x_d}]$:

$$D([\mathbf{x_d}])=\{\mathbf{p}|\forall i=[1,\ldots,m],\ \forall\mathbf{x}\in[\mathbf{x_d}],\ C_i(\mathbf{x},\mathbf{p})\} \quad (7)$$

For the design problem, if a box (ex : $[\mathbf{p}_t]$) is internal, a condition $\forall\mathbf{p}\in[\mathbf{p}_t]$, $[\mathbf{x_d}]\subset W_{all}(\mathbf{p})$ or $[\mathbf{x_d}]\subset W_{all}([\mathbf{p}_t])$ must be met.

In the design methodology, there are boxes for both the search space (geometrical parameters of the PKM) and the variation domain of the parameters (workspace). Bisecting on the variation domain of the parameters and evaluating smaller domains reduces the pessimism and can improve the results of the box evaluation, thus constituting an improvement on the method efficiency and ensuring the convergence of the algorithm. Being $[\mathbf{x}]$ the search space and $[\mathbf{y}]$ the variation domain, for a given quantified constraint ($\forall\mathbf{x}\in[\mathbf{x}]$, $\forall\mathbf{y}\in[\mathbf{y}]$, $f(\mathbf{x},\mathbf{y})\leq 0$) one can compute the following interval evaluations $[\mathbf{z}]:=f([\mathbf{x}],[\mathbf{y}])$, $[\mathbf{z}]^1:=f([\mathbf{x}],[\mathbf{y}]^1)$ and $[\mathbf{z}]^2:=f([\mathbf{x}],[\mathbf{y}]^2)$, where $[\mathbf{y}]^1$ and $[\mathbf{y}]^2$ are obtained by bisecting the variation domain $[\mathbf{y}]$, and knowing that the interval hull $\square([\mathbf{z}]^1\cup[\mathbf{z}]^2)\subseteq[\mathbf{z}]$. Then, if either $[\mathbf{z}]^1$ or $[\mathbf{z}]^2$ lies outside the solution set, one can discard the entire parameter set $[\mathbf{x}]$. To know when to perform a bisection on parameter domains, in a similar fashion to the work of Goldsztejn [25], one defines a threshold on the ratio:

$$\frac{wid(\square([\mathbf{z}]^1\cup[\mathbf{z}]^2))}{wid([\mathbf{z}])} \quad (8)$$

Below which the parameter domains are bisected. This solution relies on three interval evaluations of the function f, which, as the previous author states, is cheap with respect to the use of interval contractors. In this work, a threshold of 0.80 was used and shown to lead to good performances in average. Other methods exist in literature [26], [27] and although not discussed here may be tested in the future to compare with this method.

Once chosen the domain for bisection, the best bisection direction choice is also critical for the efficiency of the algorithm. The *Classical Method* for the subdivision process is the bisection of the box $[\mathbf{x}]$ perpendicular to a direction of maximum width. For an interval $I^k=[a_k,b_k]$, bisection occurs at its middle point in order to create two new intervals $I_1^k=[a_k,(a_k+b_k)/2]$ and $I_2^k=[(a_k+b_k)/2,b_k]$.

However, the evaluation function f might not variate as much for that direction of bisection as for others, resulting in the creation of an unnecessary large number of boxes. For this reason, one should look for efficient methods for the selection of the direction of bisection to reduce the number of sub-boxes generated, thus reducing the required computation space and time. Ratz has studied four different rules for the selection of subdivision directions [28]. Each of the rules selects a direction k by using a merit function:

$$k = min\{j|j \in \{1,...,n\} \, and \, \mathscr{R}(j) = max_{i=1}^{n} \mathscr{R}(i))\} \quad (9)$$

where $\mathscr{R}(i)$ is determined by the given rule. They have empirically proved, using a wide spectrum of unconstrained test problems, that the correct choice of bisection rules can effectively reduce calculation time and function evaluations by around 20% and the space complexity by around 15%, when compared with the *Classical Method*, which selects the direction of maximum width ($\mathscr{R}(i) = wid(\mathbf{x_i})$). The most effective bisection rules where Rule B (Hansen and Walster), defined by:

$$\mathscr{R}(i) = wid(\bigtriangledown f_i([\mathbf{x}]))wid([\mathbf{x_i}]) \quad (10)$$

And Rule C (Ratz), defined by:

$$\mathscr{R}(i) = wid(\bigtriangledown f_i([\mathbf{x}])([\mathbf{x_i}] - mid([\mathbf{x_i}]))) \quad (11)$$

The relative efficiency of each bisection method depends on each problem and for one specific case, one method might present serious advantages or disadvantages over all others. For the proposed algorithm in this work, three different bisection methods were tested in order to find the most efficient one.

Bisection can occur until a minimum box size is achieved. In this case, if still no conclusion can be drawn about the nature of the box, it is characterized as a boundary box. This minimum box size is called study minimum resolution. This is the principle of the Branch and Prune algorithm and constitutes the basis of this design algorithm.

Property evaluations require solving by intervals linear equality or inequality constraints. For the equality constraint problem, one can apply the method proposed in [29]. However, since they can also be interpreted as inequalities with no prejudice to the result or method, and in an effort to maintain coherence throughout the whole text, the authors opted to use inequality constrains, which will be discussed in detail in the next section, but can be roughly represented by the following linear interval system:

$$\begin{aligned} \underline{\mathbf{b}} \leq \mathbf{A}\mathbf{x} \leq \overline{\mathbf{b}}, \mathbf{A} \in [\mathbf{A}] \\ \text{or } \mathbf{A}\mathbf{x} = \mathbf{b}, \mathbf{A} \in [\mathbf{A}], \mathbf{b} \in [\underline{\mathbf{b}}, \overline{\mathbf{b}}] \end{aligned} \quad (12)$$

Where $[\mathbf{A}]$ is composed by invertible matrices. The problem consists in finding out the subset x, in the form of an interval vector:

$$\Sigma_{\exists,\exists}([\mathbf{A}],[\mathbf{b}]) := \{\mathbf{x} \in \mathbb{R}^n | \, \exists \mathbf{A} \in [\mathbf{A}], \, \exists \mathbf{b} \in [\mathbf{b}], \, \mathbf{A}\mathbf{x} = \mathbf{b}\} \quad (13)$$

In this case, and in a similar way to the evaluation of interval polynomial equations, a simple adaptation of scalar algorithms is not feasible. The main source of difficulties connected with computing the solution set x is its complicated structure, which is generally nonconvex.

Oettli and Prager first proposed a technique to deal with this problem, in 1964 [30]. Taking advantage from the fact that the intersection of the solution set x with each orthant is, in fact, a convex polyhedron, Oettli [31] proposed, using a linear programming procedure in each orthant to determine the infimum and supremum for the solution set. Though this method effectively obtains larger solution boxes than other methods, it is extremely computation intensive, since it requires an evaluation of the linear system for each orthant.

