AN INTERVAL ANALYSIS METHOD FOR WRENCH WORKSPACE DETERMINATION OF PARALLEL MANIPULATOR ARCHITECTURES

Joshua K. Pickard[1], Juan A. Carretero[1]
[1]*Department of Mechanical Engineering, University of New Brunswick, Fredericton, NB, Canada*
Email: joshp.unb@gmail.com; Juan.Carretero@unb.ca

ABSTRACT

This paper deals with the wrench workspace (WW) determination of parallel manipulators. The WW is the set of end-effector poses (positions and orientations) for which the active joints are able to balance a set of external wrenches acting at the end-effector. The determination of the WW is important when selecting an appropriate robotic design since the size and shape of the WW are dependent on the robot's geometry (design) and selected actuators. Algorithms for the determination of the reachable workspace and the WW are presented. The algorithms are applicable to robotic architectures utilizing actuators with positive and negative limits on the force/torque they can generate, as well as cable-driven parallel manipulator architectures which require nonnegative actuator limits to maintain positive cable tensions. The approaches used in this paper provide guaranteed results and are based on methods utilizing interval analysis techniques for the representation of end-effector poses and design parameters.

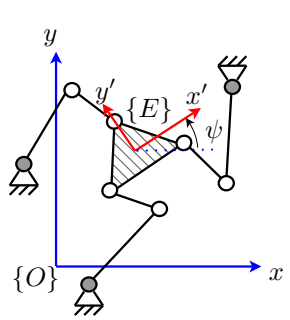
Keywords: interval analysis; wrench capability; wrench workspace.

1. INTRODUCTION

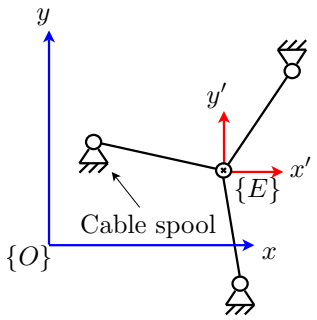
Parallel manipulators (PMs) are a classification of robotic mechanism consisting of a moving platform which is connected to a fixed base through multiple serial kinematic chains (limbs) forming a closed-loop architecture, as illustrated in Figure 1. The closed-loop architecture allows n-degrees-of-freedom (DOF) motions using m active joints, where m must be equal to or greater than n. All remaining joints are passive. PMs tend to have a) high load carrying capacity as the total load can be shared by the limbs acting in parallel, b) low inertia of most moving parts due to the heavy actuators typically being located near or at the fixed base, and c) high structural stiffness. However, PMs tend to have smaller and less dexterous workspaces due to link interferences, internal singular configurations, and physical constraints. Cable-driven parallel manipulators (CDPMs) are a special class of PM where the moving platform is connected in parallel to fixed actuated spools by cables (see Figure 1b). If each cable is modelled as a rigid-body, the CDPM in Figure 1b is kinematically equivalent to the rigid-link 3-RPR PM. Actuation of the cable lengths allows for control of the moving platform and generation of end-effector wrenches. A limiting factor of a CDPM is that the cables are non-rigid members and therefore require nonnegative cable tensions to constrain the robot's moving platform. CDPMs have several benefits over rigid-link PM architectures, such as a very large reachable workspace (since large cable lengths can be used), low visual intrusion, low limb mass (since each limb consists of only a cable), rapid deployment and easy reconfigurability.

1.1. Task Completion

When selecting a robot for a specific task, it is important to select an architecture with an appropriate design which is capable of: generating desired end-effector wrenches (*i.e.*, force/moment couples, herein referred to as simply wrenches), and traversing desired trajectories. Several papers have considered the workspace analysis for a task, given a required set of wrenches described geometrically as a point (a single wrench) [2, 5, 7], a solid hyper-ellipse [2, 3, 8], a solid hyper-rectangle [10], or a solid convex polytope [2, 7]. Point representation is important when obtaining a CDPM's static workspace (the set of poses of the moving



(a) 3-RRR architecture with rigid links and 3-DOF (x, y, and ψ).



(b) 2-3 cable-driven parallel manipulator with 2-DOF (x, y) and 3 cables.

Fig. 1. Parallel manipulator architectures.

platform for which the cables can balance the weight of both the platform and the payload with tension forces only), as the single wrench corresponds to the combined platform and payload weight; the importance of the hyper-ellipse representation is that the force and moment contributions of a wrench can be modelled independently as being isotropic; *e.g.*, a force of magnitude 1 N in any direction and a moment of magnitude 0.5 Nm in any direction; the hyper-rectangle representation independently defines the lower and upper bounds of the desired forces and moments and is suitable for interval analysis techniques; and a polytopic representation allows for complex descriptions of task wrench requirements described by the intersection of a set of half-spaces.

The usable workspace for the robot when considering a desired task is termed the *wrench workspace*. A similar term, *wrench-feasible workspace*, is typically given to the workspace of a CDPM which considers nonnegative cable tensions. The concept of a wrench workspace has been studied in [3, 6–9, 18, 21]. An interval-based method [10, 12] and a convex-hull method [2, 3, 6, 7] are two current techniques for numerically obtaining the wrench workspace. A comparison of the two techniques applied to the 3-R$\underline{\text{P}}$R CDPM is provided in [17]. The term wrench workspace (WW) will be used throughout this paper to refer to both CDPMs with nonnegative cable tensions and rigid-link PM architectures for the completion of desired tasks. Determination of the WW is commonly performed by considering a grid of end-effector poses, where each pose is tested for being wrench-capable (*i.e.*, the robot is able to generate the desired wrenches at the given pose). The grid is used to discretize the search space and each discretized pose is evaluated for being wrench-capable. Discretization provides a straightforward algorithm for determining a set of wrench-capable poses which collectively form the WW. However, the solution set of poses only provides an approximation of the true WW. This is because out of the infinitely many poses of the discretized search space, only a finite quantity – those on the discretization grid – are tested; therefore, all of the poses of the grid may be wrench-capable but some poses which are not wrench-capable may be missed, especially since the WW typically has a non-convex geometry, may contain holes, and may also be separable. Gouttefarde et al. [10] state *"the result provided by a discretization can never be guaranteed, i.e., one can never know if this result can be trusted."*.

1.2. Interval Analysis Techniques

This paper expands on the work by Gouttefarde et al. [10] on the guaranteed determination of a CDPMs wrench-feasible workspace using interval analysis techniques. Interval analysis techniques allow computations using an interval representation for variables. These techniques provide an alternative to the typical discretization methods since they allow an infinite set of poses to be represented as a single interval. Classification tests can then be used on the pose intervals and can provide high resolution and guaranteed WW

determinations. The solution is guaranteed since each of the infinitely many poses are explored. Interval analysis also has an added benefit of accounting for numerical rounding errors by properly representing a floating point value by an interval which tightly contains the desired value. The verification of a pose interval being wrench-capable is based on an inside test which ensures feasibility for a system of interval linear equations using two theorems proposed by Rohn [19] for strong feasibility (for CDPMs with nonnegative cable tensions) and strong solvability. A complementary outside test is used to determine if a pose interval falls completely outside of the WW, *i.e.*, no pose inside the pose interval can generate the desired wrenches. The outside test was proposed in [10] and utilizes interval filtering or consistency techniques and provides guaranteed results that the interval under inspection is completely outside of the WW.

