A Statistical Approach for the Computation of the Forward Kinematic Model of Redundantly Actuated Mechanisms
Frédéric Marquet, François Pierrot, Olivier Company

To cite this version:

HAL Id: lirmm-00269463
https://hal-lirmm.ccsd.cnrs.fr/lirmm-00269463
Submitted on 3 Apr 2008

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

L’archive ouverte pluridisciplinaire HAL, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d’enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.
A statistical approach for the computation of the forward kinematic model of redundantly actuated mechanisms

Frédéric Marquet, François Pierrot, Olivier Company

LIRMM - UMR 5506 CNRS / UM2
161, rue Ada - 34392 Montpellier cedex 5, France
<marquet, pierrot>@lirmm.fr

Abstract – The forward kinematic model (FKM) of a redundantly actuated robot is not unique: if each actuator is equipped with an encoder, there are more joint data than strictly necessary for computing nacelle position. It is then possible to fuse competitive data to find the nacelle position. This paper proposes then a method based on a probabilistic approach to determine how computing the FKM to obtain, in terms of probability, the lowest Cartesian error.

1. Introduction

The problem of the FKM computation of redundantly actuated robots can appear as quite innocuous. In fact, these robots have more actuators than degrees of freedom (dof): there are more sensors data than necessary to find the Cartesian position. As there are several ways for computing the FKM (using the complete set or, for instance, only the minimal number of sensors data), because of the errors on mechanism parameters (arms length, nacelle dimensions…), each algorithm generates to its own Cartesian error. For redundantly actuated robots, this is an important point because the controller must necessarily be implemented in the Cartesian space: a joint controller would lead to drives forces divergence because of the presence of its integral element [1][2][3]. For such a control, except if the Cartesian error is computed with the jacobian matrix evaluated for the desired position, the FKM has to be used directly in the control loop. The accuracy of the solution it gives is then of the utmost importance.

This paper proposes a probabilistic approach to determine the best computation algorithm. With this aim in view, we propose to determine the models sensibilities to variations of their parameters (as mechanism geometrical dimensions or drives positions) and then to compute the associated Cartesian standard deviation (std) when assuming that parameters are normally distributed with a given covariance matrix.

This paper is divided into 6 sections.

Section 2 introduces the problem of the FKM computation and gives some possible ways for its resolution. The choice of a probabilistic approach for determining how to solve it is then discussed.

Section 3 permits to put in place the necessary mathematical tools related to the problem i.e. covariance matrices, transformation of random variables by a linear application and advantages of averaging.

Section 4 is dedicated to the choice of a FKM strictly speaking. The equations giving the errors generated by computing a given model and the corresponding Cartesian error are, in particular, derived.

Section 5 proposes the application of the developed approach to the choice of the FKM of ARCHI robot, a 3-dof mechanism actuated by four drives.

Lastly, conclusions are given in section 6.

2. Problem of the FKM computation and choice of a probabilistic approach

2.1. Non-uniqueness of the FKM

A redundantly actuated mechanism has more drives than its number of dof. To perform its Cartesian position, different computations are possible, using or not all joint data coming from the encoders.

For instance, let's consider the 1-dof over-actuated mechanism presented in Figure 1:

![Figure 1. 1-dof mechanism actuated by two drives (P: Prismatic joint, R: Revolute joint)](image)

Nacelle position can obviously be computed with the two following equations:

\[ x_1 = FKM_1(q_1, L_1) = \sqrt{L_1^2 - q_1^2} \]
\[ x_2 = FKM_2(q_2, L_2) = \sqrt{L_2^2 - q_2^2} \]  

If the mechanism is perfect (i.e. has no errors on geometrical parameters), these two models give the same Cartesian position. Of course in reality it's not the case (errors on arms length or on drives position) and the models are not equivalent at all.

It's also possible to compute the average of the two solutions given by (1) or to calculate it by weighting the solutions taking into account the jacobian condition number of the sub-mechanisms composed of only one of the two
arms (obviously, for any mechanism, when the condition number increases the precision decreases).

The FKM can as well be computed iteratively. For an over-actuated mechanism, the relation between joint and Cartesian velocities can be expressed as:

\[ \dot{q} = J_m \dot{x} \]  

where \( J_m \) is a rectangular matrix.