For this reason, the proposed algorithm is based on a theorem proposed by Beaumont [32], which is an evolution of the Oettli-Prager theorem:

$$\Sigma_{\exists,\exists}([\mathbf{A}],[\mathbf{b}]) \subset \square \left\{ x; \begin{pmatrix} A_c - \delta_A D_\alpha \\ -A_c - \delta_A D_\alpha \end{pmatrix} x \leq \begin{pmatrix} \overline{b} + \delta_A \beta \\ -\underline{b} + \delta_A \beta \end{pmatrix} \right\} \quad (14)$$

Where D_α is a diagonal matrix whose elements are α_i and β is a vector whose elements are given are the β_i. Both this scalar matrix and vector depend on an initial approximation of the solution set x and are given by:

$$\alpha_i = \frac{|\overline{x}_j| - |\underline{x}_j|}{\overline{x}_j - \underline{x}_j} \quad and \quad \beta_i = \frac{\overline{x}_j |\underline{x}_j| - \underline{x}_j |\overline{x}_j|}{\overline{x}_j - \underline{x}_j} \quad (15)$$

While it does not require an evaluation for each orthant, it is an iterative method, which might turn out to be computation intensive. However, tests have shown that, with a good initial approximation to the solution set, one obtains sharper results than Oettli and Prager, for only one or two iterations. To reduce the effect of the overestimation and contract the bounds of the solution sets, filtering methods are employed:

$$[\mathbf{x}_{new}] = Filtering([\mathbf{x}_{old}], C([\mathbf{x}])) \; so \; that \; [\mathbf{x}_{new}] \subseteq [\mathbf{x}_{old}] \quad (16)$$

Filtering can be made using 2B, 3B, Gauss Elimination, Taylor, Hansen-Blink, Newton, among other methods [13]. If the filtering leads to an empty box, this box is sent to the list of outside boxes.

C. Robots properties characterization

1) Joint Range: To obtain the manipulator's workspace limited by the reachable extent of its drives and joints, called reachable workspace, one has to first develop the kinematics of the robot.

The interval extension of the inverse kinematics problem (IK) to a box $[\mathbf{x}]$ ($\forall \mathbf{p} \in [\mathbf{p}_t]$) allows to overestimate all possible variations of the joint coordinates for all $\mathbf{x} \in [\mathbf{x}]$. Generally there exist several solutions to the IK problem, although in this work, an unique solution of interest or a way to select it is assumed, as described in section III.B Kinematics. In fact, adding more constraints to the problem, such as leg collisions, would eventually lead to the unique solution of interest considered. The constraint for joint range property checks if the joint coordinates obtained are inside the defined joint ranges, noted $[\mathbf{q_d}] = [\underline{\mathbf{q_d}}, \overline{\mathbf{q_d}}]$, for the property workspace W:

$$C_1([\mathbf{x}],[\mathbf{p}]) \Leftrightarrow \underline{\mathbf{q_d}} \leq \square(IK([\mathbf{x}],[\mathbf{p}])) \leq \overline{\mathbf{q_d}} \quad (17)$$

In this case, if $IK([\mathbf{x}],[\mathbf{p}]) \subset [\mathbf{q_d}]$, it is an internal box. Otherwise, if $IK([\mathbf{x}],[\mathbf{p}]) \cap [\mathbf{q_d}] = \emptyset$, it is an external box.

2) Singularities: Singular configurations are particular poses of a parallel manipulator, in which the mechanism loses its rigidity and degrees of freedom or becomes uncontrollable. Hence, singularities should be avoided at all costs for most applications. As in the work by Gosselin [33], the analysis of the manipulator's two Jacobian matrices, parallel and serial, is proposed, to establish three types of singularities:

- First kind: they occur when the determinant of the serial Jacobian matrix is null. This is the case when the directions of one or more of the legs are perpendicular to their corresponding actuator directions leading to the loss of one or more DOF.
- Second kind: they occur when the determinant of the parallel Jacobian matrix is null. This condition implies that the links are aligned with the moving platform, with the extensions of lines AiBi passing through the center point P of the end-effector. The robot gains one or more degrees of freedom. These are the most dangerous types of singularities and are associated with the loss of stiffness of the manipulator.
- Third kind: also called combined singularity, they occur when both serial and parallel Jacobian matrices are not full rank. In this situation, the robot gains two more DOF and becomes uncontrollable.

The singularity constraint to a box $[\mathbf{x}]$ ($\forall \mathbf{p} \in [\mathbf{p}_t]$) is defined as:

$$C_2([\mathbf{x}],[\mathbf{p}]) \Leftrightarrow det(J_{inv}([\mathbf{x}],[\mathbf{p}])) < 0 \vee 0 < det(J_{inv}([\mathbf{x}],[\mathbf{p}])) \quad (18)$$

If the set $\{0\} = [0,0] \notin det(J_{inv}([\mathbf{x_t}],[\mathbf{p}]))$, one is sure there is no singular pose in the workspace of the robot.

Another approach is to check the regularity of a matrix ($\forall \mathbf{x} \in [\mathbf{x_t}] | \forall \mathbf{p} \in [\mathbf{p}_t], J_{inv}(\mathbf{x},\mathbf{p})$ are regular) as an alternative to the evaluation of the Jacobian determinant [34], [35]. A different approach is used in [36], where the authors compute the determinant of the jacobian for single poses corresponding to the upper and lower bound of an interval, and try to find inversions of the signal of the determinant, meaning that there is a singularity inside the pose interval, as the determinant of J_inv is a real valued continuous and differentiable function.

3) Motion Accuracy: Error analysis is an essential study for any PKM design exercise, as it is shown by the numerous works on this subject [24], [37]–[47]. It consists on finding the positioning errors of a given robot at some specific location within the workspace, by solving the following interval linear system of equations:

$$J_{inv}([\mathbf{x}],[\mathbf{p}])[\Delta\mathbf{x}] = [\delta\mathbf{q}] \quad (19)$$

Which relates the positioning errors $[\Delta\mathbf{x}]$ of the end-effector with the actuated joints accuracy $[\delta\mathbf{q}]$, through the inverse jacobian matrix $Jinv$, which is pose dependent but also depends on the geometrical parameters $[\mathbf{p}]$ that define the geometry of the robot (considered as intervals to account for the bounded manufacturing errors).

While this is actually a first order approximation of the pose error, near singularities the whole process may turn out to be non reliable. An approximation to the distance to singularities can be found in [48], where the authors avoid singularities by restraining the workspace to a set of static poses where the joint forces do not exceed a certain threshold. A similar solution is employed for the force workspace determination (constraint C_4) in this work, and can be used for the same purpose, to improve on the reliability of this method.

A PKM moves within a given workspace W that is defined as intervals for $[\mathbf{x}]$ parameters. The desired vector of maximal positioning errors $[\Delta\mathbf{x_d}]$, is defined as a set of allowed ranges for the errors on $[\mathbf{x}]$. The goal is to find robot geometries for which one can ensure that whatever is the pose of the robot within the workspace, the positioning error will be included in $[\Delta\mathbf{x_d}]$.

Solving by interval the problem for a given accuracy $[\delta\mathbf{q}]$, the internal box $[\mathbf{x}]$ ($\forall \mathbf{p} \in [\mathbf{p}_t]$) test condition consists on checking if the obtained accuracy $[\Delta\mathbf{x_{res}}]$, which is an overestimation of the real accuracy, is inside a desired accuracy interval $[\Delta\mathbf{x_d}]$ is done by:

$$C_3([\mathbf{x}],[\mathbf{p}]) \Leftrightarrow \underline{\Delta\mathbf{x_d}} \leq \square\Sigma_{\exists,\exists}(J([\mathbf{x}],[\mathbf{p}]),[\delta\mathbf{q}]) \leq \overline{\Delta\mathbf{x_d}} \quad (20)$$

If $J([\mathbf{x}],[\mathbf{p}])[\delta\mathbf{q}] \subseteq [\Delta\mathbf{x_d}]$ then it is an internal box. Otherwise, if $J([\mathbf{x}],[\mathbf{p}])[\delta\mathbf{q}] \cap [\Delta\mathbf{x_d}] = \emptyset$ it is an external box.

4) Joint Forces: Static analysis reveals one very interesting phenomenon in the vicinity of singularities, characterized by the existence of a load such that the internal forces in the joints of the structure tend to infinity [48]. Large payloads also require bigger actuation forces. Such large forces can lock the entire mechanism and, in the worst scenario, lead to its breakdown.

To avoid this, the designer can define a threshold τ_{max} for the maximum internal forces in the joints. The areas of the manipulator workspace in which the internal forces in the joints do not exceed this threshold constitute the force workspace.