The contribution of this paper is to extend the interval analysis algorithm originally proposed for CDPMs with nonnegative cable tensions by Gouttefarde et al. [10] to general PM architectures utilizing actuators with both positive and negative torque/force limits (capabilities). Section 2 introduces interval arithmetic and describes general solving procedures for the interval evaluation of a function, filtering techniques which can be applied to the function to enforce consistency in the variables, and a branch-and-bound method for improving the interval solution. The *reachable workspace* evaluation is described in Section 3 and is based on the algorithm proposed by Oetomo et al. [16] which describes an interval analysis method to certify the reachable workspace of a flexure-based precision mechanism using the forward and inverse kinematics combined with interval constraint satisfaction techniques. This algorithm is easily adaptable to many kinematic architectures, with various kinematic chains, expressed as constraints in terms of mathematical equalities/inequalities and is able to manage parameter uncertainties (*e.g.*, the length of the proximal and distal links are specified as $l_i \pm \delta$ to model flexure-joint deflections). The implementation of this algorithm for general PM architectures is presented which consists of an inside test which is able to efficiently determine the set of pose intervals which are completely inside the reachable workspace and an outside test to determine the set of pose intervals which are completely outside the reachable workspace. The WW is described in Section 4 and an algorithm for determining the set of pose intervals which are inside and outside the WW is provided. The examples in this paper are coded in C++ and use the interval arithmetic and HC4 constraint propagation loop of the Ibex C++ library, and the simplex method of the Computational Infrastructure for Operations Research (COIN) library.

2. INTERVAL-BASED KINEMATICS

2.1. Interval Arithmetic

Interval analysis is a mathematical framework which allows for a computation using interval quantities, such that an interval variable $[x]$ denotes the natural extension of the closed interval $[x] = [\underline{x}, \overline{x}] = \{x \mid x \in \mathbb{R}, \underline{x} \leq x \leq \overline{x}\}$. A fundamental feature of interval analysis is the interval evaluation of a function which yields a closed interval bounding the set of solutions. The evaluation of the function $f(x)$ over the interval $[x]$ yields an interval $[f]$ which encloses the image of $[x]$ under $f([x])$. The function $[f]$ is called an *inclusion function* for $f([x])$, such that $f([x]) = \{f(x) \mid x \in [x]\} \subseteq [f]$. The converse inclusion does not hold in general, and $[f]$ overestimates $f([x])$, thereby introducing pessimism in the evaluation [10]. The overestimation caused by the interval evaluation $f([x])$ is known as the *wrapping effect* [11, 15]. This states that there exists solutions in $[f]$ which are not solutions to the original problem. Thus, $[f]$ does not accurately represent the problem solution and instead only provides boundaries of the solution. The *dependency problem* [11] is another source of overestimation which is caused by an interval variable appearing multiple times in a calculation. Each occurrence is taken independently which can lead to an unwanted expansion of the resulting interval. Cancellation or reduction of the number of occurrences of a variable before interval evaluation can reduce interval widths [14] (*e.g.*, if $[x] = [-2,2]$, then the interval solution for $[x]^2 = [0,4]$, whereas $[x] * [x] = [-4,4]$).

2.2. Interval Classification

The goal of this work is to evaluate a PM in terms of its wrench capabilities, *i.e.*, the entire set of wrenches that the robot can generate given certain actuator capabilities. $[\mathbf{p}]$ is defined as the interval pose vector containing the task space variables (*e.g.*, x, y, and ψ for the 3-RRR PM). A Cartesian product of intervals is typically referred to as a *box*. Interval analysis allows us to model other variables as interval quantities, such as, actuator placement, and link lengths. A common strategy in interval analysis is to classify a box into one of three categories: *inside*, *outside*, or *boundary*. As an example, consider an inequality representing a constraint ($C([\mathbf{p}],[\mathbf{h}]) \leq 0$) to be satisfied over the search space $\mathcal{S} \subset \mathbb{R}^n$, where $[\mathbf{h}]$ is a box containing the design parameters of the manipulator, such as actuator placements, link lengths, and actuator capabilities. The problem can be formulated such that

$$\begin{aligned} \textit{inside boxes} &= \{[\mathbf{p}] \in \mathcal{S} \mid \forall \mathbf{p} \in [\mathbf{p}],\ \forall \mathbf{h} \in [\mathbf{h}],\ C([\mathbf{p}],[\mathbf{h}]) \leq 0\} \\ \textit{outside boxes} &= \{[\mathbf{p}] \in \mathcal{S} \mid \forall \mathbf{p} \in [\mathbf{p}],\ \forall \mathbf{h} \in [\mathbf{h}],\ C([\mathbf{p}],[\mathbf{h}]) > 0\} \\ \textit{boundary boxes} &= \{[\mathbf{p}] \in \mathcal{S} \mid \forall \mathbf{p} \in [\mathbf{p}],\ \forall \mathbf{h} \in [\mathbf{h}],\ \inf(C([\mathbf{p}],[\mathbf{h}])) \leq 0,\ \sup(C([\mathbf{p}],[\mathbf{h}])) > 0\} \end{aligned} \tag{1}$$

Due to the overestimation of the interval evaluation of $C([\mathbf{p}],[\mathbf{h}])$ caused by the wrapping effect and dependency problem, it is necessary to apply interval filtering techniques (contractors) to the constraints such that consistency in the interval variables is enforced in the evaluation of the constraint. Filtering techniques use additional information contained in the mathematical equations or through the use of additional physical constraints to sharpen the resulting solution interval. Oetomo et al. [16] propose the use of the forward kinematic equations as physical constraints in addition to the inverse kinematic equations when obtaining the reachable workspace. The Ibex C++ library provides a classical constraint programming algorithm known as the forward-backward contractor (or HC4Revise [20]), which is used in this work. Other filtering techniques, such as the 2-B-consistency and 3-B-consistency [13] local consistency techniques are effective methods with reasonable computation times.

It can often be difficult to conclude whether a given interval satisfies the requirements for being an inside or outside box for interval variables with a large width. Filtering techniques contract the width of a given box in an attempt to obtain a sharp solution; however, it can only return the sharpest box which bounds the solution. The actual solution may only occupy a portion of this box. The *branch-and-bound* strategy performs an automated bisection routine which is applied to all boundary boxes following the classification routine. Each boundary box is split according to a bisection strategy. A common strategy is the *largest first* strategy which bisects the box equally along the dimension (interval variable) with the largest width. With such a strategy, each interval variable is bisected in turn and the size of the box is continually reduced until an inside or outside box is found. This procedure is applied to all boundary boxes until the maximum width of any remaining boundary boxes is smaller than a desired threshold. Unless otherwise specified, the threshold, ε, used in the examples throughout this paper is $\varepsilon = 0.01$ m.