To calculate the Cartesian velocities, it is then necessary to solve an over-determined linear system. In such a way, it is possible to find its least square (LSQ) solution:

\[ \dot{x}_{LSQ} = J_m^{-1} \dot{q} \]

where the operator "\(^{-1}\)" denotes the pseudo-inversion [4]. The iterative equation that gives the Cartesian position is then:

\[ x_{n+1} = x_n + J_m^{-1} (x_n \dot{q}_n) \]

Notice that the algorithm stop condition can’t be \( \| \dot{q} - \dot{q}_n \| < \varepsilon \) as usual for non-redundant mechanisms but has to be \( \| x - x_n \| < \varepsilon \) because as the mechanism is not perfect drives can't reach the exact desired joint position (encoders data used measure the geometrical errors consequences).

Other solutions are also imaginable, as an example finding directly the LSQ solution to the non-linear system composed of the equations corresponding to the two arms or the solution that generates the lowest std.

There are then many different ways for computing the Cartesian position of a redundant mechanism.

2.2. Probabilistic approach for choosing a FKM

Let’s suppose that the robot direct and inverse kinematic models are:

\[ x = FKM(q, P_{mech}) \]
\[ q = IKM(x, P_{mech}) \]

where \( P_{mech} \) is the vector containing the \( N \) geometrical parameters of the mechanism as arms lengths or nacelle dimensions for example.

For a given nacelle position \( x \), the Cartesian error due to an error \( dP_{mech} \) of the geometrical parameters and \( dP_{act} \) of actuators locations (due, for instance, to the encoders offset error) is (Figure 2):

\[ dx = FKM(IKM(x, P_{mech} + dP_{mech}) + dP_{act}, P_{mech}) - x \]

\[ \text{Figure 2. Cartesian error generated by errors on models parameters} \]

Considering a given interval error for all the components of \( P_{mech} \) and \( P_{act} \), it is then possible to perform the error interval with respect to \( dx \). Nevertheless, this approach has a drawback: it considers the worst case.

The approach developed here is a little bit different: it takes into account a given normal probability density distribution of the geometrical parameters and actuators locations (it has been verified time and again that a calibration or a measure lead to such a probability density distribution) characterized by a given covariance matrix to find the one generated in the Cartesian space.

3. Mathematical tools, elements of probabilities

This section describes the necessary mathematical tools for choosing a model. In particular, the notions of probability distributions, covariance matrices, correlation between variables and transformation of random variables by linear applications are recalled.

3.1. Covariance matrix and random variable transformation by a linear application

Let’s consider a vector \( x = [x_1, ..., x_n]^T \) containing \( n \) random values. Its covariance matrix is defined by [5]:

\[ C_x = E[(x - \bar{x})(x - \bar{x})^T] \]

where \( E[...] \) represents the expectation value. The covariance matrix \( C_x \) of \( x \) can also be written as:

\[ C_x = \begin{bmatrix}
\sigma_{x_1}^2 & \rho_{12}\sigma_{x_1}\sigma_{x_2} & \rho_{13}\sigma_{x_1}\sigma_{x_3} & \cdots \\
\rho_{12}\sigma_{x_1}\sigma_{x_2} & \sigma_{x_2}^2 & \rho_{13}\sigma_{x_2}\sigma_{x_3} & \cdots \\
\rho_{13}\sigma_{x_1}\sigma_{x_3} & \rho_{13}\sigma_{x_2}\sigma_{x_3} & \sigma_{x_3}^2 & \cdots \\
\cdots & \cdots & \cdots & \cdots 
\end{bmatrix} \]

where \( \rho_{ij} \) is the correlation coefficient [6] that describe the degree of relationship between \( x_i \) and \( x_j \) and \( \sigma_x \) the std of \( x_i \).

If the components of \( x \) are independent, correlation coefficients are equal to zero and the covariance matrix is diagonal.

Now, suppose that \( y \) is the image of the random vector \( x \) by the linear application associated to the \( m \)-by-\( n \) matrix \( A \):

\[ y = Ax, \quad A \in \mathbb{R}^{m \times n} \]

Its covariance matrix is then [7][8]:

\[ C_y = AC_xA^T \]

3.2. Advantages of averaging

If \( \bar{x} \) is the scalar corresponding to the average of \( x \) components:

\[ \bar{x} = \frac{1}{n} \sum_{i=1}^{n} x_i \]

the std of \( \bar{x} \) is then:

\[ \sigma^2(\bar{x}) = \frac{1}{n^2} \left( \sum_{i=1}^{n} \sigma_{x_i}^2 + 2 \sum_{i<j} \text{cov}(x_i, x_j) \right) \]