At static equilibrium, the fundamental relation between the joint forces interval vector $[\tau]$, the external wrench exerted on the environment $[F]$ and the transpose of the inverse kinematic jacobian matrix J_{inv}^T, is given by:

$$J_{inv}^T([\mathbf{x}],[\mathbf{p}])[\tau] = [F] \quad (21)$$

The wrench $[F]$ contains all forces applied by the geometrical center of the end-effector, i.e. in the origin of its reference frame, to the environment. When, for instance, the manipulator carries an $[m]$ payload, where $[m]$ is the interval mass of

the payload, the end-effector must counterbalance the weight $[F] = [m].\mathbf{g}$, where $\mathbf{g}$ is the gravity acceleration vector. Solving by interval the linear problem for a given wrench $[F]$, the force constraint for a box $[\mathbf{x}]$ ($\forall \mathbf{p} \in [\mathbf{p}_t]$), consists on checking if the obtained joint forces $[\tau_{\mathbf{res}}]$ are inferior to the maximum joint forces $\tau_{\mathbf{max}}$, and can be defined as:

$$C_4([\mathbf{x}],[\mathbf{p}]) \Leftrightarrow \forall F \in [F], \underline{\tau} \leq \square\Sigma_{\exists,\exists}(J^T([\mathbf{x}],[\mathbf{p}])F) \leq \overline{\tau} \quad (22)$$

With $[\tau] = [-\tau_{max}, \tau_{max}]$.

If $J^T([\mathbf{x}],[\mathbf{p}])[F] \subseteq [\tau]$ then it is an internal box. Otherwise, if $J^T([\mathbf{x}],[\mathbf{p}])[F] \cap [\tau] = \emptyset$, it is an external box.

D. Algorithm Outline

Here, an outline of the algorithm developed for the study and design of the parallel manipulator, using interval analysis, is presented.

In the verification routine Algorithm 1, the user obtains the workspace for the robot characterized by the unique set of manipulator geometrical parameters $\mathbf{p}$ such as the length of the limbs l_i or the width of the end-effector w.

In the design routine Algorithm 2, the result is a set $[\mathbf{D}]$ of kinematic parameters $[\mathbf{p}_j]$ which form a family of certified PKM's. Then, even if the physical realization of the robot differs from the theoretical model while staying within the given manufacturing errors bounds, one can certify the robot design for the required performance parameters.

Algorithm 1 Verification Routine

```
1:  procedure WORKSPACE DETERMINATION
2:      inputs:[Q_d]; [ΔX_d]; [τ]; p;
3:      for i=1,...,m do
4:          L = {initial[x]}; n = size(L);
5:          while n>0 do
6:              [b] = L(1);                          ▷ take 1st box of L
7:              L(1) = []; n = n − 1;                ▷ erase 1st box of L
8:              [b_new] = Contract[b];
9:              if Contract fails then
10:                 goto 5;
11:             end if
12:             if ∀x ∈ [b_new], C_i(x,p) met then
13:                 [W_i] = [W_i] ⋃ [b_new];
14:             else if ∀x ∈ [b_new], C_i(x,p) not met then
15:                 goto 5;
16:             else
17:                 if [b_new] > minDim then
18:                     L ← bisect [b_new]; n=n+2;
19:                 else
20:                     B_i ← [b_new];               ▷ list of boundary boxes
21:                 end if
22:             end if
23:         end while
24:     end for
25:     return [W];
26: end procedure
```

Algorithm 2 Design Routine

```
1:  procedure DESIGN CERTIFICATION
2:      inputs:[Q_d]; [ΔX_d]; [τ]; [W];
3:      [D_0] = {initial[p]};
4:      for i=1,...,m do
5:          L = [D_{i−1}]; n = size(L);
6:          while n>0 do
7:              [b] = L(1);                          ▷ take 1st box of L
8:              L(1) = []; n = n − 1;                ▷ erase 1st box of L
9:              [b_new] = Contract[b];
10:             if Contract fails then
11:                 goto 6;
12:             end if
13:             if ∀x ∈ [W], ∀p ∈ [b_new], C_i(x,p) met then
14:                 [D_i] = [D_i] ⋃ [b_new];
15:             else if ∀x ∈ [W], ∀p ∈ [b_new], C_i(x,p) not met then
16:                 goto 6;
17:             else
18:                 if [b_new] > minDim then
19:                     if eq.(8) > 0.8 then
20:                         L ← bisect [b_new]; n=n+2;
21:                     else
22:                         bisect [W]; goto 13;
23:                     end if
24:                 else
25:                     B_i ← [b_new];               ▷ list of boundary boxes
26:                 end if
27:             end if
28:         end while
29:     end for
30:     return [D];
31: end procedure
```

If $\mathbf{p}_j^m$ is used as nominal value of a given geometrical parameter $\mathbf{p}_j$, for the manufacturing process one may assume that the real value of $\mathbf{p}_j$ will lie in the range $[\mathbf{p}_j^m - \varepsilon_j, \mathbf{p}_j^m + \varepsilon_j]$. This implies that if a solution interval $[\mathbf{p}_j] = [a,b]$ for the parameter $\mathbf{p}_j$ is found, whose width is larger or equal to $2\varepsilon_j$, then one is able to guarantee that the real robot will satisfy property (7), by choosing as theoretical manufacturing value a number in the range $[a - \varepsilon_j, b + \varepsilon_j]$, as this guarantee that the real value will be in $[\mathbf{p}_j]$.

Notice that even though the inclusion tests show as performed in serial, they can also be performed in parallel fashion for both routines. Both strategies have their advantages and disadvantages, demonstrating higher speeds and efficiency depending on the calculation conditions, as will be discussed in the results section. Notice also that in the Algorithm 2, line 22, bisection occurs on the variation domain of the parameters, in this case, the workspace $[W]$. While not shown in the pseudo-code, this is followed by new evaluations of the constraint for a smaller part of the workspace, in order to reduce the overestimation effect. If a single of these evaluations results in an outside box, then the entire parameter set $[b_{new}]$ can be discarded.

III. CASE STUDY - 3 PˆUR SPATIAL PARALLEL MANIPULATOR

In this section, the architecture of the PKM used as a case study is presented. The kinematics are developed and a singularity study on its workspace is performed. Then, by using the verification routine of the algorithm for given discrete values of the kinematic parameters, a deeper study on its performance is made. The goal is to get the reader to fully understand the characteristics of this specific PKM architecture before performing the PKM design.

A. Architecture

The architecture of the case study PKM relies on 3 limbs with a PˆUR joint pair (Figure 1). Its arrangement, with two parallel and one perpendicular limb, and coupling ensures that the extra rotation degree of freedom from each limb is suppressed, resulting in a spatial manipulator with only translational movements. The difference to the similar Triglide manipulator lies in the orientation of its actuators. In this case, two prismatic actuators, a_1 and a_2, are co-linear and the third one, a_3 is perpendicular, being all three co-planar,as depicted in Figure 1. The following system geometrical parameters shall be considered:

- Ω_O - fixed Cartesian reference frame (Ω_O,$\vec{x}$,$\vec{y}$,$\vec{z}$);
- Ω_P - moving Cartesian end-effector reference frame (Ω_P,$\vec{x_p}$,$\vec{y_p}$,$\vec{z_p}$);
- *X(x,y,z)* - coordinates of the end-effector relative to Ω_O;
- q_i - set of actuator i position coordinates relative to Ω_O;
- A_i - attach. point of limb i to actuator i relative to Ω_O;
- B_i - attach. point of limb i to end-effector relative to Ω_O;
- b_i - attach. point of limb i to end-effector relative to Ω_P;

And the design parameters:

- l_i - limb i length;
- w - end-effector width;
- d - distance of b_3 to end-effector center;

With i = *1,2,3*.