3. REACHABLE WORKSPACE EVALUATION – 3-RRR EXAMPLE

The reachable workspace (*RW*) for a PM is defined as the set of end-effector poses $\mathbf{p}$ that a robotic architecture is able to reach for a given set of design parameters $\mathbf{h}$. The *RW* for a PM can be found by solving the inverse kinematics for the robot and ensuring that the pose has a real solution. CDPMs have the added complexity that the cables must maintain a positive tension, thus requiring an analysis which combines kinematics with statics for determining the CDPM's *RW*. A CDPM's *RW* can be determined using the *WW* algorithm in Section 4 which accounts for the nonnegative cable tensions.

The 3-RRR PM architecture (see Figure 1a and Figure 2) consists of three limbs, each consisting of three revolute joints and two links, attaching the moving platform to the fixed base. The inverse kinematics can be solved in terms of a pose $\mathbf{p}$ to give the joint angles, α_i, β_i, γ_i, for each limb i. Each limb is given an individual

reference frame $\{O_i\}$ located at the limb's fixed base position with the same orientation as the base frame $\{O\}$. The origin of each reference frame will be defined as Point O_i. Point $P = (x_e, y_e)^T$ represents the location of the end-effector in terms of the base frame, where $\mathbf{p}_e = (x_e, y_e)^T$ is the vector pointing from the origin of $\{O\}$ (O) to P. The robot's complete pose will be defined as $\mathbf{p} = (x_e, y_e, \psi)^T$. $\mathbf{R}_z(\psi)$ decribes the transformation from the end-effector frame $\{E\}$ to frame $\{O\}$. The position of the moving platform attachment point, ${}^{0_i}\mathbf{b}_i$, expressed in terms of frame $\{0_i\}$ is given by

$$ {}^{O_i}\mathbf{b}_i = \mathbf{b}_i - O_i = \mathbf{p}_e + \mathbf{R}_z(\psi) \cdot {}^{E}\mathbf{d}_i - O_i \tag{2} $$

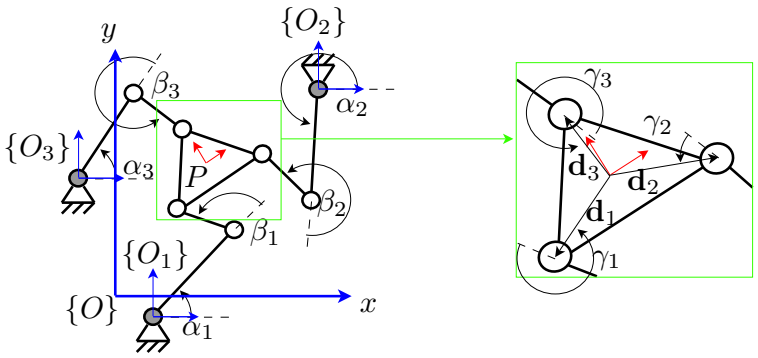


Fig. 2. 3-RRR kinematic diagram.

3.1. Inverse Kinematics Problem

A Constraint Satisfaction Problem (CSP) can be used to define the inverse kinematics for a PM in the form of a set of equality and inequality constraints and a set of interval variables. Generally, the inverse kinematics solution is obtained by first calculating the angle β_i for limb i, which has two possible solutions – elbow-left and elbow-right configurations. By selecting the elbow-right configuration, $\cos(\beta_i)$ can be evaluated (along with $\sin(\beta_i) = \sin(\cos^{-1}(\cos(\beta_i)))$). The joint angle α_i can then be calculated using both the sine and cosine. The length of the proximal and distal links are r_i and l_i, respectively. The closed-form solution of the inverse kinematics is given as follows:

$$ \begin{aligned} \cos(\beta_i) &= ({}^{O_i}b_{ix}^2 + {}^{O_i}b_{iy}^2 - (r_i^2 + l_i^2))/(2l_i r_i) \\ \cos(\alpha_i) &= ({}^{O_i}b_{ix}(r_i + l_i\cos(\beta_i)) + {}^{O_i}b_{iy} l_i \sin(\beta_i))/({}^{O_i}b_{ix}^2 + {}^{O_i}b_{iy}^2) \\ \sin(\alpha_i) &= (-{}^{O_i}b_{ix} l_i \sin(\beta_i) + {}^{O_i}b_{iy}(r_i + l_i\cos(\beta_i))/({}^{O_i}b_{ix}^2 + {}^{O_i}b_{iy}^2) \end{aligned} \tag{3} $$

Since $[\mathbf{p}] = ([x_e], [y_e], [\psi])^T$ is an interval, the solutions to the inverse kinematic equations in (5) are also intervals which must satisfy certain trigonometric constraints defined by:

$$ \begin{aligned} C_1([\mathbf{p}], [\mathbf{h}]) &= \cos(\beta_i) \in [\underline{\cos(\beta_i)}, \overline{\cos(\beta_i)}] \subseteq [-1, 1] \quad \rightarrow \beta_i \in [0, \pi] \text{ (elbow-right)} \\ C_2([\mathbf{p}], [\mathbf{h}]) &= \cos(\alpha_i) \in [\underline{\cos(\alpha_i)}, \overline{\cos(\alpha_i)}] \subseteq [-1, 1] \\ C_3([\mathbf{p}], [\mathbf{h}]) &= \sin(\alpha_i) \in [\underline{\sin(\alpha_i)}, \overline{\sin(\alpha_i)}] \subseteq [-1, 1] \end{aligned} \tag{4} $$

The box $[\mathbf{p}]$ is classified as an inside box when all of the constraints in Eq. (4) are satisfied, and is classified as an outside box if any of the inverse kinematic solutions falls in the complement of Eq. (4). Additional constraints can be imposed on the robot by adjusting the interval bounds of the constraints in Eq. (4). For example, poses where the proximal and distal links overlap can be eliminated by setting $\overline{cos(\beta_i)} = \rho$, where

Algorithm 1 Compute the inverse kinematics with interval analysis

Type, [**k**] (optional) = COMPUTE_INVERSEKINEMATICS([**p**], [**h**])

for all i in $C_i([\mathbf{p}],[\mathbf{h}])$ **do**
  try
    $[k_i] = C_i([\mathbf{p}],[\mathbf{h}])$
  catch
    return $Type = 0$
  if CONSTRAINT_SATISFIED$(i,[k_i])== -1$ **then**
    return $Type = -1$ ▷ Classify as outside box
[**k**] = FILTERING([**k**])
if all CONSTRAINT_SATISFIED$(i,[k_i])== 1$ **then**
  return ($Type = 1$, [**k**]) ▷ Classify as inside box
else if any CONSTRAINT_SATISFIED$(i,[k_i])== -1$ **then**
  return $Type = -1$ ▷ Classify as outside box
else
  return $Type = 0$ ▷ Classify as boundary box

ρ specifies the maximum allowable range of β_i. The *RW* of the mechanism is described by the union of the set of inside boxes. As explained by Gouttefarde et al. [10], a pose **p** which lies on the boundary of the reachable workspace belongs neither to the set of inside or outside boxes, and, thus necessarily belongs to the set of boundary boxes. Therefore, the *RW* boundary is necessarily contained in the set of boundary boxes.

An algorithm for computing the inverse kinematics problem (IKP) is proposed in Algorithm 1. The inputs for the algorithm are the current pose $[\mathbf{p}]$ and the design parameters $[\mathbf{h}]$. The algorithm returns the box classification, and the inverse kinematics solution if the box is classified as inside. In certain cases, the evaluation of a constraint may fail due to interval related issues (*e.g.*, division by zero, trigonometric bounds exceeded); therefore, a try–catch statement classifies a failed evaluation as a boundary box. The function CONSTRAINT_SATISFIED$(i,[k_i])$ determines if constraint i is satisfied over the interval $[k_i]$. This function returns a 1 if the constraints are all completely satisfied, -1 if any constraint is completely dissatisfied, and 0 otherwise. The function FILTERING$([\mathbf{k}])$ attempts to filter the inverse kinematics solutions $[\mathbf{k}]$ using a combination of inverse and direct kinematic equations and interval filtering techniques (*e.g.*, HC4Revise). Oetomo et al. [16] thoroughly explain interval filtering using the inverse and direct kinematics.

3.2. Reachable Workspace Algorithm

The *RW* determination algorithm is summarized in Algorithm 2. Algorithm 2 is applicable to PM achitectures which have a closed form for the inverse kinematics; however, it is not directly applicable to CDPMs due to their requirement for nonnegative cable tensions. Several lists are used to store classified pose intervals including $\mathcal{L}_{inside}$, $\mathcal{L}_{outside}$, $\mathcal{L}_{boundary}$, and also the list $\mathcal{L}$ to store unclassified intervals. The function EXTRACT$(\mathcal{L})$ extracts an interval from the top of the list. If an interval is classified as a boundary box and the width of the box exceeds the threshold ε, the function BISECT$([\mathbf{p}])$ bisects the boundary box into two smaller boxes as previously decribed in Section 2.2. The symbol $\leftarrow$ denotes insertion of an element to the bottom of the list.

Figure 3a provides a plot of the *RW* for the 3-RRR PM with constant orientation, *i.e.*, $[\psi] = [0,0]$, for link lengths $l_i = 0.3$ *m*, $r_i = 0.2$ *m*. The design parameters used are: ${}^E\mathbf{d}_1 = (0.2,0.0)^T$ m, ${}^E\mathbf{d}_2 = (-0.1,0.1732)^T$ m, ${}^E\mathbf{d}_3 = (-0.1,-0.1732)^T$ m, $O_1 = (0.4,0.0)^T$ m, $O_2 = (-0.2,0.3464)^T$ m, and $O_3 = (-0.2,-0.3464)^T$ m. The *RW* boundaries are completely contained within the set of boundary boxes and interior poses located on the *RW* boundaries are appropriately detected.

Algorithm 2 Reachable workspace evaluation with interval analysis

$\mathcal{L}_{inside}, \mathcal{L}_{outside}, \mathcal{L}_{boundary} =$ COMPUTE_REACHABLEWORKSPACE($\mathcal{S}$, [**h**], ε)

Initialize empty lists $\mathcal{L}$, $\mathcal{L}_{inside}$, $\mathcal{L}_{outside}$, and $\mathcal{L}_{boundary}$
$\mathcal{L} \leftarrow \mathcal{S}$ ▷ Initialize list $\mathcal{L}$ with search box $\mathcal{S}$
while $\mathcal{L} \neq \{\emptyset\}$ **do**
  [**p**] $\leftarrow$ EXTRACT($\mathcal{L}$) ▷ Select box from top of list $\mathcal{L}$
  $(Type, [\mathbf{k}]) =$ COMPUTE_INVERSEKINEMATICS([**p**], [**h**])
  if $Type == 1$ **then**
    $\mathcal{L}_{inside} \leftarrow$ [**p**], [**k**] ▷ [**p**] $\subseteq RW$, insert [**p**] and [**k**] to bottom of *inside* list
  else if $Type == -1$ **then**
    $\mathcal{L}_{outside} \leftarrow$ [**p**] ▷ [**p**] $\nsubseteq RW$, insert [**p**] to bottom of *outside* list
  else
    if Width([**p**])$> \varepsilon$ **then**
      $\mathcal{L} \leftarrow$ BISECT([**p**])
    else ▷ [**p**] is too small to be bisected
      $\mathcal{L}_{boundary} \leftarrow$ [**p**] ▷ insert [**p**] to bottom of *boundary* list

4. WRENCH WORKSPACE EVALUATION

4.1. Task Description

A task requires a robot to be able to generate certain wrench sets and be able to continuously apply/sustain them throughout a trajectory. A *WW* describes the portion of the robot's *RW* where a particular task can be performed. The term *wrench capability* (*WC*) is given to the set of wrenches that a robot can generate at its end-effector at a given pose. Due to its polytopic geometry [1], a *WC* can be described by a closed set represented by an intersection of half-spaces, where the *WC* at a pose **p** is denoted by $WC(\mathbf{p})$. Similarly, the task wrench set is typically represented by a closed set, which is termed the *minimum allowable wrench capability* (*MAWC*) [17], which is used to define the minimum wrench set required for a specific task. That is, the *WC* at each pose inside the *WW* must satisfy the *MAWC* in order to be able to generate the wrenches required by the task. In terms of interval analysis, the *WW* is defined as:

$$WW = \{[\mathbf{p}] \mid [\mathbf{p}] \in RW,\ \forall\, \mathbf{p} \in [\mathbf{p}],\ MAWC \subseteq WC(\mathbf{p})\} \tag{5}$$