If the elements of \( x \) are independent and if their std are equal, it is easy to demonstrate that the std of the average is
lower than the one of each component:

\[ \sigma_l(\lambda) = \frac{1}{\sqrt{n}} \sigma_l. \]  

(13)

4. Choice of a model

4.1. Sensibility to the errors on parameters

Assume that \( f \) is the implicit function that gives the relation between the positions \( x \) and \( q \) of a \( n \)-dof robot actuated by \( m \) drives:

\[
f(x, q, P_{mech}) = 0 \iff \begin{cases}
  f_1(x, q, P_{mech}) = 0 \\
  f_m(x, q, P_{mech}) = 0
\end{cases}
\]

(14)

A differentiation of this equation leads to:

\[
J_x dx = J_{P_{mech}} dP_{mech} + J_q dq,
\]

(15)

with:

\[
J_x = \begin{bmatrix}
\frac{\partial f_1}{\partial x} & \cdots & \frac{\partial f_m}{\partial x}
\end{bmatrix}_{1 \times m},
\]

\[
J_{P_{mech}} = \begin{bmatrix}
\frac{\partial f_1}{\partial P_{mech}} & \cdots & \frac{\partial f_m}{\partial P_{mech}}
\end{bmatrix}_{1 \times m},\]

and:

\[
J_q = \begin{bmatrix}
\frac{\partial f_1}{\partial q} & \cdots & \frac{\partial f_m}{\partial q}
\end{bmatrix}_{1 \times m}.
\]

(16)

The Cartesian error defined by (6) due to \( dP_{mech} \) and \( dP_{act} \) is then:

\[
dx = \frac{\partial \text{FKM}}{\partial \mathbf{q}} \begin{bmatrix}
\frac{\partial \text{FKM}}{\partial P_{mech}} dP_{mech} + dP_{act}
\end{bmatrix} = J_{SP} dp,
\]

(17)

with:

\[
ds \mathbf{p} = \begin{bmatrix}
dP_{mech} \\
dP_{act}
\end{bmatrix},
\]

(18)

and:

\[
J_{SP} = J_x^{-1}\begin{bmatrix}
-J_{P_{mech}} & J_q
\end{bmatrix}.
\]

(19)

4.2. Cartesian error distribution

Assuming that the components of \( dp \) are independent and normally distributed with a mean equal to zero and a covariance matrix \( \mathbf{C}_p \) defined by:

\[
\mathbf{C}_p = \text{diag}([\sigma_{p_1}, \ldots, \sigma_{p_n}]),
\]

(20)

according to (10), the covariance matrix characterizing the probability density of the Cartesian error \( ds \) is:

\[
\mathbf{C}_{ds} = J_{SP} \mathbf{C}_p J_{SP}^T,
\]

(21)

and the \text{std} of the Cartesian error norm is given by:

\[
\sigma_{\lambda_{ds}} = \sqrt{\lambda_{\text{max}}(\mathbf{C}_{ds})},
\]

(22)

where \( \lambda_{\text{max}}(\mathbf{C}_{ds}) \) is \( \mathbf{C}_{ds} \) maximal eigen value.

For the considered probability density distribution, 68 % of the errors will belong to the interval \([-\sigma_{\lambda_{ds}}, \sigma_{\lambda_{ds}}] \) and

99.8 % to \([-3\sigma_{\lambda_{ds}}, 3\sigma_{\lambda_{ds}}] \).

To illustrate the Cartesian \text{std} computation, let's consider again the 1-dof robot presented in sub-section 2.1.

If \( \text{FKM} \) is one of its models depending on the geometrical parameters \( L_1 \) and \( L_2 \):

\[
x = \text{FKM}(q_1, q_2, L_1, L_2),
\]

(23)

the error defined by (17) is then:

\[
dx = \begin{bmatrix}
-\frac{\partial \text{FKM}}{\partial L_1} & -\frac{\partial \text{FKM}}{\partial L_2} & -\frac{\partial \text{FKM}}{\partial q_1} & -\frac{\partial \text{FKM}}{\partial q_2}
\end{bmatrix} \begin{bmatrix}
  \frac{dL_1}{dq_1} \\
  \frac{dL_2}{dq_1} \\
  \frac{dq_1}{dq_2} \\
  \frac{dq_2}{dq_2}
\end{bmatrix},
\]

