An Interval Analysis Based Study for the Design and the Comparison of Three-Degrees-of-Freedom Parallel Kinematic Machines

D. Chablat
Ph. Wenger
F. Majou

Institut de Recherche en Communications
et Cybernétique de Nantes*
1, rue de la Noë, 44321 Nantes, France
Damien.Chablat@irccyn.ec-nantes.fr
Philippe.Wenger@irccyn.ec-nantes.fr
Félix.Majou@irccyn.ec-nantes.fr

J.-P. Merlet

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

Abstract

This paper addresses an interval analysis based study that is applied to the design and the comparison of three-degrees-of-freedom (3-DoF) parallel kinematic machines. Two design criteria are used: (i) a regular workspace shape and (ii) a kinetostatic performance index that needs to be as homogeneous as possible throughout the workspace. The interval analysis based method takes these two criteria into account; on the basis of prescribed kinetostatic performances, the workspace is analyzed to find the largest regular dextrous workspace enclosed in the Cartesian workspace. An algorithm describing this method is introduced. Two 3-DoF translational parallel mechanisms designed for machining applications are compared using this method. The first machine features three fixed linear joints which are mounted orthogonally and the second features three linear joints which are mounted in parallel. In both cases, the mobile platform moves in the Cartesian x–y–z space with fixed orientation.

KEY WORDS—parallel kinematic machine, design, interval analysis, comparison, workspace, transmission factors

1. Introduction

Parallel kinematic machines (PKMs) are known for their high dynamic performances and low positioning errors. The kinematic design of PKMs has drawn the interest of several researchers. The workspace is usually considered as a relevant design criterion (Merlet 1996; Clavel 1988; Gosselin and Angeles 1991). Parallel singularities (Wenger and Chablat 1997) occur in the workspace where the moving platform cannot resist any effort, and thus are very undesirable and generally eliminated by design. The Jacobian matrix, which relates the joint rates to the output velocities is generally not constant and not isotropic. Consequently, the performances (e.g., maximum speeds, forces, accuracy, and stiffness) vary considerably for different points in the Cartesian workspace and for different directions at one given point. This is a serious drawback for machining applications (Kim et al. 1997; Treib and Zirn 1998; Wenger, Gosselin, and Chablat 2001). Few parallel mechanisms are isotropic throughout the workspace (Carricato and Parenti-Castelli 2002, Kong and Gosselin 2002). However, their low structural stiffness makes them inadequate for machining applications because their legs are subject to bending.

To be of interest for machining applications, a PKM should preserve good workspace properties, that is, regular workspace shape and acceptable kinetostatic performances throughout. For example, in milling applications, the machining conditions must remain constant along the whole tool path (Rehsteiner et al. 1999; Tlusty, Ziegert, and Ridgeway 1999). In many research papers, this criterion is not taken into account in the algorithmic methods used to compute the workspace volume (Luh et al. 1996; Merlet 1999). Other papers present methods that compute the well-conditioned workspace using discretization (Gosselin and Angeles 1988; Stougthon and Arai 1993). Thus, the results they provide cannot be proved formally. Conversely, interval analysis methods applied to well-conditioned workspace computation provide

*IRCCyN: UMR n° 6597 CNRS, Ecole Centrale de Nantes, Université de Nantes, Ecole des Mines de Nantes.

The International Journal of Robotics Research
Vol. 23, No. 6, June 2004, pp. 615-624,
DOI: 10.1177/0278364904044079

guaranteed results (Merlet 2000; Chablat, Wenger, and Merlet 2002).

The comparison of PKM architectures is a difficult but relevant challenge (Wenger, Gosselin, and Chablat 2001; Joshi and Tsai 2003). Providing tools to allow designers or end-users to rigorously compare PKM is indeed necessary because the variety of existing PKMs makes it hard to choose which one is best suited for a specific task.

In this paper, an interval analysis based method is addressed for the design and comparison of a three-degrees-of-freedom (3-DoF) PKM. This method takes into account two criteria: (i) a regular workspace shape and (ii) a kinetostatic performance index that needs to be as homogeneous as possible throughout the workspace. Two basic tools and an algorithm that considers these two criteria are introduced; on the basis of prescribed kinetostatic performances, the workspace is analyzed to find the largest regular dextrous workspace (square, cube, cylinder, etc.) enclosed in the Cartesian workspace.

Two translational parallel mechanisms derived from the Delta robot (Clavel 1988) are compared using this method. The first machine, called Orthoglide (Chablat and Wenger 2003), features three fixed linear joints which are mounted orthogonally and the second, called UraneSX (Renault Automation; Company and Pierrot 2002), features three linear joints which are mounted in parallel. In both cases, the mobile platform moves in the Cartesian x–y–z space with fixed orientation.

In the next section we present the interval analysis based method for 3-DoF PKM design. In Section 3 we present the Orthoglide and UraneSX mechanisms, their kinematic equations and singularity analysis. In Section 4 we report on the comparison between the two mechanisms through the determination of the largest dextrous cube for the Orthoglide and the largest dextrous square for the UraneSX enclosed in the workspace.

2. Description of the Interval Analysis Based Method for 3-DoF Translational PKM Design

2.1. Preliminaries

2.1.1. Dextrous Cartesian Workspace

For a three-axis serial machine tool, a parallelepiped-shaped Cartesian workspace allows the end-user to visualize easily where to place cutting paths. This consideration should also hold for PKMs. However, the workspace shape is often geometrically complex and thus hard to visualize. Therefore, a regular-shaped workspace is needed for PKMs. Thus, we need to define a regular dextrous workspace which is a regular-shaped workspace included in the machine Cartesian workspace. Throughout the dextrous workspace, a kinetostatic performance index (that is chosen beforehand) remains as homogeneous as possible. This index can be the local or global conditioning (Gosselin and Angeles 1991) of the Jacobian matrix **J** (that maps the actuated joint rates of the manipulator into the velocity of the mobile platform), the force or velocity transmission factors. These last two indices make sense for a 3-DoF translational PKM with identical actuated joints.

The method presented in this paper aims at designing such a 3-DoF PKM. The velocity transmission factors are the ratio between the actuated joints velocities and the velocity of the mobile platform. They are the square roots ψ_1, ψ_2, and ψ_3 of the real eigenvalues σ_1, σ_2, and σ_3 of $(\mathbf{J}\mathbf{J}^{\mathrm{T}})^{-1}$. In order to keep homogeneous kinetostatic properties, these factors are bounded inside the dextrous workspace. The "regular dextrous Cartesian workspace" can be defined as a set of points P in the workspace such that ψ_1, ψ_2, and ψ_3 are bounded, i.e.,

$$\mathcal{W}_{Dextrous} = \{P \in \mathcal{W} \mid \psi_{min} \leq \psi_i(P) \leq \psi_{max}, \quad i = 1, 2, 3\}. \tag{1}$$

Points P in $\mathcal{W}_{Dextrous}$ are called "dextrous points".

The values of ψ_{min} and ψ_{max} (σ_{min} and σ_{max}, respectively) depend on given performance requirements. The method described further aims at computing the largest dextrous Cartesian workspace included in the Cartesian workspace, so that its ratio to the Cartesian workspace is the best one. To be of real interest for milling applications, a PKM must indeed include a large regular dextrous workspace in its Cartesian workspace.

2.1.2. Introduction to the ALIAS Library

An algorithm for the definition of the largest dextrous workspace included in the Cartesian workspace is described in the following sections. This algorithm uses the ALIAS library (Merlet 2000), which is a C++ library of algorithms based on interval analysis. These algorithms deal with systems of equations and inequalities whose expressions are an arbitrary combination of the most classical mathematical functions (algebraic terms, sine, cosine, log, etc.) and whose coefficients are real numbers or, in some cases, intervals. An interface exists with Maple that allows the automatic generation of C++ codes being given the Maple description of the system and then to compile and run the generated code in order to obtain the result within the Maple session. Without being exhaustive, the ALIAS library provides algorithms that enable one (i) to find an approximation of the real roots of n-dimensional systems, (ii) to find an approximation of the variety defined by n-dimensional systems, (iii) to find an approximation of the global minimum or maximum of a function (eventually under equations and/or inequalities constraints) up to an accuracy provided by the user, and (iv) to analyze a system of algebraic equations to determine bounds for its real roots.

2.1.3. Geometric Constraints

The dextrous workspace $\mathcal{W}_{Dextrous}$ is defined in eq. 1 as a function of the eigenvalues of $(\mathbf{J}\mathbf{J}^{\mathrm{T}})^{-1}$. These eigenvalues are determined by solving the third degree characteristic polynomial $\mathcal{P}$ of $(\mathbf{J}\mathbf{J}^{\mathrm{T}})^{-1}$. To decrease the computing time and to avoid numerical problems on singularities, it is recommended to add geometrical constraints. These constraints naturally depend on the mechanism architecture (see Section 4.2).

2.2. A First Basic Tool: Box Verification

Our purpose is to determine the largest regular dextrous workspace that is enclosed in the Cartesian workspace. For a given point, we note "valid point" if it is a dextrous point and "invalid point" otherwise. For that purpose we need to design first a procedure, called $\mathcal{M}(B)$, that takes as input a Cartesian box B and returns:

1 : if every point in B is valid;

-1 : if no point in B is valid;

0 : if neither of the other two conditions could be verified.

The first step of this procedure consists in considering an arbitrary point of the box (e.g., its center) and to compute the eigenvalues at this point; either all of them lie in the range $[\sigma_{min}, \sigma_{max}]$ in which case the center is called "valid" or at least one of them lies outside this range and the center point is denoted "invalid".

2.2.1. Valid Center Point

In that case, if we are able to check that there is no point in B such that one of the eigenvalues at this point is equal to σ_{min} or σ_{max}, then we can guarantee that every point in B is valid. Indeed, we assume that at a given point B the lowest eigenvalue is lower than σ_{min}; this implies that somewhere along the line joining this point to the center of the box the lowest eigenvalue is exactly σ_{min}.

To perform this check we set the unknown in the characteristic polynomial $\mathcal{P}$ of $(\mathbf{J}\mathbf{J}^{\mathrm{T}})^{-1}$ to σ_{min} (and then to σ_{max}) and we obtain a polynomial in x, y, z only. We now have to determine if there exist some values for these three Cartesian coordinates that cancel the polynomial, it being understood that these values have to define a point belonging to B. This is done by using an interval analysis algorithm from the ALIAS library (Merlet 2000). The principle of this algorithm is to calculate first the polynomial value for the center point C_B of B. Without lack of generality we may assume that this value is positive. If we are able to determine a point S_B in B such that the polynomial value at this point is negative, then we can guarantee that there exists a point on the line joining C_B to S_B such that the polynomial is exactly 0. The purpose of the algorithm is now to determine if such a point exists. Now let B_i be a box included in B. Using interval analysis we are able to calculate a range $[m_{B_i}, M_{B_i}]$ such that for any point X in B_i we have $m_{B_i} \leq \mathcal{P}(X) \leq M_{B_i}$. Note that this interval evaluation is numerically safe as the bounds of the range are calculated by taking into account round-off errors. On the other hand, these bounds may not be "sharp", i.e., there may be no X in B_i such that $\mathcal{P}(X) = m_{B_i}$ or M_{B_i}. Note, however, that the width of the overestimation decreases with the width of B_i. Furthermore, we may obtain a sharp evaluation by using, for instance, the derivatives of $\mathcal{P}$. Indeed we may calculate the interval evaluation $[r_{x,y,z}, R_{x,y,z}]$ of $\partial\mathcal{P}/\partial x, y, z$ and, if all three interval evaluations have constant signs (i.e., $r_{x,y,z} > 0$ or $R_{x,y,z} < 0$), then sharp m_{B_i}, M_{B_i} are obtained by setting the variables to fixed values. For instance, if $r_x > 0$, then $m_{B_i}(M_{B_i})$ is obtained by setting x to its lower (upper) bound. Note that other methods may also be used to determine sharp bounds (see Moore 1979; Neumaier 1990; Ratscheck and Rokne 1995).

Hence we have the following properties:

1. if $m_{B_i} > 0$, then for any point in B_i the polynomial $\mathcal{P}$ is positive;
2. if $M_{B_i} < 0$, then for any point in B_i the polynomial $\mathcal{P}$ is negative;
3. if $m_{B_i} < 0$, $M_{B_i} > 0$ and the bounds are sharp, then the polynomial $\mathcal{P}$ cancels in B_i;
4. if $m_{B_i} < 0$, $M_{B_i} > 0$ and the bounds are not sharp, then we cannot guarantee the sign of the polynomial $\mathcal{P}$ within B_i.

At that point a simple branch-and-bound algorithm is used; the initial box B is bisected until either all the subboxes resulting from the bisection satisfy property 1 (in which case we can guarantee that the polynomial $\mathcal{P}$ never cancels for B and consequently that all the eigenvalues of $\mathcal{P}$ lie in the range $[\sigma_{min}, \sigma_{max}]$, which implies $\mathcal{M}(B) = 1$) or a subbox resulting from the bisection satisfies property 2 or 3, which means that at some point in B at least one of the eigenvalues of $\mathcal{P}$ lies outside the range $[\sigma_{min}, \sigma_{max}]$, which corresponds to $\mathcal{M}(B) = 0$.

The algorithm may indeed return 0 for a box that includes only valid points, but the width of this box will be lower than $\alpha/2$ (where α is an accuracy threshold fixed in advance for the computation) and hence the final result will be within the tolerance margin of the calculation. The only case in which the calculation will be not guaranteed will occur only when α is lower than the machine accuracy. However, we may determine that we are in such a configuration as the width of the box from the machine viewpoint will be 0; if a box of width 0 is processed and the algorithm returns 0, then a warning message will be issued indicating that the calculation is no longer guaranteed. Note, however, that we may still use the algorithm by using a multi-precision package such as MPFR that will allow us to obtain a guaranteed result. Furthermore, it is doubtful that computing the result with an accuracy better than the machine precision makes sense.

2.2.2. Invalid Center Point

Without lack of generality, we may assume that at the center of the box the largest eigenvalue is greater than σ_{max}. If there is no point in B such that one of the eigenvalues is equal to σ_{max}, then we can guarantee that for any point in B the largest eigenvalue is always greater than σ_{max} and consequently $\mathcal{M}(B) = -1$. This check is performed by using the same method as in the previous case.

2.3. A Second Basic Tool: Box Workspace Verification

During the calculation of the dextrous workspace, we consider a Cartesian box B and we have to examine if this box may contain a point that is the center of a Cartesian box B_W with edge length w, which is fully enclosed in the robot workspace. We assume here that this workspace is defined by a set of m inequalities F_j such that a point X belongs to the workspace if $F_j(X) \leq 0$ for all j in $[1, m]$. Let B_i be a subbox included in B, defined by the three ranges $[\underline{x_i}, \overline{x_i}]$, $[\underline{y_i}, \overline{y_i}]$, $[\underline{z_i}, \overline{z_i}]$. All the boxes with edge length w that have as center a point in B_i are included in the "hull box" H_{B_i} defined by the three ranges $[\underline{x_i}-w/2, \overline{x_i}+w/2]$, $[\underline{y_i}-w/2, \overline{y_i}+w/2]$, $[\underline{z_i}-w/2, \overline{z_i}+w/2]$. As in the previous section we may use interval analysis to compute an interval evaluation $[m^j_{B_i}, M^j_{B_i}]$ of all $F_j(H_{B_i})$ with the following properties:

1. if $M^j_{B_i} < 0$ for all j in $[1, m]$, then any point of B_i may be the center of a box with edge length w that is included in the workspace;
2. if $m^j_{B_i} > 0$, then no point of B_i may be the center of a box with edge length w that is included in the workspace;
3. if $m_{B_i} < 0$ and $M_{B_i} > 0$, then we cannot determine if some point within B_i may be the center of B_W.

Note also that, if the widths of all the ranges defining B_i are lower than w, any box B_W contains the four corners of the box B_i.

Using a similar branch-and-bound algorithm as in the previous section, we may now determine whether all, none or some points of B may be the center of a box B_W. The initial box B is bisected until all the subboxes resulting from the bisection satisfy either property 1 (then any point of B may be the center of a box B_W) or property 2 (no point of B may be the center of a box B_W). If a subbox satisfies property 3 and the widths of its ranges are lower than w, we check if the corners of B belong to the workspace; if all the corners either belong or do not belong to the workspace we continue the bisection. If we have a mixed situation with some corners belonging to the workspace whereas others do not, we may state that B contains both points that may be the center of a box B_W and points that cannot. A similar situation is obtained if we have found at least a subbox that satisfies property 1 and a subbox that satisfies property 2.

At that point we may define a procedure $\mathcal{G}(B, w)$ that takes as input a box B and an edge length w and returns:

-1 : there are no points in B that may be the center of a box B_W;

1 : all the points in B may be the center of a box B_W;

0 : B contains both points that may be the center of a box B_W and points that cannot.

2.4. Algorithm for the Determination of a Cubic Dextrous Cartesian Workspace

An algorithm is now described for the determination of a cube that is enclosed in the Cartesian workspace and aligned with the coordinate axis, whose edge length is $2w$ and such that there is no other cube enclosed in the workspace with an edge length of $2(w+\alpha)$. This algorithm can be applied to any 3-DoF manipulator. Other shapes for regular dextrous workspace are considered in Section 2.5.

The first step is to determine the largest cube enclosed in the workspace with a center located at $(0, 0, 0)$. This is done by using the $\mathcal{M}$ procedure on the Cartesian box B_{init} $[-k\alpha, k\alpha]$, $[-k\alpha, k\alpha]$, $[-k\alpha, k\alpha]$ where k is an integer initialized to 1. Each time the $\mathcal{M}$ procedure returns 1 for B_{init} (which means that the cube with edge length $2k\alpha$ is enclosed in the dextrous workspace) we double the value of k. If this procedure returns -1 for a value of k larger than 1 this implies that the cube with edge length $k\alpha/2$ is in the dextrous workspace whereas the cube with edge length $k\alpha$ is not. Hence, if $k > 2$ (otherwise no improvement is possible) we restart the process with $k = (k/2+k)/2$. After a failure at k_{fail} the principle is to always choose a value of k which is the mid-point between the last value k_s of k for which $\mathcal{M} = 1$ and k_{fail} until $k_{fail} = k_s+1$. For example, if $\mathcal{M}$ returns 1 for k=1, 2, 4 and returns -1 for $k = 8$ we set k to 6. Otherwise we have determined that the cube with edge length $2k\alpha$ is enclosed in the dextrous workspace, whereas the cube with edge length $2(k+1)\alpha$ is not. The value $2k\alpha$ is hence an initial value for w. Note that the above procedure may be used whatever the coordinates of the center; it is implemented as a general purpose procedure $\mathcal{C}(x_M, y_M, z_M)$ that takes as input the coordinates of a point M and returns the edge length of the largest cube centered at M, that is enclosed in the dextrous workspace.

In the algorithm for determining the largest cube enclosed in the dextrous workspace, we manage a list of Cartesian boxes $\mathcal{L}$ that are processed by the algorithm in sequence. During the processing, boxes may be added to a list. At one step of the algorithm, we have n boxes in the list. When box i is processed, boxes 0 to i–1 have already been processed and may be discarded whereas boxes i to n have to be processed. The algorithm stops when all the boxes in $\mathcal{L}$ have been processed. The box numbered i in the list is denoted B_i and the maximum number of boxes in $\mathcal{L}$ is N.

At the beginning of the algorithm, $\mathcal{L}$ has only one box B_0 that contains the workspace (for example, for the Orthoglide B_0= $\{[-L, L], [-L, L], [-L, L]\}$). The algorithm can be described by the following six steps.

1. Calculate $w = C(0, 0, 0)$.
2. If $i > n$ EXIT.
3. If $\mathcal{G}(B_i, w + \alpha) = -1$, then set i to $i + 1$ and go to 2.
4. If $\mathcal{G}(B_i, w + \alpha) = 1$, then calculate $w' = C(x_{B_i}, y_{B_i}, z_{B_i})$ where $x_{B_i}, y_{B_i}, z_{B_i}$ are the coordinates of the center of B_i. If $w' > w$, then update w. Go to step 6.
5. If $\mathcal{G}(B_i, w + \alpha)$= 0, then go to step 6.
6. Bisect the variable in the box B_i that has the largest range. For example, if the box B_i is defined as $[\underline{x_i}, \overline{x_i}]$, $[\underline{y_i}, \overline{y_i}]$, $[\underline{z_i}, \overline{z_i}]$ and the variable x has the largest range, the bisection process creates two new boxes $B_i^1 = \{[\underline{x_i}, (\underline{x_i} + \overline{x_i})/2], [\underline{y_i}, \overline{y_i}], [\underline{z_i}, \overline{z_i}]\}$ and $B_i^2 = \{[(\underline{x_i} + \overline{x_i})/2, \overline{x_i}], [\underline{y_i}, \overline{y_i}], [\underline{z_i}, \overline{z_i}]\}$. If $n < N/2$, both boxes are stored at the end of the list (and we set i=i+1), otherwise box B_i^1 is stored in $\mathcal{L}$ in place of B_i whereas box B_i^2 is stored at location $i + 1$ after a shift of the boxes $B_{i+1}, \ldots, B_n$. Set n to $n + 1$ and go to step 2.

Step 1 allows us to establish an initial value for the maximal edge length. Step 3 eliminates boxes that cannot contain the center of the maximal cube due to the workspace limits. Boxes satisfying step 4 are candidates to include the center of the largest cube; hence, we calculate the largest cube centered at the box that may allow us to update the current value of the largest edge. Step 6 is the bisection process that allows us to decrease the size of the box with the effect of a sharper calculation for the procedure $\mathcal{G}$. Note also that two storage modes are used for adding the boxes resulting from the bisection process. The second mode allows for a minimal memory storage but has the drawback of focusing on a given part of the workspace, whereas the center of the largest cube may be located in another part. The first mode makes it possible to explore various parts of the workspace which may result in large improvement of the value of w but with the drawback of possibly creating a large number of boxes. The proposed storage mode allows us to mix the advantages of both storage modes.

This procedure ensures that we can determine a cube with edge length w that is enclosed in the workspace and in the dextrous workspace, whereas there is no such cube with edge length $w + \alpha$.

Note that an incremental approach is possible. After having computed $w = w_1$ with a given accuracy α it is always possible to replace the initial value of w as calculated in step 1 of the algorithm by the value w_1 when computing the cube with a lower value for α. Computation times of the largest cube for various accuracies are given for a specific 3-DoF PKM in Section 4.3.

2.5. Other Regular Dextrous Workspace Shapes

Clearly, the largest cube may not be appropriate if the studied PKM has a rectangular or a spherical-shaped workspace. The algorithm can thus be modified. Here are, for example, the necessary changes that must be taken into account to consider the largest sphere. The idea is to use spherical coordinates and hence x, y, z are substituted by $x_c + r \sin\psi \sin\theta$, $y_c + \cos\psi \sin\theta$, $z_c + r\cos\theta$, with r in $[0, R]$, ψ, θ in $[0, 2\pi]$. x_c, y_c, z_c are the coordinates of the center of the sphere and R is its radius. Interval analysis allows us to deal with expressions involving sine and cosine and hence procedures $\mathcal{M}$, $\mathcal{G}$ can still be used with these new parameters. Similarly procedure $\mathcal{C}(x_M, y_M, z_M)$ can be used to determine the largest radius of the sphere centered at (x_M, y_M, z_M) for which the eigenvalues are valid. Hence, with this modification, the algorithm can calculate the largest sphere enclosed in the dextrous workspace.

Spheres and cubes are defined by their center and one additional parameter, but other shapes may involve more parameters; for example a cylinder needs a center but also a height and a radius. We can still perform a change in the variables so that procedures $\mathcal{M}$, $\mathcal{G}$ can still be used. The key point is that procedure $\mathcal{C}(x_M, y_M, z_M)$ has to be modified as we have now two optimization parameters. In that case, volume optimization alone has less meaning; for example, the optimization result for a cylinder may be a cylinder with a relatively small radius and a large height, which may be of no interest. A cylinder of identical radius and height with a lower volume may be the most interesting result. A possible way to manage this problem is to assign a range $[a, b]$ for the ratio R/h where R is the cylinder radius and h its height. In that case the procedure $\mathcal{C}$ has to solve an optimization problem which is to maximize the volume of the cylinder under the constraints that the eigenvalues are valid and the ratio R/h satisfies $a \leq R/h \leq b$. ALIAS is still able to manage such an optimization procedure.

2.6. Approximate Calculation of the Dextrous Workspace

Small modifications in the previous algorithm allow us to determine an approximation of the dextrous Cartesian workspace $\mathcal{W}_{Dextrous}$ as a set $\mathcal{S}$ of three-dimensional (3D) Cartesian boxes such that for any box B in $\mathcal{S}$ and for any point in B the constraints on the eigenvalues are satisfied. The width of all the boxes in the set $\mathcal{S}$ is greater than a given threshold ϵ; hence, we obtain only an approximation of the dextrous Cartesian workspace. However, the algorithm provides the volume V_a of the approximation and a volume error V_e such that the volume V_d of the dextrous Cartesian workspace satisfies $V_d \leq V_a + V_e$. Decreasing the value of ϵ makes it possible

to increase V_a and to decrease V_e. In this paper, this method is used to analyze 3D boxes but it can be applied for any mechanism with n DoFs, the result being a set of n-dimensional boxes.

Initially V_a and V_e are set to 0.

1. If $i > n$ EXIT.
2. If $\mathcal{M}(B_i) = -1$, then set i to $i + 1$ and go to 1.
3. If $\mathcal{M}(B_i) = 1$, then store B_i in $\mathcal{S}$ and add its volume to V_a. Set i to $i + 1$ and go to 1.
4. If $\mathcal{M}(B_i) = 0$, then
   (a) if the largest width of B_i is lower than ϵ, then[(b)] add its volume to V_e, set i to $i + 1$ and go to 1;
   (a) otherwise go to step 5.
5. Process bisection for box B_i. Set n to $n + 1$ and go to 1.

Note that this procedure may be incremental if the boxes neglected at step 4(a) are stored in a file $\mathcal{F}$. Indeed, a first run with a given ϵ allows us to obtain initial values for V_a and V_e. If the quality of the approximation is not satisfactory, we may choose a smaller value of ϵ (e.g. $\epsilon/2$). But instead of starting with the initial B_0, we may use the boxes stored in $\mathcal{F}$, thereby avoiding a repeat computation that has already been done during the initial run.

3. Description of Orthoglide and UraneSX

The previous interval analysis based design method is now applied to the comparison of two 3-DoF translational PKMs. It is particularly interesting to compare these two mechanisms because they belong to the same architecture family.

3.1. Orthoglide and UraneSX Architectures

Most existing PKMs can be classified into two main families. The PKMs of the first family have fixed foot points and variable length struts and are generally called "hexapods" or "tripods". The PKMs of the second family have variable foot points and fixed length struts. They are interesting because the actuators are fixed and thus the moving masses are lower than in the hexapods and tripods.

The Orthoglide and the UraneSX mechanisms studied in this paper are 3-DoF translational PKMs and belong to the second family. Figures 1 and 2 show the general kinematic architecture of the Orthoglide and of the UraneSX, respectively. Both mechanisms have three parallel $PRPaR$ identical chains (where P, R, and Pa denote prismatic, revolute, and parallelogram joints, respectively). The actuated joints are the three linear joints. These joints can be actuated by means of linear motors or by conventional rotary motors with ball screws.

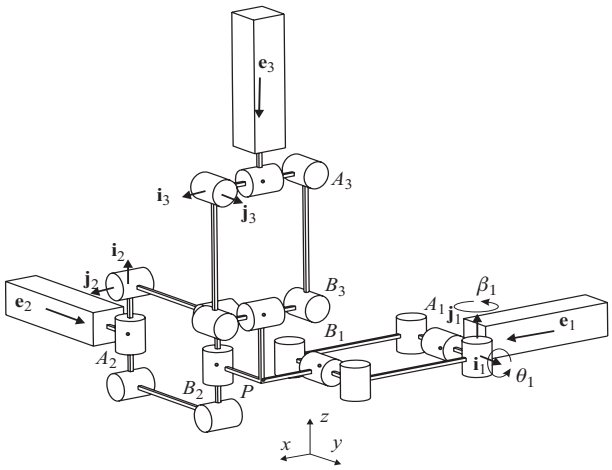

Fig. 1. Orthoglide kinematic architecture.

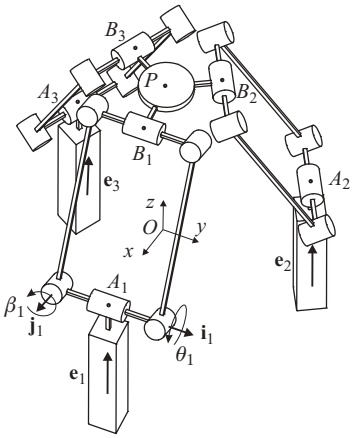

Fig. 2. UraneSX kinematic architecture.

The output body is connected to the linear joints through a set of three parallelograms of equal lengths $L = A_iB_i$, so that it can move only in translation. Vectors $\mathbf{e}_i$ coincide with the direction of the ith linear joint. The base points A_i are located at the middle of the first two revolute joints of the ith parallelogram, and B_i is at the middle of the last two revolute joints of the ith parallelogram.

For the Orthoglide mechanism, the first linear joint axis is parallel to the x-axis, the second is parallel to the y-axis, and the third is parallel to the z-axis. When each vector $\mathbf{e}_i$ is aligned with $\mathbf{A}_i\mathbf{B}_i$, the Orthoglide is in an isotropic configuration and the tool center point P is located at the intersection of the three linear joint axes.

The linear joint axes of the UraneSX mechanism are parallel to the z-axis. In Figure 2, points A_1, A_2, and A_3 are the vertices of an equilateral triangle whose geometric center is O and such that $OA_i = R$. Thus, points B_1, B_2, and B_3 are the vertices of an equilateral triangle whose geometric center is P, and such that $OB_i = r$.

3.2. Kinematic Equations and Singularity Analysis

We recall briefly here the kinematic equations and the singularities of the Orthoglide and of the UraneSX; see Company and Pierrot (2002) and Chablat and Wenger (2003) for more details.

Let θ_i and β_i denote the joint angles of the parallelogram about axes $\mathbf{i}_i$ and $\mathbf{j}_i$, respectively (Figures 1 and 2). Let ρ_1, ρ_2, and ρ_3 denote the linear joint variables, and let L denote the length of the three legs, A_iB_i.

For the Orthoglide, the position vector $\mathbf{p}$ of the tool center point P is defined in a reference frame (O, x, y, z) centered at the intersection of the three linear joint axes (note that the reference frame has been translated in Figure 1 for more legibility).

For the UraneSX, the position vector $\mathbf{p}$ of the tool center point P is defined in a reference frame (O, x, y, z) centered at the geometric center of the points A_1, A_2, and A_3 (same remark as above).

Let $\dot{\boldsymbol{\rho}}$ be referred to as the vector of actuated joint rates and $\dot{\mathbf{p}}$ as the velocity vector of point P:

$$\dot{\boldsymbol{\rho}} = [\dot{\rho}_1 \ \dot{\rho}_2 \ \dot{\rho}_3]^{\mathrm{T}}, \qquad \dot{\mathbf{p}} = [\dot{x} \ \dot{y} \ \dot{z}]^{\mathrm{T}}. \tag{2}$$

$\dot{\mathbf{p}}$ can be written in three different ways by traversing the three chains A_iB_iP:

$$\dot{\mathbf{p}} = \mathbf{e}_i\dot{\rho}_i + (\dot{\theta}_i\mathbf{i}_i + \dot{\beta}_i\mathbf{j}_i) \times (\mathbf{b}_i - \mathbf{a}_i). \tag{3}$$

Here, $\mathbf{a}_i$ and $\mathbf{b}_i$ are the position vectors of the points A_i and B_i, respectively, and $\mathbf{e}_i$ is the direction vector of the linear joints, for $i = 1, 2, 3$.

We want to eliminate the three idle joint rates $\dot{\theta}_i$ and $\dot{\beta}_i$ from eq. (3), which we do by dot-product of eq. (3) by $\mathbf{b}_i - \mathbf{a}_i$:

$$(\mathbf{b}_i - \mathbf{a}_i)^{\mathrm{T}}\dot{\mathbf{p}} = (\mathbf{b}_i - \mathbf{a}_i)^{\mathrm{T}}\mathbf{e}_i\dot{\rho}_i. \tag{4}$$

Equation (4) can now be cast in vector form, namely $\mathbf{A}\dot{\mathbf{p}} = \mathbf{B}\dot{\boldsymbol{\rho}}$, where $\mathbf{A}$ and $\mathbf{B}$ are the parallel and serial Jacobian matrices, respectively:

$$\mathbf{A} = \begin{bmatrix} (\mathbf{b}_1 - \mathbf{a}_1)^{\mathrm{T}} \\ (\mathbf{b}_2 - \mathbf{a}_2)^{\mathrm{T}} \\ (\mathbf{b}_3 - \mathbf{a}_3)^{\mathrm{T}} \end{bmatrix} \text{ and } \mathbf{B} = \begin{bmatrix} \eta_1 & 0 & 0 \\ 0 & \eta_2 & 0 \\ 0 & 0 & \eta_3 \end{bmatrix} \tag{5}$$

with $\eta_i = (\mathbf{b}_i - \mathbf{a}_i)^{\mathrm{T}}\mathbf{e}_i$ for $i = 1, 2, 3$.

Parallel singularities occur when the determinant of the matrix $\mathbf{A}$ vanishes, i.e., when $\det(\mathbf{A}) = 0$. Equation (5) shows that the parallel singularities occur when

$$(\mathbf{b}_1 - \mathbf{a}_1) = \alpha(\mathbf{b}_2 - \mathbf{a}_2) + \lambda(\mathbf{b}_3 - \mathbf{a}_3), \tag{6}$$

that is, when the points A_1, B_1, A_2, B_2, A_3, and B_3 lie in parallel planes. A particular case occurs when the links A_iB_i are parallel:

$$\begin{aligned} (\mathbf{b}_1 - \mathbf{a}_1) \quad &|| \quad (\mathbf{b}_2 - \mathbf{a}_2) \quad \text{and} \\ (\mathbf{b}_2 - \mathbf{a}_2) \quad &|| \quad (\mathbf{b}_3 - \mathbf{a}_3) \quad \text{and} \\ (\mathbf{b}_3 - \mathbf{a}_3) \quad &|| \quad (\mathbf{b}_1 - \mathbf{a}_1). \end{aligned}$$

Serial singularities arise when the serial Jacobian matrix $\mathbf{B}$ is no longer invertible, i.e., when $\det(\mathbf{B}) = 0$. At a serial singularity a direction exists along which no Cartesian velocity can be produced. Equation (5) shows that $\det(\mathbf{B}) = 0$ when for one leg i, $(\mathbf{b}_i - \mathbf{a}_i) \perp \mathbf{e}_i$.

When $\mathbf{B}$ is not singular, we can write

$$\dot{\boldsymbol{\rho}} = \mathbf{J}^{-1}\dot{\mathbf{p}} \text{ with } \mathbf{J}^{-1} = \mathbf{B}^{-1}\mathbf{A}. \tag{7}$$

4. Comparison of Orthoglide and UraneSX

In this section, we calculate the edge length of the largest cube for the Orthoglide and the edge length of the largest square for the UraneSX, as well as the location of their respective centers. To simplify the problem, the bounds on the velocity transmission factors are such that $\psi_{min} = 1/\psi_{max}$.

4.1. Regular Dextrous Workspace Shape

The Orthoglide and the UraneSX are compared according to the size of their largest regular dextrous workspace. Because of the symmetrical architecture of the Orthoglide, the Cartesian workspace has a fairly regular shape in which it is possible to include a cube whose sides are parallel to the planes xy, yz, and xz, respectively. The Cartesian workspace of the UraneSX is the intersection of three cylinders whose axes are parallel to the z-axis. Thus, the workspace is unlimited in the z-direction and the Jacobian matrix does not depend on the z-coordinate. Only the limits on the linear joints define the limits of the Cartesian workspace in the z-direction. However, it is possible to include a square in the plane xy. Regular dextrous workspaces are thus chosen to be a cube for the Orthoglide and a square for the UraneSX.

4.2. Geometric Constraints

Section 2.1 is suggested to add geometrical constraints so as to decrease the computing time and to avoid numerical problems on singularities. Here, polynomial $\mathcal{P}$ is defined only for the points within the intersection $\mathcal{I}$ of the three cylinders defined by

$$x^2 + y^2 < L^2 \quad x^2 + z^2 < L^2 \quad y^2 + z^2 < L^2 \tag{8}$$

for the Orthoglide, and

Table 1. Computation Time of the Largest Cube Enclosed in the Dextrous Workspace for Various Accuracies

Accuracy α (mm)	0.01	0.001	0.0001	0.00001
Computation time (s)	360	150	504	900

$$
\begin{aligned}
(x - R + r)^2 + y^2 &< L^2 \\
\left(x - (R-r)\frac{1}{2}\right)^2 + \left(y - (R-r)\frac{\sqrt{3}}{2}\right)^2 &< L^2 \\
\left(x - (R-r)\frac{1}{2}\right)^2 + \left(y + (R-r)\frac{\sqrt{3}}{2}\right)^2 &< L^2
\end{aligned}
$$

for the UraneSX. With these constraints, matrix **B** is never singular and thus can be always inverted. To solve numerically the above equations and to compare the two mechanisms, the length of the legs is normalized, i.e., we set $L = 1$.

4.3. Comparison Results

To compare the two mechanisms studied, the leg length L is set to 1 and the bounds on the velocity factor amplification are set to $\psi = [0.5\ 2]$, with $\alpha = 0.001$. For the UraneSX, it is necessary to define two additional lengths, r and R. However, the edge length of the workspace depends only on $R - r$.

For the Orthoglide, it is found that the largest cube has its center located at (0.086, 0.086, 0.086), and that the cube edge length is $L_{Workspace} = 0.644$. Also, using the incremental approach described in Section 2.4, we obtain for the Orthoglide the computation time of Table 1 on a Sun Blade workstation.

For the UraneSX, the design parameters are those defined in Company and Pierrot (2002), which we have normalized to have $L = 1$, i.e., $r = 3/26$ and $R = 7/13$. To compare the two mechanisms, we increase the value of R such that $R' = R + \lambda$ with $\lambda = [0.0, 0.2]$. For $R < 7/13$, the constraints on the velocity amplification factors are not satisfied.

The optimal value of R' is obtained for $\lambda = 0$, i.e., for the design parameters defined in Company and Pierrot (2002) for an industrial application (see Table 2). To expand this square workspace in the z-direction, the range limits must be equal to the edge length of the square plus the range variations necessary to move throughout the square in the x–y plane.

The constraints on the velocity amplification factors used for the design of the Orthoglide are close to those used for the design of the UraneSX which is an industrial machine tool. For the same length of the legs, the size of the cubic workspace is larger for the Orthoglide than for the UraneSX.

For the Orthoglide, the optimization puts the serial and parallel singularities far away from the Cartesian workspace (Chablat and Wenger 2003). The UraneSX has no parallel singularities due to the design parameters ($R - r < L$), but serial singularities cannot be avoided with the previous optimization

Table 2. Variations of the Edge Length of the Square Workspace for the UraneSX Mechanism

λ	Center	$L_{Workspace}$
0.00	(−0.0178, −0.0045)	0.510
0.05	(−0.0179, −0.0022)	0.470
0.10	(−0.0225, −0.0031)	0.420
0.15	(−0.0245, −0.0018)	0.370
0.20	(−0.0211, −0.0033)	0.320

function. To produce the motion in the z-direction, the range limits of the linear joints are set such that the constraints on the velocity amplification factors are not satisfied throughout the Cartesian workspace.

The range limits $\Delta\rho_i$ of each prismatic joint can be decomposed into two parts. For the Orthoglide and the UraneSX respectively, the first part Δf_i makes it possible to move the mobile platform throughout the face of the prescribed cube and prescribed square, respectively, which is perpendicular to the considered prismatic joint axis. The second part is equal to the edge length of the cubic workspace $L_{Workspace}$. The equations of the inverse kinematic model allow us to compute Δf_i for the two mechanisms.

For the Orthoglide, the position and the size of the prescribed cube define three range limits for the x–y–z platform coordinates:

$$
\begin{aligned}
x &= [-0.322 + 0.085, 0.322 + 0.085] && \text{(9a)} \\
y &= [-0.322 + 0.085, 0.322 + 0.085] && \text{(9b)} \\
z &= [-0.322 + 0.085, 0.322 + 0.085]. && \text{(9c)}
\end{aligned}
$$

For the UraneSX, the position and the size of the prescribed square define two range limits for the x–y platform coordinates:

$$
\begin{aligned}
x &= [-0.255 - 0.018, 0.255 - 0.018] && \text{(9d)} \\
y &= [-0.255, 0.255]. && \text{(9e)}
\end{aligned}
$$

For the Orthoglide, all Δf_i are equal due to the symmetrical architecture. For the UraneSX, we take $\Delta f = Max(\Delta f_i)$. The results are $\Delta f = 0.181$ and $\Delta\rho = 0.825$ for the Orthoglide and $\Delta f = 0.353$ and $\Delta\rho = 0.863$ for the UraneSX. This means that the range limits are quite similar for the same leg length. To calculate the volume of the Cartesian workspace of the two mechanisms for the previous range limits, we have used a CAD system. Results are given in Table 3.

To help understand these results, Figures 3 and 4 show the location of the largest cubic workspace inside the Cartesian

Table 3. Workspace Volumes of the Two Mechanisms

	Cartesian Workspace Volume With Optimized Ranges Limits	Cubic Dextrous Workspace Volume	Ratio
Orthoglide	0.566	0.265	46.8%
UraneSX	0.544	0.132	24.3%

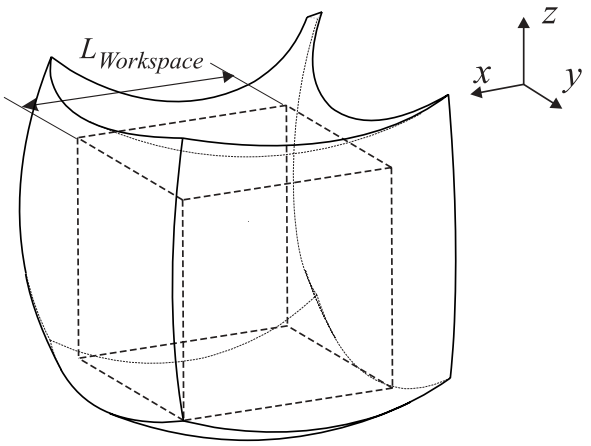

Fig. 3. Cartesian workspace and dextrous workspace for the Orthoglide mechanism with optimized range limits.

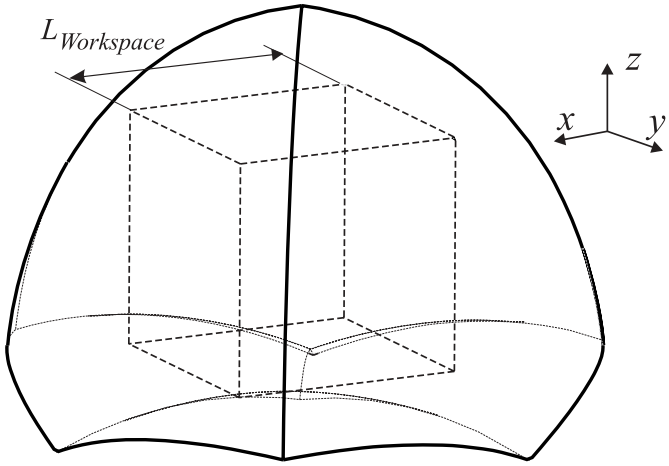

Fig. 4. Cartesian workspace and dextrous workspace for the UraneSX mechanism with optimized range limits.

workspace. As the Cartesian workspace of the Orthoglide is regular and admits a quasi-cubic shape, the ratio between the cubic workspace and the Cartesian workspace is better than for the UraneSX.

In Table 4, the design parameters are compared to achieve the same cubic dextrous workspace with $L_{Workspace} = 1$. The leg length is directly connected to the dynamic properties of the mechanism. The range limits and the leg length are important parameters in the determination of the total size of the mechanism and in its global cost. The volume of the Cartesian workspace allows us to characterize the shape and the volume of motion of the tool with regard to the useful Cartesian workspace dedicated to manufacturing tasks (cubic workspace).

These criteria allow us to optimize some geometric parameters to design a machine tool for milling applications. Although in this approach the kinetostatic properties of the Orthoglide are better than those of UraneSX, we cannot assert that the Orthoglide is better than the UraneSX. One reason is that these two PKMs are not aimed at identical manufacturing tasks. The main applications of the UraneSX are drilling, facing, and tapping whereas the Orthoglide is more universal.

Other shapes of regular dextrous workspaces can be computed for the Orthoglide and the UraneSX by using cylindrical or spherical coordinates to have the largest cylinder or sphere respectively, even if these shapes are generally less relevant for milling applications.

5. Conclusions

In this paper we introduce an interval analysis based study for the design and the comparison of a 3-DoF PKM. Two basic tools and an algorithm are described to determine the largest regular dextrous workspace enclosed in the Cartesian workspace. The dextrous workspace is part of the Cartesian workspace in which the velocity amplification factors remain within a predefined range. This means that throughout the dextrous workspace, milling tool paths are available because the variations of the kinematic performances index remain under reasonable values. The regular dextrous workspace shape is a cube for the Orthoglide and a square for the UraneSX. This general method is coupled with geometric constraints associated with the mechanisms studied to avoid numerical problems at singular configurations. The shape of the dextrous workspace was chosen for milling applications but it can be different for other applications. The range limits and the volume of the Cartesian workspace were calculated to compare the two mechanisms.

Table 4. Synthesis of the Comparative Study For the Same Cubic Cartesian Workspace

	Leg Length	Range Limits	Volume of the Cartesian Workspace
Orthoglide	1.55	1.28	2.13
UraneSX	1.96	1.69	4.12

Acknowledgments

This research was supported by the CNRS (Project ROBEA "Machine à Architecture compleXe"). We would like to thank the anonymous reviewers for their very useful comments.

References

Carricato, M., and Parenti-Castelli, V. 2002. Singularity-free fully-isotropic translational parallel mechanisms. *International Journal of Robotics Research* 21(2):161–174.

Chablat, D., and Wenger, Ph. 2003. Architecture optimization of a 3-DoF parallel mechanism for machining applications: the Orthoglide. *IEEE Transactions on Robotics and Automation* 19(3):403–410.

Chablat, D., Wenger, Ph., and Merlet, J.-P. June 2002. Workspace analysis of the Orthoglide using interval analysis. *Proceedings of the 8th International Symposium on Advances in Robot Kinematics*, Caldes de Malavella, Spain. Kluwer Academic, Dordrecht, pp. 397–406.

Clavel, R. 1988. DELTA, a fast robot with parallel geometry. *Proceedings of the 18th International Symposium of Industrial Robots*, Lausanne, Switzerland, pp. 91–100.

Company, O., and Pierrot, F. 2002. Modelling and design issues of a three-axis parallel machine tool. *Mechanism and Machine Theory* 37:1325–1345.

Gosselin, C., and Angeles, J. 1991. A global performance index for the kinematic optimization of robotic manipulators. *Journal of Mechanical Design* 113:220–226.

Gosselin, C., and Angeles, J. 1988. The optimum kinematic design of a planar three-degree-of-freedom parallel manipulator. *ASME, Journal of Mechanisms, Transmissions, and Automation in Design* 110(1):35–41.

Joshi, S., and Tsai, L. W. 2003. A Comparison study of two 3-DoF parallel manipulators: one with three and the other with four supporting legs. *IEEE Transactions on Robotics and Automation* 19(2):200–209.

Kim, J., Park, C., Kim, J., and Park, F. C. 1997. Performance analysis of parallel manipulator architectures for CNC machining applications. *Proceedings of the IMECE Symposium on Machine Tools*, Dallas, TX.

Kong, X., and Gosselin, C. M. 2002. A class of 3-DoF translational parallel manipulators with linear I–O equations. *Proceedings of Workshop on Fundamental Issues and Future Research Directions for Parallel Mechanisms and Manipulators*, Québec, Canada (on CD-ROM).

Luh, C.-M., Adkins, F. A., Haug, E. J., and Qui, C. C. 1996. Working capability analysis of Stewart platforms. *Transactions of ASME, Journal of Mechanical Design* 118(2):221–227.

Merlet, J.-P. 1996. Workspace-oriented methodology for designing a parallel manipulator. *Proceedings of the IEEE International Conference on Robotics and Automation*, Minneapolis, MN, pp. 3726–3731.

Merlet, J.-P. 1999. Determination of 6D workspace of Gough-type parallel manipulator and comparison between different geometries. *International Journal of Robotic Research* 19(9):902–916.

Merlet, J.-P. June 2000. ALIAS: an interval analysis based library for solving and analyzing system of equations. Séminaire Systèmes et équations algébriques, Toulouse, *Automation* pp. 1964–1969.

Moore, R. E. 1979. *Methods and Applications of Interval Analysis*. SIAM Studies in Applied Mathematics, Philadelphia, PA.

Neumaier, A. 1990. *Interval Methods for Systems of Equations*. Cambridge University Press, Cambridge.

Ratscheck, H., and Rokne, J. 1995. Interval methods. *Handbook of Global Optimization*. R. Horst and P. M. Pardalos, editors. Kluwer, Dordrecht, pp. 751–819.

Rehsteiner, F., Neugebauer, R., Spiewak, S., and Wieland, F. 1999. Putting parallel kinematics machines (PKM) to productive work. *Annals of the CIRP* 48(1):345–350.

Stoughton, R. S., and Arai, T. 1993. A modified Stewart platform manipulator with improved dexterity. *IEEE Transactions on Robotics and Automation* 9(2):166–173.

Tlusty, J., Ziegert, J., and Ridgeway, S. 1999. Fundamental comparison of the use of serial and parallel kinematics for machine tools. *Annals of the CIRP* 48(1):351–356.

Treib, T., and Zirn, O. 1998. Similarity laws of serial and parallel manipulators for machine tools. *Proceedings of the International Seminar on Improving Machine Tool Performance*, San Sebastian, Spain, Vol. 1, pp. 125–131.

Wenger, Ph., and Chablat, D. 1997. Definition sets for the direct kinematics of parallel manipulators. *Proceedings of the 8th International Conference on Advanced Robotics*, Monterey, CA, pp. 859–864.

Wenger, P., Gosselin, C., and Chablat, D. 2001. A comparative study of parallel kinematic architectures for machining applications. *Proceedings of the Workshop on Computational Kinematics 2001*, Seoul, Korea, pp. 249–258.