4.2. Interval Evaluation of the Jacobian Matrix

Interval analysis provides a useful alternative to conventional discretization techniques due to its ability to represent an infinite set of poses in terms of an interval vector [**p**]. Since the Jacobian matrix, $\mathbf{J} = {\mathbf{J_q}}^{-1}\mathbf{J_x}$ ($\mathbf{J_q}$ is the inverse Jacobian and $\mathbf{J_x}$ is the direct Jacobian), is pose-dependent, each element J_{ij} of **J** is interval evaluated over the pose [**p**], thereby yielding an interval $[J_{ij}]$. The $m \times n$ interval matrix [**J**] whose elements are the intervals $[J_{ij}]$ has the fundamental property $\forall\, \mathbf{p} \in [\mathbf{p}],\ \mathbf{J}(\mathbf{p}) \in [\mathbf{J}]$. In other words, for every pose $\mathbf{p} \in [\mathbf{p}]$, the Jacobian matrix obtained for **p** belongs to [**J**]. Consequently, the interval matrix [**J**] overestimates the set $\{\mathbf{J}(\mathbf{p}) \mid \mathbf{p} \in [\mathbf{p}]\}$. There exists some matrices $\mathbf{J}_0 \in [\mathbf{J}]$ where $\forall\, \mathbf{p} \in [\mathbf{p}],\ \mathbf{J}_0 \neq \mathbf{J}(\mathbf{p})$; this is due to the wrapping effect [10]. Note that the Jacobian matrix is also a function of the design parameters [**h**], *i.e.*, $\mathbf{J}([\mathbf{p}], [\mathbf{h}])$, such that $\forall\, \mathbf{p} \in [\mathbf{p}],\ \forall\, \mathbf{h} \in [\mathbf{h}],\ \mathbf{J}(\mathbf{p}, \mathbf{h}) \in [\mathbf{J}]$.

Each active joint is capable of supplying a force/torque τ_i. If $\boldsymbol{\tau}$ is a vector containing the actuator forces/torques of all active joints, then $[\boldsymbol{\tau}]$ represents the capabilities of all *m* actuators. Assuming that the *MAWC* is represented in the form of an interval, the *WW* can be represented by the forward-force solution in the form of a *system of interval linear equations*, such that at each pose:

$$[\mathbf{J}]^T \boldsymbol{\tau} = [MAWC],\ \boldsymbol{\tau} \in [\underline{\boldsymbol{\tau}}, \overline{\boldsymbol{\tau}}] \tag{6}$$

which amounts to testing infinitely many linear systems such that

$$\forall\, \mathbf{J} \in [\mathbf{J}],\ \forall\, \mathbf{f} \in [MAWC],\ \exists\, \boldsymbol{\tau} \in [\underline{\boldsymbol{\tau}}, \overline{\boldsymbol{\tau}}] \mid \mathbf{J}^T \boldsymbol{\tau} = \mathbf{f} \tag{7}$$

where $\mathbf{f}$ denotes a wrench. Eq. (7) states that the entire set of required wrenches (*i.e.*, $[MAWC]$) can be generated for all $\mathbf{J} \in [\mathbf{J}]$ given the actuator capabilities $[\boldsymbol{\tau}]$. In order for $[\mathbf{J}]$ to be finite, it is important that $[\mathbf{J_q}]$ be invertible, such that $\forall\, \mathbf{J_q} \in [\mathbf{J_q}],\ \exists\, \mathbf{J_q}^{-1}$.

4.3. System of Interval Linear Equations with Bounded Solutions

According to Rohn [19], a system of linear equations ($\mathbf{Ax} = \mathbf{b}$) is called a) *solvable* if it has a solution and b) *feasible* if it has a nonnegative solution, *i.e.*, feasibility implies nonnegative solvability. The system of interval linear equations $[\mathbf{A}]\mathbf{x} = [\mathbf{b}]$ is understood to represent the family of all systems of linear equations $\mathbf{Ax} = \mathbf{b},\ \mathbf{A} \in [\mathbf{A}],\ \mathbf{b} \in [\mathbf{b}]$. The interval system is said to be *strongly solvable* (*strongly feasible*) if each subsystem is solvable (feasible). Eq. (6) is a system of interval linear equations with a bounded solution, $\boldsymbol{\tau} \in [\boldsymbol{\tau}]$, such that the interval system is strongly solvable (strongly feasible) and also accounts for the actuator capabilities. It is important to use the proper theorem to ensure that each of the infinitely many linear systems are verified. The strong feasibility test can be applied directly to CDPMs since they do not have nonnegative cable tensions, whereas PMs with revolute or prismatic actuators, for example, can generate both positive and negative torques/forces and thus require analysis using the strong solvability theorem.

4.3.1. Vertex Matrices and Vertex Vectors

The strong solvability and strong feasibility theorem by Rohn [19] require a vertex representation for the system of interval linear equations, where Y_n is the set of 2^n unique n-dimensional vectors $\mathbf{y}$ whose components y_i are either 1 or -1.

For an $n \times m$ interval matrix $[\mathbf{A}]$, whose components are intervals $[A_{ij}] = [\underline{A_{ij}}, \overline{A_{ij}}]$, the corresponding vertex matrix $\mathbf{A_y}$ for each $\mathbf{y} \in Y_n$ has components

$$A_{\mathbf{y}ij} = \underline{A_{ij}} + (\overline{A_{ij}} - \underline{A_{ij}})(1 - y_i)/2 \tag{8}$$

For an n-dimensional interval vector $[\mathbf{b}]$, whose components are intervals $[b_i] = [\underline{b_i}, \overline{b_i}]$, the corresponding vertex vector $\mathbf{b_y}$ for each $\mathbf{y} \in Y_n$ has components

$$b_{\mathbf{y}i} = \underline{b_i} + (\overline{b_i} - \underline{b_i})(1 + y_i)/2 \tag{9}$$

4.3.2. Strong Feasibility

Theorem 1. *[19] A system* $\mathbf{Ax} = \mathbf{b}$ *is strongly feasible if and only if for each* $\mathbf{y} \in Y_n$ *the system*

$$\mathbf{A_y x} = \mathbf{b_y} \tag{10}$$

has a nonnegative solution $\mathbf{x_y}$. *For each* $\mathbf{A} \in [\mathbf{A}],\ \mathbf{b} \in [\mathbf{b}]$, *the system* $\mathbf{Ax} = \mathbf{b}$ *has a solution in the set* $Conv\{\mathbf{x_y} \mid \mathbf{y} \in Y_n\}$.

Gouttefarde et al. [10] propose a technique utilizing linear programming (LP) to determine the strong feasibility of Eq. (6) for $\underline{\boldsymbol{\tau}} \geq 0$. The system of interval linear equations is strongly feasible if and only if the 2^n systems of linear equations $\mathbf{J_y}^T \boldsymbol{\tau} = \mathbf{f_y},\ \mathbf{y} \in Y_n$ are all feasible. The feasibility of a system of linear equations can be tested by means of the first phase of the simplex method applied to the LP problem

$$\begin{aligned} &\min\ \mathbf{0}^T \boldsymbol{\tau} \\ &\text{s.t.} \qquad \mathbf{J_y}^T \boldsymbol{\tau} = \mathbf{f_y} \\ &\qquad\quad\ \ \boldsymbol{\tau} \in [\boldsymbol{\tau}] \end{aligned} \tag{11}$$

where the objective function is trivial since only feasibility of the system of linear equations is desired and the solution set satisfies

$$\text{Conv}\{\boldsymbol{\tau_y} \mid \mathbf{y} \in Y_n\} \ \subseteq\ [\boldsymbol{\tau}] \tag{12}$$

Algorithm 3 Wrench workspace inside box determination

Type = COMPUTE_WRENCHWORKSPACE_INSIDE([**J**], [*MAWC*], [$\boldsymbol{\tau}$])

if $\underline{\boldsymbol{\tau}} < 0$ **then**
  FEASIBLE = &STRONG_SOLVABILITY ▷ Use the STRONG_SOLVABILITY test
else
  FEASIBLE = &STRONG_FEASIBILITY ▷ Use the STRONG_FEASIBILITY test
for all $\mathbf{y} \in Y_n$ **do**
  if FEASIBLE($\mathbf{y}$, [**J**], [*MAWC*], [$\boldsymbol{\tau}$]) == -1 **then**
    return *Type* = 0 ▷ Cannot be classified
return *Type* = 1 ▷ Classify as inside box

Therefore, if each system $\mathbf{J}_\mathbf{y}^T \boldsymbol{\tau} = \mathbf{f}_\mathbf{y},\ \mathbf{y} \in Y_n$ is feasible via Eq. (11), than the robot is guaranteed to be able to generate the task wrenches, [*MAWC*], with the actuator capabilities, [$\boldsymbol{\tau}$], for $\underline{\boldsymbol{\tau}} \geq 0$.

4.3.3. Strong solvability

Theorem 2. *[19] A system* $\mathbf{Ax} = \mathbf{b}$ *is strongly solvable if and only if for each* $\mathbf{y} \in Y_n$ *the system*

$$\begin{gathered} \mathbf{A}_\mathbf{y}\mathbf{x}^1 - \mathbf{A}_{-\mathbf{y}}\mathbf{x}^2 = \mathbf{b}_\mathbf{y} \\ \mathbf{x}^1 \geq 0,\ \mathbf{x}^2 \geq 0 \end{gathered} \tag{13}$$

has a solution $\mathbf{x}_\mathbf{y}^1,\ \mathbf{x}_\mathbf{y}^2$. *For each* $\mathbf{A} \in [\mathbf{A}],\ \mathbf{b} \in [\mathbf{b}]$, *the system* $\mathbf{Ax} = \mathbf{b}$ *has a solution in the set Conv*$\{\mathbf{x}_\mathbf{y}^1 - \mathbf{x}_\mathbf{y}^2 \mid \mathbf{y} \in Y_n\}$.

A similar LP test can be used to determine the strong solvability of Eq. (6) by testing the feasibility of each system of linear equations $\mathbf{J}_\mathbf{y}^T\mathbf{x}^1 - \mathbf{J}_{-\mathbf{y}}^T\mathbf{x}^2 = \mathbf{f}_\mathbf{y},\ \mathbf{y} \in Y_n$ via the LP problem

$$\begin{aligned} &\min\ \mathbf{0}^T(\mathbf{x}^1 - \mathbf{x}^2) \\ &\text{s.t.} \qquad\quad \mathbf{J}_\mathbf{y}^T\mathbf{x}^1 - \mathbf{J}_{-\mathbf{y}}^T\mathbf{x}^2 = \mathbf{f}_\mathbf{y} \\ &\qquad\quad\ \ \mathbf{x}^1 \geq 0,\ \mathbf{x}^2 \geq 0 \\ &\qquad\quad\ \ \mathbf{x}^1 - \mathbf{x}^2 \in [\boldsymbol{\tau}] \end{aligned} \tag{14}$$

where [$\boldsymbol{\tau}$] is not strictly positive and the solution set satisfies

$$\text{Conv}\{\mathbf{x}_\mathbf{y}^1 - \mathbf{x}_\mathbf{y}^2 \mid \mathbf{y} \in Y_n\} \ \subseteq\ [\boldsymbol{\tau}] \tag{15}$$

Therefore, if each system $\mathbf{J}_\mathbf{y}^T\mathbf{x}^1 - \mathbf{J}_{-\mathbf{y}}^T\mathbf{x}^2 = \mathbf{f}_\mathbf{y},\ \mathbf{y} \in Y_n$ is feasible via Eq. (14), than the robot is guaranteed to be able to generate the task wrenches, [*MAWC*], with the actuator capabilities, [$\boldsymbol{\tau}$], for $\underline{\boldsymbol{\tau}} < 0$.

4.4. Wrench Workspace – Inside Box Classification

By means of the strong solvability and strong feasibility theorems, a pose interval [**p**] can be classified as an inside box via Algorithm 3. The functions STRONG_SOLVABILITY and STRONG_FEASIBILITY return 1 if true, and -1 if false. COMPUTE_WRENCHWORKSPACE_INSIDE([**J**], [*MAWC*], [$\boldsymbol{\tau}$]) returns 1 if the box is inside and 0 if the box is not inside. A return value of 0 does not necessary imply an outside box.

4.5. Wrench Workspace – Outside Box Classification

The conditions for testing if a pose interval [**p**] is fully outside of the wrench workspace is proposed by Gouttefarde et al. [10]

$$\exists\, \mathbf{f}_\mathbf{y} \in [MAWC] \mid \forall\, \mathbf{J} \in [\mathbf{J}],\ \forall\, \boldsymbol{\tau} \in [\boldsymbol{\tau}],\ \mathbf{J}^T\boldsymbol{\tau} \neq \mathbf{f}_\mathbf{y} \tag{16}$$

which implies that some wrench $\mathbf{f}_\mathbf{y}$ cannot be generated with admissible actuator capabilities [$\boldsymbol{\tau}$] for all $\mathbf{p} \in [\mathbf{p}]$, and therefore [**p**] must be an outside box.

The condition (16) can be tested by applying interval filtering techniques to the system of interval linear equations $[\mathbf{J}]^T\boldsymbol{\tau} = \mathbf{f_y},\ \boldsymbol{\tau} \in [\boldsymbol{\tau}]$ for each vertex vector $\mathbf{f_y}$ of $[MAWC]$ to determine a new box $[\boldsymbol{\tau}]'$, such that $[\boldsymbol{\tau}]' \subseteq [\boldsymbol{\tau}]$. If the filtering technique returns $[\boldsymbol{\tau}]' = \{\emptyset\}$ for any of the systems of interval linear equations with domain $[\boldsymbol{\tau}]$, the system is inconsistent and Eq. (16) is true. COMPUTE_WRENCHWORKSPACE_OUTSIDE($[\mathbf{J}]$, $[MAWC]$, $[\boldsymbol{\tau}]$) performs the wrench workspace outside box classification test and returns 1 if classified as an outside box and 0 if the box is not outside.