Figure 2 illustrates the simplified manipulator geometry.

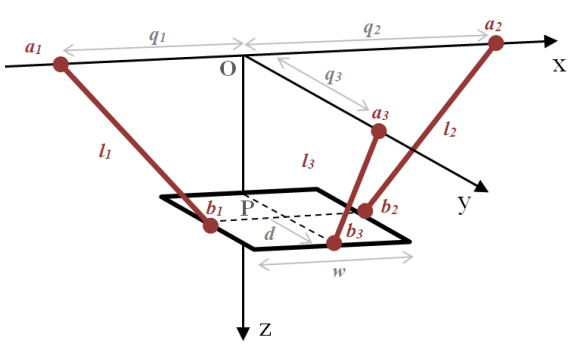

Fig. 2. Referential and coordinates for the PKM.

The fixed Cartesian reference frame's origin is in the point O. The tree limbs are assumed to be equal in length ($l_1 = l_2 = l_3 = l$). Links l_1 and l_2 are anchored at the extremities of the end-effector. Link l_3 is anchored at the a point situated at a distance d from the center point P of the end-effector, measured in the y axis. In this case study, this distance is equal to zero, meaning all three anchor points to the end-effector are co-linear. This patent-pending configuration facilitates the reconfiguration of the manipulator, even though this is a feature which will not explore in this paper. However, for generality sake, the robot's kinematics were still developed with the possibility of not having all three points co-linear.

Let b_i be the position vector of the attachment point of limb i to end-effector relative to the moving Cartesian end-effector reference frame P. Then b_i coordinates are given by:

$$b_1 = [-\frac{w}{2},0,0]^T \ , \ b_2 = [\frac{w}{2},0,0]^T \ , \ b_3 = [0,d,0]^T \qquad (23)$$

The transformation from the moving platform to the fixed base can be described by a position vector p = $\overline{OP}$ and a $3x3$ rotation matrix R_P. Since the manipulator displays spatial movements with only translations and no rotations, by calibration, the $3x3$ rotation matrix R_P is equal to the identity matrix. The position vector B_i with respect to the fixed coordinate system is obtained by the following transformation:

$$B_i = P + R_P b_i \qquad i = 1,2,3 \qquad (24)$$

Prismatic actuators a_1 and a_2 work on the x axis. The actuator a_3 works on the y axis. Actuator position coordinates, relative to the fixed reference frame, are given by:

$$A_1 = [q_1,0,0]^T \ , \ A_2 = [q_2,0,0]^T \ , \ A_3 = [0,q_3,0]^T \qquad (25)$$

B. Kinematics

1) Inverse Kinematics Problem: The inverse kinematic implicit model is obtained by using the three closure equations, constraints of the kinematic chains, which link the Cartesian space variables to the joint space variables. The three F constraints equations for the robot are given by:

$$F_i(X,q) = \|P(X) + I.b_i - A_i(q)\|^2 - l^2 = 0 \qquad i = 1,2,3 \quad (26)$$

For this specific problem there are 2^3 different sets of solutions to the inverse kinematics problem, i.e., for a given location of the end-effector, several sets of actuator positions are possible. From these, a single solution of interest is selected, using defined model constraints. All possible solutions are presented, by considering both positive and negative ($\pm$) roots, but in brackets it is shows which one should be chosen to obtain the unique desired set of solutions.

With equations (23), (25) and (26), the IK implicit model equations (27) are obtained:

$$IK \begin{cases} q_1 = x \pm \sqrt{l^2 - y^2 - z^2} - w/2 & (-) \\ q_2 = x \pm \sqrt{l^2 - y^2 - z^2} + w/2 & (+) \\ q_3 = y \pm \sqrt{l^2 - x^2 - z^2} + d & (+) \end{cases} \qquad (27)$$

2) Forward Kinematics Problem: The forward kinematics problem (FK) can be obtained by solving the three F constrain equations (26) in order to the end-effector coordinates:

$$FK \begin{cases} x = \frac{q_1+q_2}{2} \\ y = \frac{(q_1+w/2).(q_2-w/2)+(q_3-d)^2}{2(q_3-d)} \\ z = \pm\sqrt{l^2-(\frac{q_1-q_2+w}{2})^2-y^2} \qquad (+) \end{cases} \tag{28}$$

In the z coordinate expression one opted to use y instead of its corresponding expression for simplification purposes. Once again, for a given set of actuator positions there are two possibilities for the position of the end-effector, which correspond to the intersection points of three spherical surfaces. By considering that the z coordinate is always positive, one finds the unique desired solution.

3) Jacobians: Differentiating the closure equations (26) leads to the velocity model, written in the matrix form as:

$$J_x.\dot{X} + J_q.\dot{q} = 0 \tag{29}$$

Where $\dot{X} = \begin{bmatrix} v_p & w_p \end{bmatrix}^T$ is the vector of the end-effector velocities and $\dot{q} = \begin{bmatrix} \dot{q}_1 & \dot{q}_2 & \dot{q}_3 \end{bmatrix}^T$ is the vector of actuated joint rate. The J_x is the 3×3 parallel Jacobian matrix (reduced since the end-effector does not have angular velocities) and J_q is the 3×3 serial Jacobian matrix, given by:

$$J_x = \begin{bmatrix} \sqrt{l^2-y^2-z^2} & y & z \\ -\sqrt{l^2-y^2-z^2} & y & z \\ x & -\sqrt{l^2-x^2-z^2} & z \end{bmatrix} \tag{30}$$

And:

$$J_q = \begin{bmatrix} \sqrt{l^2-y^2-z^2} & 0 & 0 \\ 0 & -\sqrt{l^2-y^2-z^2} & 0 \\ 0 & 0 & -\sqrt{l^2-x^2-z^2} \end{bmatrix} \tag{31}$$

When J_q is not singular, i.e., det(J_q)$\neq 0$, one can obtain the Inverse Jacobian matrix ($Jinv = J_q^{-1}.J_x$):

$$J_{inv} = \begin{bmatrix} 1 & \frac{y}{\sqrt{l^2-y^2-z^2}} & \frac{z}{\sqrt{l^2-y^2-z^2}} \\ 1 & -\frac{y}{\sqrt{l^2-y^2-z^2}} & -\frac{z}{\sqrt{l^2-y^2-z^2}} \\ -\frac{x}{\sqrt{l^2-x^2-z^2}} & 1 & -\frac{z}{\sqrt{l^2-x^2-z^2}} \end{bmatrix} \tag{32}$$

C. Singularity Loci

The determinant of the J_x reduced parallel Jacobian matrix (30) is given by:

$$|J_x| = 2z(y+\sqrt{l^2-x^2-z^2})(\sqrt{l^2-y^2-z^2}) \tag{33}$$

Which in turn is equivalent to:

TABLE I
CASE STUDY GEOMETRICAL PARAMETERS

Parameter	Value (mm)
l (link length)	400
$q1_d$, $q2_d$ (a1 and a2 ranges)	-500 to +500
$q3_d$ (a3 range)	0 to +500
w (end-effector width)	142
d (distance of b_3 to the center of end-effector)	0
Δq (actuator accuracy)	0.1
τ_{max} (actuator max force in Newton)	15

$$|J_x| = -q_3(q_1-q_2+w)z \tag{34}$$

The determinant of the J_q serial Jacobian matrix (31) is trivial to calculate since it is a diagonal matrix:

$$|J_q| = (\sqrt{l^2-x^2-z^2})(l^2-y^2-z^2) \tag{35}$$

Which in turn is equivalent to:

$$|J_q| = (x-\frac{w}{2}-q_1)(x+\frac{w}{2}-q_2)(y-q_3) \tag{36}$$

As mentioned earlier, singularities occur when the determinants of the jacobians are null. By equaling equations (33) and (35) to zero, one obtains the expressions for the singular loci surfaces, shown in Figure 3. In this particular case, all singularities happen at the vicinity or at the boundary of the robot's workspace, which in turn guarantees that no internal singularities exist. In fact, when met, this set of 3 constrain conditions ensures that one never reach a singular pose:

$$\left\{q_1 < x-\frac{w}{2}\ ,\ q_2 > x+\frac{w}{2}\ ,\ q_3 > y\right\} \tag{37}$$

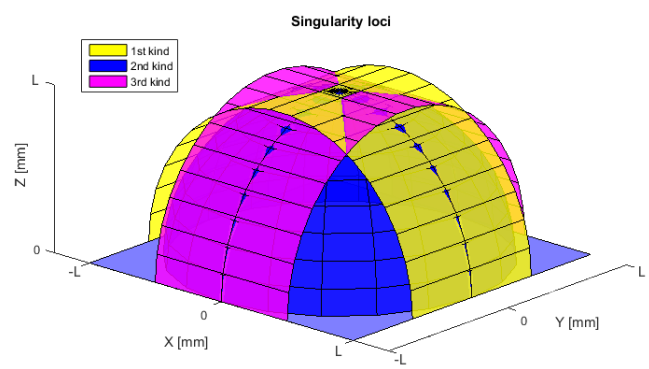

Fig. 3. Singularities loci according to type.

D. Performance Analysis

In this section the proposed algorithm is used to evaluate the performance of the case study manipulator given the geometrical parameters present in Table I.

The algorithm was developed in the Matlab R2015a environment, with the INTLAB V7.1 package, developed by Siegfried M. Rump, head of the Institute for Scientific Computing at the

Hamburg University of Technology, Germany, Copyright (c) 1998 – 2013, under academic licenses. The algorithm ran on a computer with a AMD A6-7400K Radeon R5 6 Compute Cores (2C+4G) at 3.50 GHz and 8Gb of RAM.

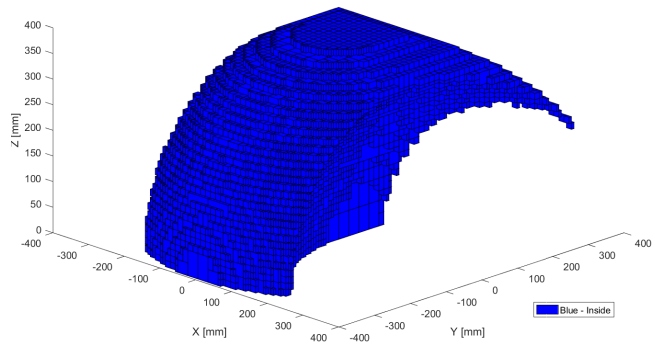

Fig. 4. 3D Reachable Manipulator Workspace. Calculation time: 3375s, N. of Intervals: 100000, Min. Resolution: $25mm^3$.

1) Reachable Workspace: The Figure 4 shows the reachable workspace of the manipulator, in 3D space. This is useful to give an idea of the overall shape and size of the workspace, but due to the high number of boxes (100000), it consumes a lot of time and processing power. It also becomes more difficult to analyze interior parts of the workspace volume. In fact, most of the time, one is only interested in the behavior and properties of the robot on a specific horizontal plane. For this reason, from this point forward, the analysis will focus on a horizontal plane situated 310mm above the rails (z=310).

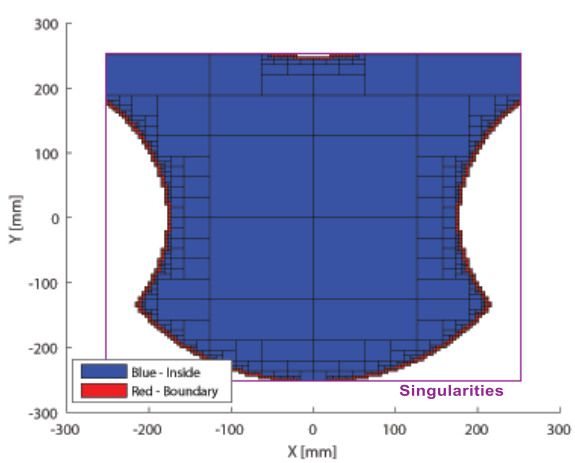

Fig. 5. Reachable Manipulator Workspace on plane z=310. Calculation time: 22.1s, N. of Intervals: 1010, Interior area: $0.191m^2$.

Figure 5 shows the 2D reachable work area for this horizontal plane. The interior area is equal to 0.191 m^2. The minimum resolution considered was $25mm^2$. Notice the red boundary boxes separating the interior boxes from the exterior space. Where there is no such boundary is in fact the place where boundary singularities occur. Their loci is represented by the purple square.

2) Accuracy Workspace: The manipulator accuracy workspace is shown in Figure 6. The useful work area (shown in bright green) reduces comparatively to the whole workspace area (delimited by the red boundary boxes) as one demands more accuracy from the manipulator. For an accuracy of 2mm, 1mm, 0.5mm and 0.15mm, one gets and certified work area of 0.183 m^2, 0.174 m^2, 0.156 m^2 and 0.074 m^2, respectively.

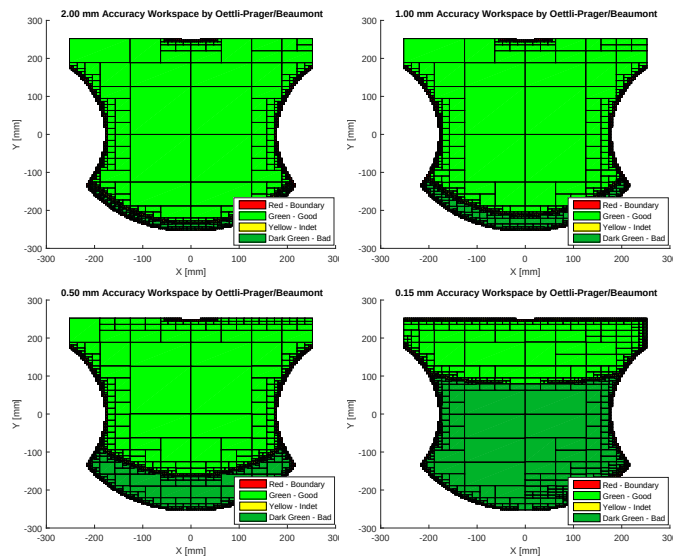

Fig. 6. Accuracy Workspace on plane z=310. Accuracy of 2mm, 1mm, 0.5mm and 0.15mm workspaces.

3) Force Workspace: Force workspace was obtained as a function of the manipulator's payload, and is shown in Figure 7. This figure shows the 2D workspaces for a payload of 0.5kg, 2kg, 3.5kg and 5kg, corresponding to certified work areas of 0.173 m^2, 0.140 m^2, 0.109 m^2 and 0.019 m^2, respectively. Average computation time was 92 seconds while the average number of intervals was 1136 (excluding reachable workspace boundary intervals shown as red boxes).

As the manipulator load is increased, the internal forces on the joints also increase to the point where, for some manipulator poses, they overcome the maximum achievable force to the actuated joints, thus reducing the useful area of work (shown in bright green).

IV. CASE STUDY - PKM DESIGN

For the PKM design case study, the defined goal is to find the sets of link length and platform width dimensions, which guarantee a workspace box of 200 by 200 by 40 millimeters, centered on the point (0,0,300), as shown in Figure 8, using the design routine of the algorithm. Inside such workspace, an accuracy of 1mm and a payload of 1kg, for all poses, must be ensured.

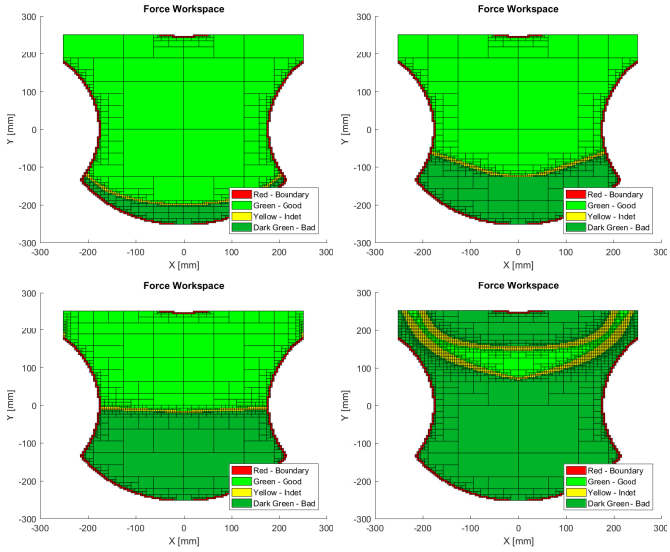

Fig. 7. Force Workspace on plane z=310. Payload of 0.5kg, 2kg, 3.5kg and 5kg workspaces.

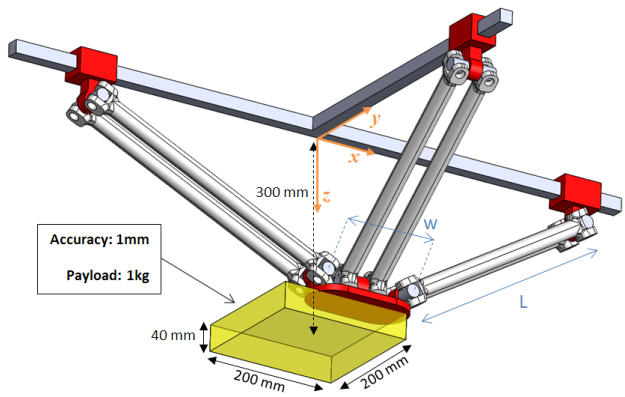

Fig. 8. PKM design case study parameters. Desired workspace is represented by the yellow box. Kinematic parameters to be determined are shown in blue.

The design algorithm ran in two fashions: the first one performing the constraint tests in serial (starting from joint limits constraints, then accuracy constraints and then force constraints) and the other one in parallel. When performing serial calculation, the first property constraint is applied to the entire search space. However, for the subsequent evaluations, only the inside boxes of the previous evaluation are used. The number of boxes evaluated is then much inferior to parallel calculation, where all search space is used for each property constraint. The final result is the intersection of each property evaluation. Results for serial and parallel calculations are shown in Figure 9 and Figure 10, respectively.

One can see that the accuracy and force workspace calculation times are much lower than joint range workspace. In fact, the bisection for the former only occurs for one of the design parameters (this particular robot's Jacobian matrices can be expressed only in terms of L and X coordinates of the end-effector with the end-effector width w bearing no influence

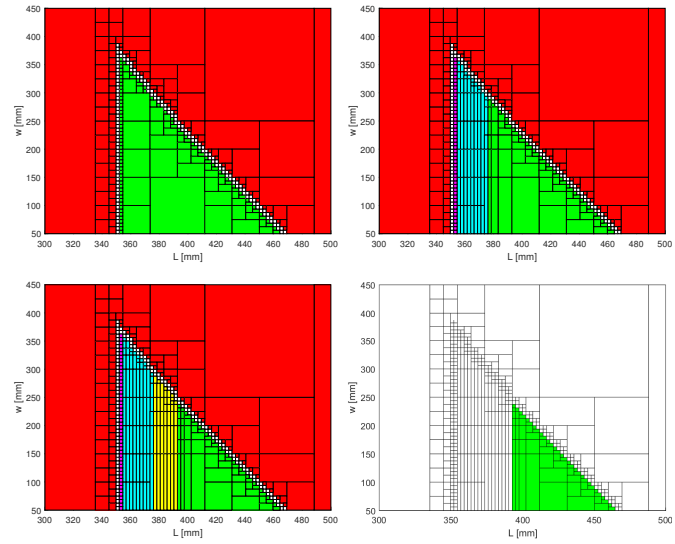

Fig. 9. Design Algorithm running in serial. Inside boxes shown in green; outside boxes shown in red and purple; boundary boxes shown in white, cyan and yellow. Last image shows the final result. Calculation times (from top left to bottom right): Joint Range and Singularity Constrains- 25.52s; Accuracy Constrains- 18.27s; Force Constraints- 6.50s; Total calculation time- 50.29s

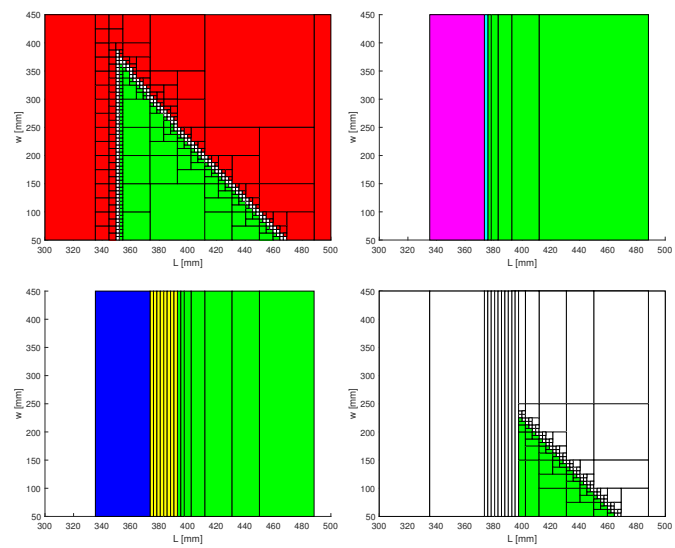

Fig. 10. Design Algorithm running in parallel. Inside boxes shown in green; outside boxes shown in red, purple and dark blue; boundary boxes shown in white, cyan and yellow. Last image shows the final result after intersection. Calculation times (from top left to bottom right): Joint Range and Singularity Constrains- 25.52s; Accuracy Constrains- 3.00s; Force Constraints- 1.63s; Total calculation time- 25.52s

on the results). For this reason, the number of boxes is much lower as well as the calculation times. In some calculations, the boundary areas do not seem to converge to zero. In fact, by reducing the size of the minimum box evaluated, both in the search space and variation domain of the parameters, one can make the boundary area converge to zero. Serial calculation is the most efficient when only one computer is available to run the algorithm. However, one can see that the final

calculation time for the parallel calculation is inferior to the serial calculation. This is because a distributed approach is employed, i.e. using a set of computers: a master program will manage the list $\mathscr{L}$ and send boxes to process to a free slave computer S. This slave computer is responsible for a single property constrain and evaluates its own boxes list $\mathscr{L}_S$ until either $\mathscr{L}_S$ is exhausted or that the number of boxes in $\mathscr{L}_S$ has reached a given threshold. Then the slave computer will return to the master program the list $\mathscr{L}_S$ (possible empty) that has to be processed together with the remaining sets (also possibly empty) of synthesis solutions. The result is that the calculation time is equal to the longer property evaluation time, in this case, the accuracy constraints.

However, a new serial run of the algorithm was performed, but this time the constraint evaluations were done in order of time efficiency, i.e., Force Constraints (1.63s), then Accuracy Constraints (0.69s) and finally Joint Range (10.71s), obtaining a final calculation time of 12,71s.

The efficiency of the bisection method adopted was also tested. Figure 11 shows the results for bisection Rule B (Hansen and Walster) and *Classical Method*. One can see that Rule B generates less boundary and outside boxes, meaning its more efficient as a bisection method. The average gain in space complexity is 58% relative to the *Classical Method*. This is also shown by the average 63% reduction on the calculation times, relative to the *Classical Method*. Rule C (Ratz) was also tested, and although it was not as efficient as Rule B, still an average improvement of 57% in calculation time and 56% in space complexity could be obtained, when comparing to the *Classical Method*. The reason why the *Classical Bisection Method* deems such bad results is thanks to not being able to detect the independency of the parameter w in the accuracy and force constraints evaluation. Thus it creates many more boxes by bisecting on two design variables. For this reason, and in problems of this nature, a good selection of the bisection method is crucial to obtain a fast and efficient algorithm.

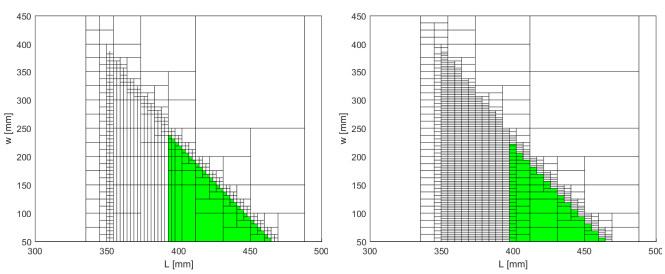

Fig. 11. Design Algorithm results for bisection Rule B (Hansen and Walster) and classical method. Inside boxes shown in green; outside and boundary boxes shown in white. Calculation times: Rule B- 50.29s; Classical Method- 184.39s

V. DISCUSSION

Parallel machine's performance is strongly dependent on their pose, so properties such as accuracy, rigidity and stiffness vary inside their workspace. It is then extremely important to know exactly how a PKM behaves on each region of its workspace since it might have great implications on the task it is supposed to perform. Imagine the case of a 3D printer or 3D laser scanning application, where one must know which regions of the workspace guarantee better accuracy, in order to correctly position the model, according to its features.

The method presented in this work proved to be a useful and efficient tool for the analysis and design study of the parallel manipulator, and can be used not only for this PKM but also for any other parallel architecture. With the results obtained, one can design any PKM machine being sure that its kinematic parameters ensure the required performance over its workspace. The advantage of having sets of values for these parameters is that one can choose the nominal values for the PKM kinematic parameters to be in the middle of these sets (as for instance in the middle of the green triangles shown in figures 9 and 10) thus having some margin of error to consider the manufacturing tolerances. These geometric errors, uncertainties and parasitic errors on non wanted DOF (orientation) as a function of the mechanical defaults, are unavoidable during the manufacturing process, and may not be compensated by calibration and may severely affect the overall behavior of the manipulator. This solution set also grants more liberty and flexibility to the design engineer to choose from a range of components and actuators for the robot. If any dimension or parameter needs to be changed, it is not necessary to do again the design from the beginning, as one can be sure that, as long as it is still inside the solution set, the design of the PKM is robust. If the obtained solution set is large or is very small, the designer can add new constraints, or relax the existing ones. A cost constraint function can also be easily added to the algorithm.

Failure of the algorithm may occur if the terms of the inclusion function have a very complex form. Indeed interval analysis will usually overestimate the ranges for these components and the size of this overestimation increase with the complexity of the analytical form of the terms. A consequence of this overestimation is that the procedure may fail to determine if all solutions of the linear systems are included in the set of solutions, even if the size of the ranges for the geometry and workspace parameters is small. Another possible cause of failure is not taking into account the dependency of the components in the inclusion function. While in this work, some strategies employed to improve the efficiency of the this algorithm are presented and discussed, the authors recon it still can be largely improved by using different filtering methods or different solvers, such as RSolver [49], [50], IBEX [51], [52] and Alias [53], although this is not explored here as it is not the main focus of this work. It should be interesting to compare this algorithm to other solvers in future publications of different nature, as well as try to improve the algorithm performance using other filtering methods or monotonicity tests.

A full scale prototype of the parallel mechanism, shown in Figure 12, is also being developed for testing and validation of the design and performance evaluation methods proposed here. This is the subject of the future second part of this work.

Fig. 12. PKM prototype developed at ISR-UC, using the design method proposed in this work.

VI. ACKNOWLEDGMENT

This research work was partially supported by the Portuguese Foundation of Science and Technology, contract SFRH/BD/94272/2013 and PTDC/EME-CRO/121547/2010.

REFERENCES

[1] S. Bhattacharya, H. Hatwal, and A. Ghosh, "On the optimum design of stewart platform type parallel manipulators," *Robotica*, vol. 13, no. 02, pp. 133–140, 1995.

[2] M. Badescu and C. Mavroidis, "Workspace optimization of 3-legged upu and ups parallel platforms with joint constraints," *Journal of Mechanical Design*, vol. 126, no. 2, pp. 291–300, 2004.

[3] A. G. Erdman, *Modern kinematics: developments in the last forty years*. John Wiley & Sons, 1993.

[4] I. Das and J. E. Dennis, "A closer look at drawbacks of minimizing weighted sums of objectives for pareto set generation in multicriteria optimization problems," *Structural optimization*, vol. 14, no. 1, pp. 63–69, 1997.

[5] M. Keler, "Dual expansion of an optimal in-parallel spherical platform device into a spatial one," in *Advances in Robot Kinematics: Analysis and Control*. Springer, 1998, pp. 79–86.

[6] W. Chen, M. M. Wiecek, and J. Zhang, "Quality utility—a compromise programming approach to robust design," *Journal of mechanical design*, vol. 121, no. 2, pp. 179–187, 1999.

[7] W. Chen, A. Sahai, A. Messac, and G. J. Sundararaj, "Exploration of the effectiveness of physical programming in robust design," *Journal of Mechanical Design*, vol. 122, no. 2, pp. 155–163, 2000.

[8] D. Chablat and P. Wenger, "Architecture optimization of a 3-dof translational parallel mechanism for machining applications, the orthoglide," *Robotics and Automation, IEEE Transactions on*, vol. 19, no. 3, pp. 403–410, 2003.

[9] H. S. Kim and L.-W. Tsai, "Design optimization of a cartesian parallel manipulator," *Journal of Mechanical Design*, vol. 125, no. 1, pp. 43–51, 2003.

[10] J.-P. Merlet and D. Daney, "Appropriate design of parallel manipulators," in *Smart Devices and machines for advanced manufacturing*. Springer, 2008, pp. 1–25.

[11] A. H. Land and A. G. Doig, "An automatic method of solving discrete programming problems," *Econometrica: Journal of the Econometric Society*, pp. 497–520, 1960.

[12] J. D. Little, K. G. Murty, D. W. Sweeney, and C. Karel, "An algorithm for the traveling salesman problem," *Operations research*, vol. 11, no. 6, pp. 972–989, 1963.

[13] L. Jaulin, *Applied interval analysis: with examples in parameter and state estimation, robust control and robotics*. Springer Science & Business Media, 2001, vol. 1.

[14] R. E. Moore, *Interval analysis*. Prentice-Hall Englewood Cliffs, 1966, vol. 4.

[15] F. Hao and J.-P. Merlet, "Multi-criteria optimal design of parallel manipulators based on interval analysis," *Mechanism and machine theory*, vol. 40, no. 2, pp. 157–171, 2005.

[16] J.-P. Merlet and D. Daney, "Dimensional synthesis of parallel robots with a guaranteed given accuracy over a specific workspace," in *Robotics and Automation, 2005. ICRA 2005. Proceedings of the 2005 IEEE International Conference on*. IEEE, 2005, pp. 942–947.

[17] M. Tannous, S. Caro, and A. Goldsztejn, "Sensitivity analysis of parallel manipulators using an interval linearization method," *Mechanism and Machine Theory*, vol. 71, pp. 93–114, 2014.

[18] J.-P. Merlet, "An improved design algorithm based on interval analysis for spatial parallel manipulator with specified workspace," in *Robotics and Automation, 2001. Proceedings 2001 ICRA. IEEE International Conference on*, vol. 2. IEEE, 2001, pp. 1289–1294.

[19] M. Kaloorazi, M. T. Masouleh, and S. Caro, "Interval-analysis-based determination of the singularity-free workspace of gough-stewart parallel robots," in *Electrical Engineering (ICEE), 2013 21st Iranian Conference on*. IEEE, 2013, pp. 1–6.

[20] S. Caro, D. Chablat, A. Goldsztejn, D. Ishii, and C. Jermann, "A branch and prune algorithm for the computation of generalized aspects of parallel robots," *Artificial Intelligence*, vol. 211, pp. 34–50, 2014.

[21] N. Ramdani, M. Gouttefarde, F. Pierrot, and J.-P. Merlet, "First results on the design of high speed parallel robots in presence of uncertainty," in *2008 IEEE/RSJ International Conference on Intelligent Robots and Systems*. IEEE, 2008, pp. 2410–2415.

[22] A. Goldsztejn, S. Caro, and G. Chabert, "A three-step methodology for dimensional tolerance synthesis of parallel manipulators," *Mechanism and Machine Theory*, vol. 105, pp. 213–234, 2016.

[23] C. Budde, P. Last, and J. Hesselbach, "Development of a triglide-robot with enlarged workspace," in *Robotics and Automation, 2007 IEEE International Conference on*. IEEE, 2007, pp. 543–548.

[24] R. Di Gregorio and V. Parenti-Castelli, "Geometric error effects on the performances of a parallel wrist," in *3rd Chemnitzer Parallelkinematik Seminar*, 2002, pp. 1011–1024.

[25] A. Goldsztejn, C. Michel, and M. Rueher, "Efficient handling of universally quantified inequalities," *Constraints*, vol. 14, no. 1, pp. 117–135, 2009.

[26] L. Jaulin and É. Walter, "Guaranteed tuning, with application to robust control and motion planning," *Automatica*, vol. 32, no. 8, pp. 1217–1221, 1996.

[27] S. Ratschan, "Efficient solving of quantified inequality constraints over the real numbers," *ACM Transactions on Computational Logic (TOCL)*, vol. 7, no. 4, pp. 723–748, 2006.

[28] D. Ratz and T. Csendes, "On the selection of subdivision directions in interval branch-and-bound methods for global optimization," *Journal of Global Optimization*, vol. 7, no. 2, pp. 183–207, 1995.

[29] A. Goldsztejn and L. Jaulin, "Inner and outer approximations of existentially quantified equality constraints," in *International Conference on Principles and Practice of Constraint Programming*. Springer, 2006, pp. 198–212.

[30] W. Oettli and W. Prager, "Compatibility of approximate solution of linear equations with given error bounds for coefficients and right-hand sides," *Numerische Mathematik*, vol. 6, no. 1, pp. 405–409, 1964.

[31] W. Oettli, "On the solution set of a linear system with inaccurate coefficients," *Journal of the Society for Industrial and Applied Mathematics: Series B, Numerical Analysis*, pp. 115–118, 1965.

[32] O. Beaumont, "Algorithmique pour les intervalles: Comment obtenir un résultat sûr quand les données sont incertaines," Ph.D. dissertation, Université de Rennes, Inria, project ALADIN, École Doctorale Informatique, Traitement du Signal et Télécommunications, 1 1999.

[33] C. Gosselin and J. Angeles, "Singularity analysis of closed-loop kinematic chains," *Robotics and Automation, IEEE Transactions on*, vol. 6, no. 3, pp. 281–290, 1990.

[34] G. Rex and J. Rohn, "Sufficient conditions for regularity and singularity of interval matrices," *SIAM Journal on Matrix Analysis and Applications*, vol. 20, no. 2, pp. 437–445, 1998.

[35] C. Jansson and J. Rohn, "An algorithm for checking regularity of interval matrices," *SIAM Journal on Matrix Analysis and Applications*, vol. 20, no. 3, pp. 756–776, 1999.

[36] J.-P. Merlet and D. Daney, "A formal-numerical approach to determine the presence of singularity within the workspace of a parallel robot." in *In 2nd Workshop on Computational Kinematics*, 2001, pp. 167–176.

[37] C. Brisan, D. Franitza, and M. Hiller, "Modelling and analysis of errors for parallel robots," in *1st Int. Colloquium, Collaborative Research Centre*, vol. 562, 2002, pp. 83–96.

[38] C. Han, J. Kim, J. Kim, and F. C. Park, "Kinematic sensitivity analysis of the 3-upu parallel mechanism," *Mechanism and Machine Theory*, vol. 37, no. 8, pp. 787–798, 2002.

[39] C.-S. Han, J. C. Hudgens, D. Tesar, and A. E. Traver, "Modeling, synthesis, analysis, and design of high resolution micromanipulator to enhance robot accuracy," in *Intelligent Robots and Systems' 91.'Intelligence for Mechanical Systems, Proceedings IROS'91. IEEE/RSJ International Workshop on*. IEEE, 1991, pp. 1157–1162.

[40] H. S. Kim and Y. J. Choi, "The kinematic error bound analysis of the stewart platform," *Journal of Robotic Systems*, vol. 17, no. 1, pp. 63–73, 2000.

[41] O. Masory, J. Wang, and H. Zhuang, "On the accuracy of a stewart platform. ii. kinematic calibration and compensation," in *Robotics and Automation, 1993. Proceedings., 1993 IEEE International Conference on*. IEEE, 1993, pp. 725–731.

[42] V. Parenti-Castelli and R. Di Gregorio, "Influence of manufacturing errors on the kinematic performance of the 3-upu parallel mechanism," in *2nd Chemnitzer Parallelkinematik Seminar*, 2000, pp. 85–99.

[43] A. J. Patel and K. Ehmann, "Volumetric error analysis of a stewart platform-based machine tool," *CIRP Annals-Manufacturing Technology*, vol. 46, no. 1, pp. 287–290, 1997.

[44] T. Ropponen and T. Arai, "Accuracy analysis of a modified stewart platform manipulator," in *Robotics and Automation, 1995. Proceedings., 1995 IEEE International Conference on*, vol. 1. IEEE, 1995, pp. 521–525.

[45] J. Ryu and J. Cha, "Volumetric error analysis and architecture optimization for accuracy of hexaslide type parallel manipulators," *Mechanism and Machine Theory*, vol. 38, no. 3, pp. 227–240, 2003.

[46] C. Tischler and A. Samuel, "Predicting the slop of in-series/parallel manipulators caused by joint clearances," in *Advances in robot kinematics: analysis and control*. Springer, 1998, pp. 227–236.

[47] J. Wang and O. Masory, "On the accuracy of a stewart platform. i. the effect of manufacturing tolerances," in *Robotics and Automation, 1993. Proceedings., 1993 IEEE International Conference on*. IEEE, 1993, pp. 114–120.

[48] J. Hubert and J.-P. Merlet, "Singularity analysis through static analysis," in *Advances in Robot Kinematics: Analysis and Design*. Springer, 2008, pp. 13–20.

[49] S. Ratschan, "Efficient solving of quantified inequality constraints over the real numbers," *ACM Transactions on Computational Logic*, vol. 7, no. 4, pp. 723–748, 2006.

[50] S. Ratschan *et al.*, "Rsolver," `http://rsolver.sourceforge.net`, 2004, software Package.

[51] G. Chabert and L. Jaulin, "Contractor programming," *Artificial Intelligence*, vol. 173, no. 11, pp. 1079–1100, 2009.

[52] G. Chabert, "A simple slam example with ibex," in *Small Workshop on Interval Methods, SWIM*, 2013.

[53] J.-P. Merlet, "Alias: an interval analysis based library for solving and analyzing system of equations," *SEA, June. Automation*, pp. 1964–1969, 2000.