(24)

and for errors characterized by \( \sigma_{L_1}, \sigma_{L_2}, \sigma_{q_1}, \) and \( \sigma_{q_2}, \) the corresponding Cartesian variance is:

\[
\sigma_{ds}^2 = \begin{bmatrix}
  & \sigma_{L_1}^2 & 0 & 0 \\
  & 0 & \sigma_{L_2}^2 & 0 \\
  & 0 & 0 & \sigma_{q_1}^2 \\
  & 0 & 0 & \sigma_{q_2}^2
\end{bmatrix},
\]

(25)

and therefore:

\[
\sigma_{ds}^2 = \begin{bmatrix}
  & \sigma_{L_1}^2 & \sigma_{L_2}^2 & 0 \\
  & \sigma_{L_2}^2 & \sigma_{L_1}^2 & 0 \\
  & 0 & 0 & \sigma_{q_1}^2 \\
  & 0 & 0 & \sigma_{q_2}^2
\end{bmatrix}
\]

(26)

Considering \( \text{FKM}_1 \) model using only the arm \( n'1 \), the corresponding variance is then:

\[
\sigma_{ds_1}^2 = \sigma_{s_1}^2 = \begin{bmatrix}
  \frac{L_1}{x_1} \\
  \frac{q_1}{x_1}
\end{bmatrix}^2 \sigma_{L_1}^2 + \begin{bmatrix}
  \frac{q_1}{x_1}
\end{bmatrix}^2 \sigma_{q_1}^2,
\]

(27)

and similarly for \( \text{FKM}_2 \).

In a same way, it is easy to demonstrate that the variance obtained by averaging the two models solutions is:

\[
\sigma_{ds}^2 = \frac{1}{4}(\sigma_{L_1}^2 + \sigma_{L_2}^2)
\]

(28)

and then, if \( \sigma_{L_1} = \sigma_{L_2} \) and \( \sigma_{q_1} = \sigma_{q_2} \) we have \( \sigma_{s_1} = \sigma_{s_2} = \sigma \) and:

\[
\sigma_{ds} = \sigma \sqrt{2}.
\]

(29)

As it could have been expected, averaging permits to reduce notably the \text{std}.

The \text{LSQ} solution \( x_{LSQ} \) is:

\[
x_{LSQ}^2 = \frac{1}{2}(x_1^2 + x_2^2),
\]

(30)
and therefore:

\[ dx_{LSQ} = \frac{1}{2x_{LSQ}}(x_1 dx_1 + x_2 dx_2) \approx dx. \] (31)

The std obtained with the LSQ solution is then close to the one generated by averaging.

Lastly, the FKM can be obtained by weighting the solutions performed with FKM1 and FKM2:

\[ x_w = w x_1 + (1-w) x_2. \] (32)

The weights can take into account the condition numbers of the sub-mechanisms jacobian matrices or they can be computed for minimizing the std.

In the last case, the variance is then the polynomial function:

\[ \sigma^2(w) = w^2 \sigma_1^2 + (1-w)^2 \sigma_2^2, \] (33)

whose minimal value is given by:

\[ w_{opt} = \frac{\sigma_2^2}{\sigma_1^2 + \sigma_2^2}. \] (34)

If \( \sigma_1 = \sigma_2 \), \( w_{opt} = 0.5 \): the lowest std corresponds to the average of \( x_1 \) and \( x_2 \).

The different std for \( x \in [0.3;0.8] \) are plotted in Figure 3.

5. Application to the computation of the FKM of ARCHI, a redundantly actuated 3-dof robot

5.1. Presentation of ARCHI robot

ARCHI [9] is a 3-dof robot (allowing the nacelle two translations and one unlimited rotation) actuated by 4 linear drives (Figure 4).

![ARCHI robot and its joint and loops graph (U: Universal joint, S: Spherical joint)](image)

5.2. FKM computations

Many different ways are possible for computing the FKM, such as:
- use of only 3 joint positions;
- sub-mechanisms FKM solutions weighting;
- direct use of the hole sensors data directly;
- iterative model;
- LSQ minimization.

Those models are first derived and then the Cartesian error they generate are compared.