4.6. Classification Algorithm

The function COMPUTE_JACOBIAN($[\mathbf{k}], [\mathbf{h}]$) computes the interval Jacobian matrix for the PM given the interval solution to the IKP, $[\mathbf{k}]$, and the design variables, $[\mathbf{h}]$, and returns Err. Err has a value of 1 if the interval Jacobian matrix cannot be computed (*e.g.*, the inverse Jacobian cannot be inverted) and 0 otherwise. Utilizing the *RW* computation (Algorithm 2), the *WW* inside box classification test (Algorithm 3) and the *WW* outside box classification test, the *WW* can be computed for many PM architectures. The *WW* computation algorithm is described in Algorithm 4.

The *RW* is used to initialize the search space for the *WW* determination. Due to $WW \subseteq RW$, the set of inside boxes from the *RW* determination are added to the unclassified list $\mathcal{L}$ in the *WW* determination. All other boxes are reclassified as being outside boxes. Each box in list $\mathcal{L}$ is tested for being wrench-capable. The interval Jacobian matrix, $[\mathbf{J}]$, is computed for each box, and each box is then tested for being inside or outside, given the desired wrench set, $[MAWC]$, and actuator capabilities, $[\boldsymbol{\tau}]$. The branch-and-bound strategy is applied to refine the list of boundary boxes.

It is important to obtain a tight representation of $[\mathbf{J}]$ which minimizes the impact of the wrapping effect and dependency problem (refer to Section 2.1) in order to avoid large layers of boundary boxes. Several procedures provided by ALIAS-Maple (based on the ALIAS C++ library) attempt to transform an expression into an equivalent expression which leads to better interval evaluation. Preconditioning can also be applied to the system of interval linear equations (Eq.(6)) to transform the system into a new system which contains all solutions of the original system, but gives tighter bounds to the solutions of the original system [4]. Preconditioning can be performed by multiplying the matrix $\mathbf{P} = (\text{mid}([\mathbf{J}]^T))^{-1}$ to the original system such that the new system is

$$\mathbf{P}[\mathbf{J}]^T\boldsymbol{\tau} = \mathbf{P}[MAWC],\ \boldsymbol{\tau} \in [\underline{\boldsymbol{\tau}}, \overline{\boldsymbol{\tau}}] \tag{17}$$

Algorithm 4 is applied to the 3-RRR PM previously described. The *WW* determination is performed for a task with wrench requirements $[MAWC] = ([f_x], [f_y], [m_z])^T = ([10,10]\ \text{N}, [10,10]\ \text{N}, [0,0]\ \text{Nm})^T$ and a resolution $\varepsilon = 0.001$ m and is shown in Figure 3b. For clarity, the edges of the boxes have been removed. The *WW* result is guaranteed in the sense that each of the infinitely many poses contained within each inside box are guaranteed to be able to generate the task wrench requirements. The boundary box layer in the *WW* is thick due to the overestimation present in $[\mathbf{J}]$. The thin boundary box curves present in the *WW* are caused by the inverse Jacobian matrix being non-invertible.

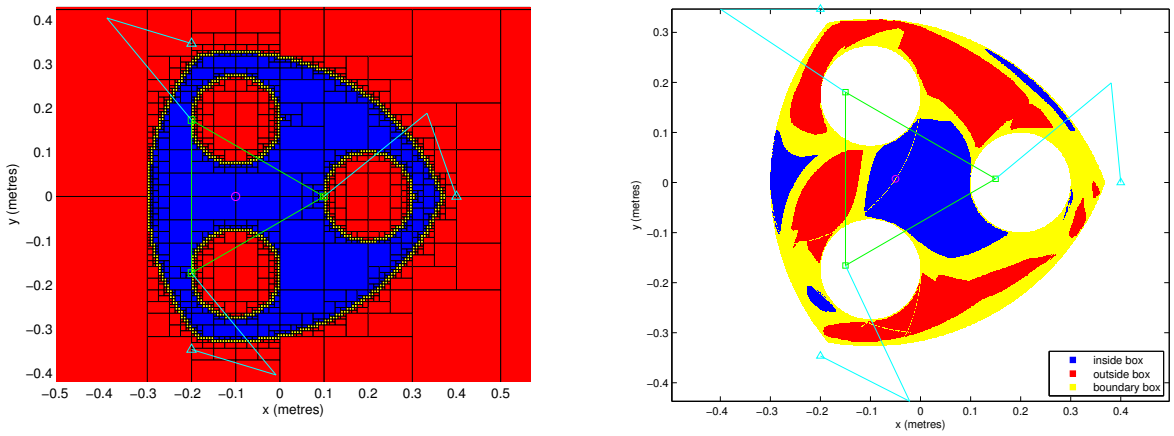
5. CONCLUSIONS

This paper presented an algorithm for the determination of the *WW* for PMs which can be applied to architectures utilizing actuators with strictly nonnegative capabilities, *i.e.*, CDPMs with nonnegative cable tensions, and architectures with general type actuators, *e.g.*, revolute or prismatic, with positive and negative capabilities. The *RW* is obtained using interval analysis techniques applied to a constraint satisfaction problem formed from the direct and inverse kinematics equations. The inside boxes of the *RW* algorithm are used to initialize the search space for the *WW* algorithm. Each box is then classified in the *WW* algorithm based on tests applied to a system of interval linear equations. The inside test utilizes linear programming techniques to determine if the system is appropriately strongly solvable or strongly feasible. The comple-

Algorithm 4 Wrench workspace evaluation with interval analysis

$\mathcal{L}_{inside}, \mathcal{L}_{outside}, \mathcal{L}_{boundary} =$ COMPUTE_WRENCHWORKSPACE($\mathcal{S}$, $[\mathbf{h}]$, ε)

Initialize empty lists $\mathcal{L}$, $\mathcal{L}_{inside}$, $\mathcal{L}_{outside}$, and $\mathcal{L}_{boundary}$
$\mathcal{L}_{inside}, \mathcal{L}_{outside}, \mathcal{L}_{boundary} =$ COMPUTE_REACHABLEWORKSPACE($\mathcal{S}$, $[\mathbf{h}]$, ε)
$\mathcal{L} \leftarrow \mathcal{L}_{inside}$ ▷ Initialize list $\mathcal{L}$ with *RW* inside boxes $\mathcal{L}_{inside}$
$\mathcal{L}_{outside} \leftarrow \mathcal{L}_{boundary}$ ▷ Move list $\mathcal{L}_{boundary}$ to list $\mathcal{L}_{outside}$
Clear lists $\mathcal{L}_{boundary}$, $\mathcal{L}_{inside}$
while $\mathcal{L} \neq \{\emptyset\}$ **do**
    $[\mathbf{p}], [\mathbf{k}] \leftarrow$ EXTRACT($\mathcal{L}$) ▷ Select boxes from top of list $\mathcal{L}$
    if EXISTS($[\mathbf{k}]$)$== -1$ **then** ▷ Compute IKP if $[\mathbf{k}]$ does not exist
        $[\mathbf{k}] =$ COMPUTE_INVERSEKINEMATICS($[\mathbf{p}], [\mathbf{h}]$)
    $(Err, [\mathbf{J}]) =$ COMPUTE_JACOBIAN($[\mathbf{k}], [\mathbf{h}]$)
    if $Err == 1$ **then**
        if Width($[\mathbf{p}]$)$> \varepsilon$ **then**
            $\mathcal{L} \leftarrow$ BISECT($[\mathbf{p}]$)
        else ▷ $[\mathbf{p}]$ is too small to be bisected
            $\mathcal{L}_{boundary} \leftarrow [\mathbf{p}]$
    else
        if COMPUTE_WRENCHWORKSPACE_OUTSIDE($[\mathbf{J}]$, $[MAWC]$, $[\boldsymbol{\tau}]$) **then**
            $\mathcal{L}_{outside} \leftarrow [\mathbf{p}]$ ▷ $[\mathbf{p}] \not\subseteq$ WW
        else if COMPUTE_WRENCHWORKSPACE_INSIDE($[\mathbf{J}]$, $[MAWC]$, $[\boldsymbol{\tau}]$) **then**
            $\mathcal{L}_{inside} \leftarrow [\mathbf{p}]$ ▷ $[\mathbf{p}] \subseteq$ WW
        else
            if Width($[\mathbf{p}]$)$> \varepsilon$ **then**
                $\mathcal{L} \leftarrow$ BISECT($[\mathbf{p}]$)
            else ▷ $[\mathbf{p}]$ is too small to be bisected
                $\mathcal{L}_{boundary} \leftarrow [\mathbf{p}]$



(a) Reachable workspace for the 3-RRR PM.

(b) Wrench workspace for the 3-RRR PM.

Fig. 3. Reachable and wrench workspaces for the 3-RRR PM.

mentary outside test applies an interval filtering technique to the system to determine if the bounded solution is inconsistent. The *WW* algorithm is applicable to many PM architectures, provided that the inverse kinematics problem can be solved in terms of a constraint satisfaction problem, and that the interval Jacobian matrix can be tightly represented.

ACKNOWLEDGMENT

The authors would like to thank the Natural Sciences and Engineering Research Council of Canada (NSERC) for their funding of this research.

REFERENCES

1. S. Bouchard and C. M. Gosselin. Workspace optimization of a very large cable-driven parallel mechanism for a radiotelescope application. In *Proceedings of the 2007 ASME Design Engineering Technical Conferences*, number DETC2007-34286, page 7, Las Vegas, Nevada, USA, September 4-7 2007.
2. S. Bouchard, C. M. Gosselin, and B. Moore. On the ability of a cable-driven robot to generate a prescribed set of wrenches. In *Proceedings of the 2008 ASME Design Engineering Technical Conferences*, number DETC2008-49518, page 12, New York, NY, USA, August 3-6 2007.
3. J. A. Carretero and C. M. Gosselin. Wrench capabilities of cable-driven parallel mechanisms using wrench polytopes. In *Proceeding of the 2010 IFToMM Symposium on Mechanism Design for Robotics*, Universidad Panamericana, Mexico City, Mexico, September 28–30 2010.
4. C. K. Chiu and J. H. M. Lee. Interval linear constraint solving using the preconditioned interval gauss-seidel method. In *Proceedings of the twelfth International Conference nn Logic Programming*, pages 17–32. The MIT Press, 1994.
5. F. Firmani, A. Zibil, S. B. Nokleby, and R. P. Podhorodeski. Force-moment capabilities of revolute-jointed planar parallel manipulators with additional actuated branches. *Transactions of the Canadian Society for Mechanical Engineering*, 31(4):469–481, 2007.
6. F. Firmani, A. Zibil, R. P. Podhorodeski, and S. B. Nokleby. Wrench capabilities of planar parallel manipulators. part I: Wrench polytopes and performance indices. *Robotica*, 26(6):791–802, November 2008.
7. F. Firmani, A. Zibil, R. P. Podhorodeski, and S. B. Nokleby. Wrench capabilities of planar parallel manipulators. part II: Redundancy and wrench workspace analysis. *Robotica*, 26(6):803–815, November 2008.
8. V. Garg, J. A. Carretero, and S. B. Nokleby. A new method to calculate the force and moment workspaces of actuation redundant spatial parallel manipulators. *Journal of Mechanisms and Robotics*, 1(3):1–8, August 2009.
9. V. Garg, S. B. Nokleby, and J. A. Carretero. Force-moment capability analysis of redundantly-actuated spatial parallel manipulators. *Mechanism and Machine Theory*, 44(5):1070–1081, May 2009.
10. M. Gouttefarde, D. Daney, and J.P. Merlet. Interval-analysis-based determination of the wrench-feasible workspace of parallel cable-driven robots. *IEEE Transactions on Robotics*, 27(1), 2011.
11. L. Jaulin, M. Kieffer, O. Didrit, and E. Walter. *Applied Interval Analysis*. Springer, 2001.
12. M.H.F. Kaloorazi, S. Esfahani, and M.T. Masouleh, editors. *Dimensional synthesis of planar parallel cable-driven parallel robots via interval analysis*, May 30–31 2013. Proceedings of the 2013 CCToMM Mechanisms, Machines, and Mechatronics Symposium.
13. Olivier Lhomme. Consistency techniques for numeric csps. pages 232–238, 1993.
14. R.E. Moore. *Interval analysis*. Prentice-Hall series in automatic computation. Prentice-Hall, 1966.
15. R.E. Moore, R.B. Kearfott, and M.J. Cloud. *Introduction to Interval Analysis*. Society for Industrial and Applied Mathematics, 3600 Market Street, 6th Floor, Philadelphia, PA, 19104-2688 USA, 2009.
16. D. Oetomo, D. Daney, B. Shirinsadeh, and J.P. Merlet. An interval-based method for workspace analysis of planar flexure-jointed mechanism. In *Journal of Mechanical Design*, volume 131. 2008.
17. J.K Pickard and J.A. Carretero. A comparison of wrench-feasible workspaces using interval analysis and convex linear mapping techniques. Toronto, ON, Canada, June 1–4 2014. Proceedings of the Canadian Society for Mechanical Engineering International Congress 2014.
18. A. T. Riechel and I. Ebert-Uphoff. Force-feasible workspace analysis for underconstrained, point-mass cable robots. In *2004 IEEE International Conference on Robotics and Automation, 2004. Proceedings. ICRA '04.*, volume 5, pages 4956–4962, April 2004.
19. J. Rohn. Solvability of systems of interval linear equations and inequalities. In *Linear Optimization Problems with Inexact Data*, pages 35–77. Springer US, 2006.
20. F. Rossi, P. van Beek, and T. Walsh. *Handbook of Constraint Programming*. Elsevier Science, 1st edition, 2006.
21. A. Zibil, F. Firmani, S. B. Nokleby, and R. P. Podhorodeski. An explicit method for determining the force-moment capabilities of redundantly actuated planar parallel manipulators. *Journal of Mechanical Design*, 129(10):1046–1055, October 2007.