5.3. FKM using only 3 joint positions

Here, the minimal number of data is used. The model is computed by finding circles intersections. For instance, to compute the FKM corresponding to the arms n o 1, 2 and 3 \( (x_{123} = FKM_{123}(q, P_{mech})) \) so called \( x_4 = FKM_{4}(q, P_{mech}) \), the positions of \( B_{12} \) and \( B_{34} \) are given by:

\[ B_{12} = C_1([q_1,0], L_1) \cap C_2([q_2,0], L_2) \]
\[ B_{34} = C_3([x_{12}, y_{12}, 2D]) \cap C_4([q_3,0], L_3). \] (35)

When the positions of \( B_{12} \) and \( B_{34} \) are known, the Cartesian position can be calculated thanks to the following equations:

\[ x = \frac{x_{12} + x_{34}}{2} \]
\[ y = \frac{y_{12} + y_{34}}{2} \]
\[ \theta = \tan^{-1}((y_{34}-y_{12})/(x_{34}-x_{12})). \] (36)

A differentiation of the above relations leads to the equation that, according to (17), gives the Cartesian error due to errors on \( dP = \begin{bmatrix} dL_1 & ... & dL_4 & dD & dq_1 & ... & dq_4 \end{bmatrix}^T \):

\[ dx_4 = J_{SP} dP. \] (37)

The covariance matrix \( C_{dx_4} \) associated to this model is then given by (21).

The 3 reminding models (FKM1, FKM2, and FKM3) and their associated covariance matrices are calculated similarly.
5.4. FKM using the whole joint positions

5.4.1. FKM obtained by solutions weighting
A direct averaging as:

$$\bar{x} = \frac{1}{d} \sum_{i=1}^{d} x_i$$  \hspace{1cm} (38)

will generate the covariance matrix:

$$C_{\delta x} = J_{SP}C_pJ_{SP}^T = \frac{1}{d^2} \left( \sum_{i=1}^{d} C_{\delta x_i} + \sum_{i,j} J_{SP_i}C_pJ_{SP_j} \right).$$  \hspace{1cm} (39)

However, this solution is not suitable because of the presence of sub-mechanisms singularities (close to those locations, the errors due to the model become very high, see section 5.5.3). A solution can then consists in weighting sub-mechanisms FKM solutions with their condition number:

$$x_{w} = \sum_{i=1}^{d} w(\text{cond}(J_i))x_i / \sum_{i=1}^{d} w(\text{cond}(J_i)),$$  \hspace{1cm} (40)

with, for example:

$$w(\text{cond}(J_i)) = \text{cond}(J_i)^{-k},$$  \hspace{1cm} (41)

choosing $k = 1$ or an higher value for taking less into account the solutions of sub-mechanisms close to singular positions. There is still a problem with this formulation: if a sub-mechanism is close to a singularity, the existence of a solution is not ensured because the vector of joint positions comes from sensors measures (and than takes into account the geometrical parameters errors). Considering for example the sub-mechanism composed of arms $n^o 1$, 2 and 3, it is possible that $C_{\delta x}[[x_{12} \, y_{12}], 2D]$ and $C_{\delta x}[[q_3], L_3]$ would not be neither secant nor tangent. It is then necessary to eliminate the sub-mechanism close to a singularity. This solution becomes nonetheless quite heavy to implement.

5.4.2. LSQ minimization

LSQ minimization corresponds to the LSQ solution to the over-determined system:

$$\begin{cases}
(x - D \cos \theta - q_1)^2 + (y - D \sin \theta)^2 = L_1^2 \\
(x - D \cos \theta - q_2)^2 + (y - D \sin \theta)^2 = L_2^2 \\
(x + D \cos \theta - q_3)^2 + (y + D \sin \theta)^2 = L_3^2 \\
(x + D \cos \theta - q_4)^2 + (y + D \sin \theta)^2 = L_4^2.
\end{cases}$$  \hspace{1cm} (42)

It is implemented with a classical gradient optimization algorithm. The sensibility matrix to the model parameters and the covariance are then easy to deduce.

5.4.3. Iterative method using the jacobian matrix

This solution is as always possible. The algorithm is implemented as described in section 2.1.

5.4.4. Use of the hole sensors data directly

This model is easily obtained by calculating the intersections of circles of radius $L_i$ centered on the points $A_i[q_i, 0]$ (when considering no error on drives positions about $y$):

$$\begin{align*}
(x_{12} - q_1)^2 + y_{12}^2 &= L_1^2 \\
(x_{12} - q_2)^2 + y_{12}^2 &= L_2^2 \\
(x_{34} - q_3)^2 + y_{34}^2 &= L_3^2 \\
(x_{34} - q_4)^2 + y_{34}^2 &= L_4^2
\end{align*}$$

\[ \begin{pmatrix} B_{12} \\ B_{34} \end{pmatrix} = \left( B_{12} + B_{34} \right) / 2. \]  \hspace{1cm} (43)

By differentiation, the sensibility matrix related to model parameters defined by (19) can be found and consequently the covariance matrix calculated.

5.5. Models comparison

5.5.1. Cartesian error / end-tool error

The vector $x$ is not homogenous. For this reason, the Cartesian error considered is the end-tool position (point $P$, see Figure 4).

The std of the 2 coordinates of point $P$ (e.g. $x_P$ and $y_P$) is computed as well as the scalar:

$$\sigma = \sqrt{\sigma_{x_P}^2 + \sigma_{y_P}^2}.$$  \hspace{1cm} (44)

5.5.2. Hypothesis on parameters errors

The geometrical parameters of the robot are the one of the prototype that has been constructed$^1$: $L = 0.88$ $m$ and $D = 0.055$ $m$. For FKM comparisons, the errors on geometrical parameters ($L_1...L_4$ and $D$) and drives positions ($q_1...q_4$) are supposed to be normally distributed with a mean equal to zero and std equal to 1 $mm$.

5.5.3. Errors generated by the FKM using 3 joint positions

The errors are computed for $y = -0.5$ $m$ and $\theta \in [0;90]$ degrees (the error doesn't depend on the position about $x$). The std of the Cartesian error and the condition number of the sub-mechanism composed of arms $n^o 1$, 2 and 3 are plotted in Figure 5.

![Figure 5. Std of the Cartesian error generated by the FKM of a sub-mechanism of ARCHI robot and its corresponding condition number](http://www.lirmm.fr/~marquet/)

It is clear that when approaching a singularity, $\sigma$ increases quickly: it's not possible to use only 3 of the 4 drives positions to compute the FKM.

5.5.4. LSQ minimization vs iterative model using the jacobian matrix

These two solutions have been compared for the same std of the models parameters. Std of the Cartesian errors have been computed by a random draw of 200 values with a normal probability density distribution for each robot.

---

$^1$ http://www.lirmm.fr/~marquet/
location (these locations, corresponding or not to sub-mechanisms singularities, have been chosen at random inside the workspace).

It has to be noticed that for several locations the gradient algorithm implemented for finding the LSQ solution converges to a local minimum that doesn’t correspond to the robot configuration.

Table 1 sums-up a part of the results obtained.

<table>
<thead>
<tr>
<th>Pos.</th>
<th>σ (mm) FKM</th>
<th>σ (mm) LSQ</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>2.23</td>
<td>1.74</td>
</tr>
<tr>
<td>2</td>
<td>2.00</td>
<td>2.01</td>
</tr>
<tr>
<td>3</td>
<td>1.71</td>
<td>1.74</td>
</tr>
<tr>
<td>4</td>
<td>1.95</td>
<td>1.88</td>
</tr>
<tr>
<td>5</td>
<td>1.65</td>
<td>1.64</td>
</tr>
<tr>
<td>6</td>
<td>2.47</td>
<td>2.44</td>
</tr>
<tr>
<td>7</td>
<td>2.03</td>
<td>2.02</td>
</tr>
</tbody>
</table>

The scalar σ has been computed for robot Cartesian positions defined in sub-section 5.5.3. It is given by a theoretical equation for the FKM using directly the whole drives positions and obtained by a random draw of 200 values for the iterative model (Figure 6).

When one sub-mechanism is singular (positions 2 and 3 in the table), σ keeps small for the two models and both std are nearly the same. For the 7 positions which have been chosen at random, σ mean value is about 2 mm: it’s half the one obtained for the FKM using only 3 drives positions (about 4 mm, see Figure 5) even when no sub-mechanism is singular. This proves that the use of the minimal number of drives positions is not suitable even when the corresponding sub-mechanism is not singular.

5.5.5. Iterative model vs FKM using directly the whole drive positions

The scalar σ has been computed for robot Cartesian positions defined in sub-section 5.5.3. It is given by a theoretical equation for the FKM using directly the whole drives positions and obtained by a random draw of 200 values for the iterative model (Figure 6).

Errors are still nearly the same for the two FKM: about 2-2.5 mm and sub-mechanisms singular positions have no effect.

6. Conclusion

When the exact dimensions of the mechanism are unknown, the probabilistic approach presented in this paper permits to compare the error probability generated by the use of a given model for finding the nacelle position of a redundantly actuated robot.

The FKM of ARCHI studied proved that neither the use of the minimal number of drives positions nor the FKM obtained by direct averaging are suitable at all. The other models led to std nearly the same. The LSQ solution is tricky to implement because of the possible convergence of the gradient minimization algorithm to local minimum and has no special interest compared to the iterative model that uses the jacobian matrix. This last solution or the one using directly the whole drives positions would then be preferred.

References: