Guzmán-Giménez, J.; Valera Fernández, Á.; Mata Amela, V.; Díaz-Rodríguez, MÁ. (2023). Automatic selection of the Groebner Basis' monomial order employed for the synthesis of the inverse kinematic model of non-redundant open-chain robotic systems. Mechanics Based Design of Structures and Machines. 51(5):2458-2480. https://doi.org/10.1080/15397734.2021.1899829

RESEARCH ARTICLE

Automatic Selection of the Groebner Basis' Monomial Order employed for the Synthesis of the Inverse Kinematic Model of Non-Redundant Open-Chain Robotic Systems

J. Guzmán-Giménez[a], A. Valera Fernández[a], V. Mata Amela[b] and M. A. Díaz-Rodríguez[c]

[a]Instituto de Automática e Informática Industrial, Universitat Politècnica de València, Camino de Vera, s/n, 46022 Valencia, Spain; [b]Centro de Investigación en Ingeniería Mecánica, Universitat Politècnica de València, Camino de Vera, s/n, 46022 Valencia, Spain; [c]Departamento de Tecnología y Diseño, Universidad de Los Andes, 5101 Mérida, Venezuela

ARTICLE HISTORY
Compiled April 21, 2021

ABSTRACT
The most commonly used method employed to synthesize the IKM of open-chain robotic systems is based on geometry methods. However, these methods strongly depend on the geometry of the analyzed robot, making it difficult to systematize. In previous work, we devised a systematic approach relying on Groebner Bases to synthesize the IKM of non-redundant open-chain robotic systems. Nevertheless, this study expands further the developed procedure. Specifically, we develop a methodology for the automatic selection of the basis' monomial order. The procedure's inputs are the robot's Denavit-Hartenberg parameters, while the outputs are the mathematical equations defining the IKM in a form that can be directly used for the control or simulation of the robot. The developed procedure can be applied to synthesize the IKM of several types of robot systems such as Cartesian, SCARA, non-redundant, and multi-legged walking robot, and all the non-redundant robotic manipulators satisfying the in-line wrist condition. The performance of the proposed approach is evaluated through two study cases considering non-redundant open-chain robotic systems: a walking hexapod robot and a PUMA serial robot. The optimal monomial order is successfully identified for all cases. The output errors of all the synthesized IKMs are negligible when evaluated in their corresponding workspaces, while their computation times are comparable to those required by the kinematic models calculated following traditional methods.

KEYWORDS
Kinematic Problem; Inverse Kinematic Model (IKM); Groebner Basis; monomial ordering; non-redundant open-chain robotic systems

AMS CLASSIFICATION
70B15, 93C85

PACS CLASSIFICATION
45.40.Ln, 45.40.Aa, 45.40.Bb

CONTACT: J. Guzmán-Giménez. Email: joguz@upvnet.upv.es

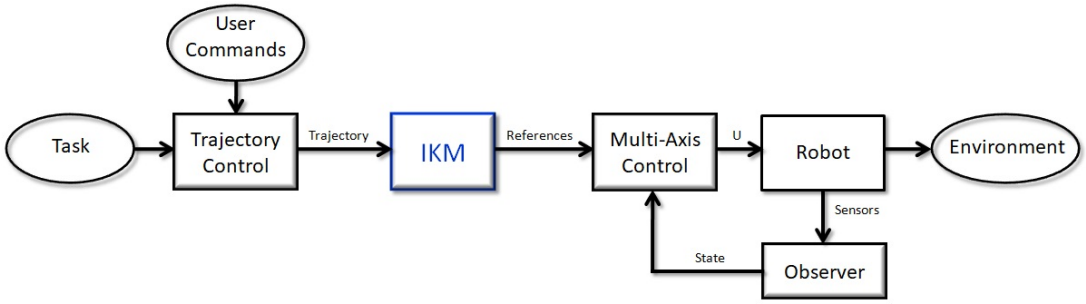

Figure 1. Inverse Kinematic Model (IKM) as a part of the robot's control system. The IKM's outputs are the references for the robot's multi-axis control.

1. Introduction

The resolution of a robot's Kinematic Problem can be divided in two parts: the Forward Kinematics Problem (FKP) and the the Inverse Kinematic Problem (IKP). The solution of the FKP, commonly referred as the robot's Forward Kinematics, is the equation system that calculates the pose of a specific point of the robot's structure according to its current state. This point is normally an important one in its structure, such as the end effector of a robotic manipulator or the center of mass of a mobile robot.

The Forward Kinematics is necessary for modeling the robot's movements, while it is also important for the synthesis of the Inverse Kinematic Model (IKM). The function of this IKM, synthesized by solving the IKP, is to calculate the position and velocity references required by the robot's actuators to follow a trajectory. The IKM, as shown in Figure 1, is a fundamental part of the robot's control system.

The Forward Kinematics of an open-chain robot can be easily calculated by the application of different systematic procedures, which include Denavit-Hartenberg's algorithm (*1–3*), dual quaternions (*4–6*) and the modeling by Displacement Matrices (*7*). All these procedures are completely independent of the mechanical complexity of the robot's structure or its geometry.

In contrast, the techniques most commonly used to solve the IKP of open-chain robotic systems, the geometric method and the analytical procedure, strongly depend on the robot's geometry (*8*). They synthesize the robot's IKM by either separating the IKP into several plane geometry problems (*3*), or by solving the state vector of the robot's actuators in the equation system that defines the Forward Kinematics (*9–13*). But in both cases, these techniques depend heavily on the geometry of the robot's structure, which implies that they are not systematic procedures that can be equally applied to all the possible cases.

To address this issue, various works have opted to use different artificial intelligence techniques to solve the IKP of open-chain robotic systems, such as Neural Networks (*14–16*), Particle Swarm Optimization (PSO) (*17*), PSO-optimized Neural Networks (*18*), Neuro-evolutionary Algorithms (combination of Neural Networks and Genetic Algorithms) (*19*) and PSO variants (*20*, *21*). While these procedures satisfactorily solve the problem, they could suffer from the known training problems of artificial intelligence techniques, such as the over-fitting of Neural Networks and Genetic Algorithms. It is also important to bear in mind that the solution offered by these procedures is not a properly defined IKM. This is because the offered solution is not composed of fully differentiable functions, therefore it will not be able to calculate the

speed or acceleration references for the robot's actuators.

In order to implement systematic solutions for the Kinematic Problem, several projects are being developed that use Groebner Basis theory (*22*). These works calculate a Groebner Basis from the analyzed robot's kinematic equations, to simplify the process of solving the Kinematic Problem and, if possible, find an analytical solution. The works of Kendricks (*23*) and Wang et al. (*24*) present a systematic method to solve the IKP of robotic manipulators using Groebner Bases. Rameau et al. (*25*) use this theory to calculate the mobility conditions of several mechanisms that can be used in robotic arms or parallel robots, while Abbasnejad et al. (*26*) use it for the kinematic analysis of cable-driven robots. Groebner Basis theory may also be used to solve the kinematic problem of closed-chain robotic systems, as is the case of the works of Gan et al. (*27*) and Huang et al. (*28*), which employ it to solve the FKP of parallel robots, while Uchida et al. (*29*) use Groebner Bases to triangularize the kinematic constraint equations of this type of robots.

Based on those previous works, in (*8*) we presented a systematic procedure for the synthesis of the IKM of non-redundant open-chain robotic systems, that uses Groebner Basis theory to find an analytical solution for the IKP of this type of robots. The main objective of that work was to develop a systematic procedure that requires the least amount of input information from the user. This objective was mainly achieved, because the only required inputs for the developed procedure are the robot's Denavit-Hartenberg (D-H) parameters and the movement range of its actuators, but the user still have to select the monomial order of the employed Groebner Basis. This monomial order selection requires extra knowledge of the robot's structure, because the optimal monomial order is related to the correlation of its degrees of freedom (DoF).

All the previously mentioned works have this monomial order selection implied. Some of those works state that they employ the same monomial order as the one in which the variables are solved by the traditional methods (*8*), while most just present the optimal order for their robotic system, without mentioning how it was selected (*23*, *25–29*).

In order to achieve a fully systematic procedure, it is necessary to develop a method that automatically selects the optimal monomial order. The present work's main objective is to expand the procedure originally shown in (*8*), which uses Groebner Basis theory to synthesize the IKM of non-redundant open-chain robotic systems, with the required method for the automatic selection of the Groebner Basis' monomial order. Section 2 contains the definition of the monomial order in Groebner Bases and presents the different types of orders that can be used and their applications. The full procedure is explained in Section 4, placing special emphasis in the automatic selection of the monomial order. This updated procedure is used to calculate the IKM of two robots: a walking hexapod and Unimate's PUMA 560 manipulator, which are both shown in Section 3. Section 5 shows the procedure's performance analysis, proving that the selected monomial order is the optimal for each application, and also comparing the outputs of the synthesized IKMs with those of the reference models calculated by traditional methods. Finally Section 6 presents the conclusions of the obtained results and the final remarks of this work, while Section 7 shows the future work that arise from this research.

2. Monomial Ordering in Groebner Bases

The synthesis of an IKM using Groebner Basis theory begins with the Forward Kinematics of the analyzed robot, which is a multivariable trigonometric equation system. Through the application of the systematic procedure originally presented in (*8*), this trigonometric equation system is transformed into a multivariable polynomial equation system, which constitutes the generator set for an Ideal (*30*). It is from this Ideal that the Groebner Basis is calculated.

The main objective of the Groebner Basis computation is to obtain a new equation system, whose solution set is the same as that of the Ideal generator set. This new equation system, also called basis, is easier to solve than the original multivariable polynomial equation system and, if certain conditions are met, it has an analytical solution (*8*, *30*).

An important factor of the Groebner Basis obtained from any Ideal is the monomial ordering used in its calculation. This monomial ordering is defined in Definition 2.1 (*30*).

Definition 2.1. Monomial Ordering. A monomial ordering over a set of variables $x = [x_1, x_2, ..., x_n]$ on the field k, normally represented as $k[x_1, x_2, ..., x_n]$, is any relation $>$ on the set of monomials x^α, with $\alpha \in \mathbb{Z}^n_{\geq 0}$, that satisfies three conditions:

(1) The relation $>$ is a total ordering on $\mathbb{Z}^n_{\geq 0}$. This condition establishes that for every pair of monomials, x^α and x^β, with $\alpha = (\alpha_1, \alpha_2, ..., \alpha_n)$ and $\beta = (\beta_1, \beta_2, ..., \beta_n) \in \mathbb{Z}^n_{\geq 0}$, exactly one of the three following statements is true: $x^\alpha > x^\beta$, $x^\alpha = x^\beta$ or $x^\beta > x^\alpha$.
(2) If $x^\alpha > x^\beta$ and $\gamma = (\gamma_1, \gamma_2, ..., \gamma_n) \in \mathbb{Z}^n_{\geq 0}$, then $x^{(\alpha+\gamma)} > x^{(\beta+\gamma)}$.
(3) $>$ is a well-ordering on $\mathbb{Z}^n_{\geq 0}$. This means that every non-empty subset of $\mathbb{Z}^n_{\geq 0}$ has a smallest element under this ordering.

The main three types of monomial orderings used in Groebner Bases calculations are: Lexicographic Order (lex), Graded Lexicographic Order (grlex) and Graded Reverse Lexicographic Order (grevlex) (*30*).

The lexicographic order (lex) arranges the monomials following a strict ordering in which a variable always precedes those of lesser value in the order, in an analogous way to the ordering of words in a dictionary. This type of monomial ordering is defined in definition 2.2 (*30*).

Definition 2.2. Lexicographic Order (lex). The lex order establishes that $x^\alpha >_{lex} x^\beta$ if and only if, in the vector difference $\alpha - \beta \in \mathbb{Z}^n_{\geq 0}$, the leftmost non-zero entry is positive.

With the lex order, a variable will always dominate any monomial involving only smaller variables, regardless of its total degree. For some purposes, it is desirable to take into account the total degree of monomials, ordering them by total degree first. This is done by using the graded lexicographic order (grlex), presented in definition 2.3 (*30*).

Definition 2.3. Graded Lexicographic Order (grlex). The grlex order establishes that $x^\alpha >_{grlex} x^\beta$ if the condition of Equation (1) is met:

$$|\alpha| = \sum_{i=1}^{n} \alpha_i > |\beta| = \sum_{i=1}^{n} \beta_i \tag{1}$$

or, in the case that $|\alpha| = |\beta|$, if $x^\alpha >_{lex} x^\beta$, i.e. if the leftmost non-zero entry of $\alpha - \beta$ is positive.

To reduce the computation time of Groebner Bases, a variation of the grlex, known as the graded reverse lexicographic order (grevlex), was developed, which is defined in definition 2.4 (*30*).

Definition 2.4. Graded Reverse Lexicographic Order (grevlex). The grevlex order establishes that $x^\alpha >_{grevlex} x^\beta$ if the condition of Equation (1) is met or, in the case that $|\alpha| = |\beta|$, if the rightmost non-zero entry of the difference $\alpha - \beta$ is positive.

This variation begins like grlex, ordering the monomials by total degree, but then breaks any possible ties by applying a reverse lexicographical order. This ordering is the most efficient way to calculate a Groebner Basis for a zero-order Ideal, i.e. an Ideal that has a finite amount of solutions (*30*), which is exactly the type of Ideal that the developed procedure has to work with (*8*).

The main objective of our procedure, as presented in (*8*), is to use the Groebner Basis theory to find an analytical solution to the Inverse Kinematic Problem of non-redundant open-chain robotic systems, as a means to synthesize their IKM. This objective is achieved by calculating a Groebner Basis in a two step process: First, an initial basis is obtained using a grevlex monomial order, using Faugère's F4 algorithm (*31*, *32*). Then this first basis is converted to a Groebner Basis with a lex order, employing the FGLM algorithm developed by Faugère et al (*33*). This conversion is done because the lex monomial order is the only one that guarantees the existence of a simple analytical solution for the calculated basis.

Section 4 contains all the details about the updated procedure, including the selection of the optimal monomial order and the two-step Groebner Basis computation, while the next section presents the test platforms that were used in this work.

3. Test Platforms

The two robotic systems that were employed as test platforms for the developed procedure are a walking hexapod robot BH3-R, built and distributed by Lynxmotion Inc. (Swanton, Vermont, USA) and a Unimate's PUMA 560 robotic arm (Danbury, CT, USA). Both robots, presented in Figure 2, are non-redundant open-chain robotic systems.

The hexapod shown in Figure 2 has circular geometry, which means that its kinematic problem can be solved by synthesizing the IKM of one of the robot's legs, to later apply the corresponding transformations between the hexapod's center and the origin of each of its extremities. This robot was selected as a test platform because its extremities have three rotational Degrees of Freedoms (DoF) for positioning, like most industrial robotic arms and many non-redundant multi-legged walking robots, and it only requires the resolution of the positioning kinematic problem.

The D-H parameters of the hexapod's leg are shown in Table 1. With those D-

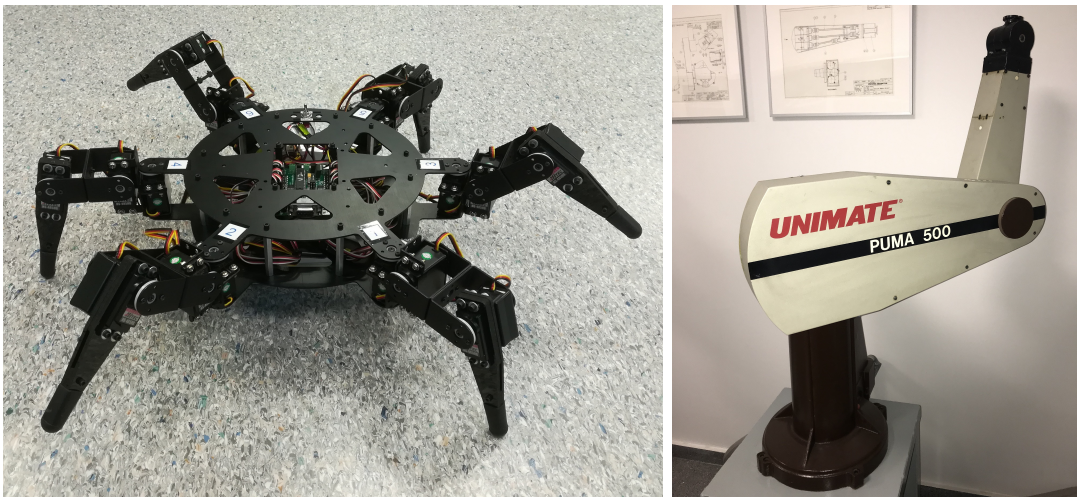

Figure 2. Robotic systems used in this work: BH3-R hexapod walking robot by Lynxmotion Inc. (**left**) and Unimate's PUMA 560 robotic arm (**right**).

Table 1. Denavit-Hartenberg parameters of the hexapod's leg.

Link	θ**[rad]**	d**[mm]**	a**[mm]**	α**[rad]**
1	q_1	0	28	$\pi/2$
2	q_2	0	58	π
3	$q_3 + (\pi/2)$	0	110	0

H parameters, and following all the steps of Denavit-Hartenberg's method (*1*), the homogeneous transformation between the origin of the hexapod's leg and its final point is presented in Equation (2).

$$ {}^{0}A_3 = \begin{bmatrix} \cos(q_1)\sin(q_2-q_3) & -\cos(q_1)\cos(q_2-q_3) & -\sin(q_1) & \cos(q_1)[28+58\cos(q_2)+110\sin(q_2-q_3)] \\ \sin(q_1)\sin(q_2-q_3) & -\sin(q_1)\cos(q_2-q_3) & \cos(q_1) & \sin(q_1)[28+58\cos(q_2)+110\sin(q_2-q_3)] \\ -\cos(q_2-q_3) & -\sin(q_2-q_3) & 0 & 58\sin(q_2)-110\cos(q_2-q_3) \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{2} $$

The second robot used as a test platform for the developed procedure is a PUMA 560 manipulator, whose D-H parameters are presented in Table 2. For simplicity reasons, in this work we will focus only on the PUMA's first three links, just before the robot's wrist. This simplification is valid because the last three links conform an in-line wrist (*3*, *9*). Therefore, the end effector of this robot will be considered to be at the anchor point of this in-line wrist, just in the base of the fourth link (*8*).

Table 2. Denavit-Hartenberg parameters of the PUMA 560 manipulator.

Link	θ**[rad]**	d**[mm]**	a**[mm]**	α**[rad]**
1	$q_1 + (\pi/2)$	660.4	0	$-\pi/2$
2	q_2	149.1	431.8	0
3	$q_3 + (\pi/2)$	0	-20.3	$\pi/2$
4	q_4	433.1	0	$-\pi/2$
5	q_5	0	0	$\pi/2$
6	q_6	56.2	0	0

Applying the Denavit-Hartenberg's method, the homogeneous transformation between the base of the PUMA and the anchor point of its in-line wrist is the matrix shown in Equation (3). The term "q_{23}" is equal to "$q_2 + q_3$".

$$ {}^{0}A_{4} = \begin{bmatrix} \sin(q_1)\sin(q_{23}) & \sin(q_1)\cos(q_{23}) & -\cos(q_1) & -\sin(q_1)[a_2\cos(q_2)+d_4\cos(q_{23})+a_3\sin(q_{23})]-d_2\cos(q_1) \\ -\cos(q_1)\sin(q_{23}) & -\cos(q_1)\cos(q_{23}) & -\sin(q_1) & \cos(q_1)[a_2\cos(q_2)+d_4\cos(q_{23})+a_3\sin(q_{23})]-d_2\sin(q_1) \\ -\cos(q_{23}) & \sin(q_{23}) & 0 & d_1-a_2\sin(q_2)-d_4\sin(q_{23})+a_3\cos(q_{23}) \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (3) $$

4. Procedure to Synthesize the IKM using Groebner Basis

The first version of the procedure developed to synthesize the IKM of non-redundant open-chain robotic systems is presented in (*8*). This second version includes the automatic selection of the optimal monomial order for the Groebner Basis, and is now divided in the following six steps:

(1) Information input
(2) Forward kinematics
(3) Obtention of the polynomial equation system
(4) Groebner Bases computation
(5) Automatic monomial order selection
(6) Final IKM algorithm

The first three steps of this second version are exactly the same as in the first one (*8*), so in this work we are just going to do a quick review of them. The main change in this second version is in the fourth and fifth steps, where the Groebner Bases are computed and the optimal monomial order is selected.

As in the first version, all these steps are only executed once, out of line, to synthesize the IKM before the robotic system is activated. So it is important to bear in mind that the execution time of these steps does not affect in any way the online computation time of the final IKM (*8*).

In the first step the user has to supply only two data sets: the Denavit-Hartenberg (D-H) parameters of the robot and the movement range of its actuators. The D-H parameters are employed to solve the Forward Kinematics problem, while the actuators' movement range is used to eliminate all the solutions that are not reachable by the robotic system (*8*).

This is all the information that the user has to know about the analyzed robot, which constitutes the absolute minimum data that is required to describe a robotic system. From this point, the procedure automatically synthesizes the robot's IKM, without any further input from the user.

The second step of the procedure is to calculate the robot's Forward Kinematics, using as inputs the D-H parameters that were provided in the first step. This is done by applying the Denavit-Hartenberg method (*1*, *8*). When the developed procedure is applied to the hexapod's leg, this step's output is the same homogeneous transformation matrix presented in Equation (2).

The procedure's third step begins by extracting the equations relating the pose of the end effector with the robot's state, from the homogeneous transformation matrix obtained in the second step. For simplicity, in this work we are only interested in the position of the end effector. However, the procedure can be extended to use the end effector's orientation, to complete the whole pose (*8*).

Continuing with the case of the hexapod's leg, the extracted position equations are

the ones presented in Equation (4), which correspond to the first three elements of the fourth column of Equation (2).

$$\begin{aligned} p_x &= \cos(q_1)[28+58\cos(q_2)+110\sin(q_2-q_3)] \\ p_y &= \sin(q_1)[28+58\cos(q_2)+110\sin(q_2-q_3)] \\ p_z &= 58\sin(q_2)-110\cos(q_2-q_3) \end{aligned} \tag{4}$$

Equation (4) conforms a trigonometric equation system that describes the end effector's position as a function of the current state of the robot's actuators. The previous equation system is completed with trigonometric identities of the form $\sin(q_i)^2 + \cos(q_i)^2 = 1$ for all the robot's rotational degrees of freedom (DoFs). Lastly, all the trigonometric expressions are expanded, and variable substitutions of the form $\sin(q_i) = s_i$ and $\cos(q_i) = c_i$ are applied.

The third step's final output is a polynomial equation system where the variables are either a pair sine-cosine of a rotational DoF (s_i and c_i), or directly the position value of a prismatic DoF (q_j).

For the hexapod's leg, this step's output is the polynomial equation system presented in Equation (5), where the input parameters are the three components of the position vector of the leg's end point ($\vec{\mathbf{p}}$), represented by p_x, p_y, and p_z. The six variables of Equation (5) that should be solved are s_1, c_1, s_2, c_2, s_3, and c_3.

$$\begin{aligned} 28c_1 + 58c_1c_2 + 110c_1s_2c_3 - 110c_1c_2s_3 - p_x = 0 \\ 28s_1 + 58s_1c_2 + 110s_1s_2c_3 - 110s_1c_2s_3 - p_y = 0 \\ 58s_2 - 110c_2c_3 - 110s_2s_3 - p_z = 0 \\ s_1^2 + c_1^2 - 1 = 0 \\ s_2^2 + c_2^2 - 1 = 0 \\ s_3^2 + c_3^2 - 1 = 0 \end{aligned} \tag{5}$$

The equation system obtained as the third step's output is the Ideal generator set (*30*). From this point begin the steps that differ from the procedure previously published in (*8*).

4.1. *Groebner Bases computation*

In the work presented in (*8*), the user of the developed procedure had to choose a lex order before the computation of the Groebner Basis. But this selection implied having extra information about the analyzed robotic system, besides the one already provided in the first step. In this second version we aim to remove this burden from the user. To do so, the procedure will automatically choose the optimal lex order for the Groebner Basis.

It is important to highlight that all the possible lex order combinations will produce an array of Groebner Bases, whose solutions are all the same solution set of the Ideal. Therefore, solving any of those Groebner Bases will also solve the polynomial equation system obtained in the third step (*30*). The only difference between obtaining the solution from the Groebner Basis generated by one specific lex order over another one, would be on the computation times required to solve each basis. So the optimal

monomial order, that the developed procedure automatically selects, is the one that requires the least computation time to find its solution.

The main objective of the procedure's fourth step is to establish the lex orders that are relevant to the analyzed robot, and calculate their associated Groebner Bases. The analysis of those bases' computations times will be done in the fifth step, along with the selection of the optimal monomial order.

The first task of this fourth step is to recognize all the possible lex order combinations that can be relevant for the analyzed system. The variables that should be solved in the polynomial equation system are the ones related with the rotational DoFs, that come in pairs of the form s_i and c_i, and the variables of the prismatic DoFs. For the case of the hexapod's leg, shown in Equation (5), the monomials of the Ideal are the variables s_1, c_1, s_2, c_2, s_3, and c_3. The optimal lex order for the Ideal generated by the set presented in Equation (5) must be a combination of those variables.

All the possible combinations of the Ideal monomials will be equal to $(n+r)!$, where n is the total amount of DoFs of the analyzed robot, and r is the quantity of those DoFs that are rotational. But all those combinations include the orders that separate the two trigonometric variables of a rotational DoF (s_i and c_i). These combinations are not relevant lex orders, because the two variables of a rotational DoF are strongly related, and should always be adjacent in the lex order. This restriction greatly reduces the amount of possible lex orders, but it can be further reduced if a relative order is established between the two variables of each rotational DoF (either $s_i > c_i$ or $c_i > s_i$). This way the total amount of relevant lex orders gets reduced to $n!$, independently if the robot's DoFs are rotational or prismatic.

To impose the aforementioned restriction, the developed procedure analyzes which of the trigonometric variables of every rotational DoF is less likely to be zero. This is done because, when solving the Groebner Basis, if the lex order is of the form "... $> c_i > s_i >$ ...", the value of s_i ($sin(q_i)$) will be calculated first, while the computation of c_i will surely depend on the calculated value of s_i. Therefore, if s_i is equal to zero, it is very probable that a term of the basis' equation that defines c_i will be canceled out, which will surely generate an indetermination in the computation of c_i.

So, by analyzing the probable values of $sin(q_i)$ and $cos(q_i)$ in the range of the rotational DoF (q_i), the procedure detects the trigonometric variable that is less likely to be zero, and orders these two variables properly inside the lex order. This analysis is done by calculating the expected value of $|sin(q_i)|$ and $|cos(q_i)|$, following the expressions shown in Equation (6).

$$\begin{aligned} E[|sin(q_i)|] &= \int_{q_{i_{lo}}}^{q_{i_{up}}} |sin(q_i)| \cdot p(q_i) \cdot dq_i \\ E[|cos(q_i)|] &= \int_{q_{i_{lo}}}^{q_{i_{up}}} |cos(q_i)| \cdot p(q_i) \cdot dq_i \end{aligned} \tag{6}$$

In Equation (6), $q_{i_{lo}}$ and $q_{i_{up}}$ are the lower and upper limits of the range of the actuator q_i, and $p(q_i)$ is the probability density function (pdf) of q_i. This pdf may be supplied by the user, if the probable distribution of the positions of q_i in all its range is known. Otherwise, the developed procedure assumes a Gaussian pdf, that covers all the rotational DoF's range, with its mean in the middle of this range. If $E[|sin(q_i)|] > E[|cos(q_i)|]$, then the relative order of the couple of variables related to q_i is selected as $c_i > s_i$. Otherwise, this relative order is established as $s_i > c_i$. This

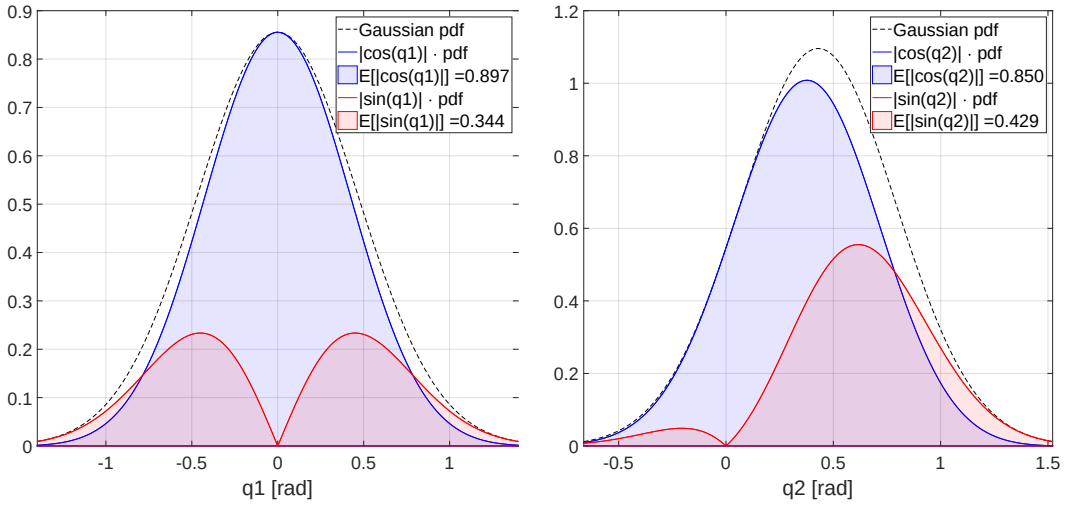

Figure 3. Graphical representation of the expected values for the variables related with two DoFs of the hexapod's leg, q_1 and q_2. The black dotted line show the chosen pdf for each DoF. The blue areas represent the $E[|cos(q_i)|]$, while the red areas are equal to $E[|sin(q_i)|]$. In both cases $E[|cos(q_i)|] > E[|sin(q_i)|]$, so the variables related with these rotational DoFs are ordered as $s_1 > c_1$ and $s_2 > c_2$.

Table 3. Expected values for the trigonometric variables related with the hexapod's rotational DoFs. The columns titled "$\mathbf{E[|cos(q_i)|]}$" and "$\mathbf{E[|sin(q_i)|]}$" contain the expected values of the corresponding trigonometric variable for each rotational DoF. The largest expected value of each DoF is marked in **bold**, and the selected order for each rotational DoF is shown in the last column.

Rotational DoF	$\mathbf{E[\lvert cos(q_i)\rvert]}$	$\mathbf{E[\lvert sin(q_i)\rvert]}$	Relative order
q_1	**0.897**	0.344	$s_1 > c_1$
q_2	**0.850**	0.429	$s_2 > c_2$
q_3	**0.831**	0.460	$s_3 > c_3$

way the variable that is less likely to be zero is always computed first.

Figure 3 shows the graphical representation of this order selection process when applied to the actuators q_1 and q_2 of the hexapod's leg. The black dots outline the Gaussian pdf chosen for each of these robot's actuators. The blue areas represent $E[|cos(q_i)|]$, while the red areas equate $E[|sin(q_i)|]$. In both cases $E[|cos(q_i)|] > E[|sin(q_i)|]$, so the selected orders for these couples are $s_1 > c_1$ and $s_2 > c_2$, and they will always appear that way in all the possible lex orders for the hexapod's leg.

Table 3 presents the expected values for each of the three rotational DoFs that compose the hexapod's leg. This table also shows the relative order of the trigonometric variables related to each of these DoFs.

After establishing all the relevant lex orders, the developed procedure proceeds to calculate the Groebner Bases related to those orders. These calculations are executed in a two step process: First, an initial Groebner Basis is obtained using Faugère's F4 algorithm (*31*, *32*), with a graded reverse lexicographic order (grevlex). After this first basis is calculated, a series of FGLM basis conversions, based on the algorithm developed by Faugère et al (*33*), are made to convert the original basis to all the lex orders previously identified as relevant. This way a set of $n!$ different bases is calculated, one for each of the relevant lex orders. The procedure's next step is the selection of the optimal monomial order.

Table 4. Relevant lex orders for the hexapod's leg.

Nº	Lexicographic Order
1	$s_1 > c_1 > s_2 > c_2 > s_3 > c_3$
2	$s_1 > c_1 > s_3 > c_3 > s_2 > c_2$
3	$s_2 > c_2 > s_1 > c_1 > s_3 > c_3$
4	$s_2 > c_2 > s_3 > c_3 > s_1 > c_1$
5	$s_3 > c_3 > s_1 > c_1 > s_2 > c_2$
6	$s_3 > c_3 > s_2 > c_2 > s_1 > c_1$

Table 5. Possible types of polynomial equations found in the calculated Groebner Bases

Type	Degree	Polynomial equation
Linear	1	$a_1x + a_0 = 0$
Quadratic	2	$a_2x^2 + a_1x + a_0 = 0$
Bi-Quadratic	4	$a_4x^4 + a_2x^2 + a_0 = 0$
Quartic	4	$a_4x^4 + a_3x^3 + a_2x^2 + a_1x + a_0 = 0$

To continue with the hexapod's leg example, Table 4 contains the relevant lex orders for this robotic system. The relevant amount of lex orders is six, because the hexapod's leg contains three DoFs ($3! = 6$).

4.2. *Automatic monomial order selection*

Once all the Groebner Bases are calculated, the procedure automatically identifies the basis related with the optimal lex order. As was stated in Section 4.1, the optimal monomial order is the one that yields the Groebner Basis that requires the lower amount of computation time to solve. So, in order to find this optimal monomial order, the following three-step classification criterion is applied:

(1) Lowest computational cost of the highest degree equation
(2) Lowest accumulated computational cost of all the basis' equations
(3) Lowest accumulated cost of all the equations' coefficients

The first step of the classification process consists on identifying the Groebner Bases whose highest degree equation presents the lesser computational time, and discard the rest.

Table 5 presents all the possible types of polynomial equations that can be found in the previously calculated Groebner Bases. The maximum possible degree of the polynomial equations shown in Table 5 is four, because the IKP of the positioning of non-redundant open-chain robotic systems has at most four solutions (*8*).

The linear and quadratic equations that our procedure encounters are solved by the well known algorithms for these types of polynomials. The bi-quadratic polynomial equations are solved as two concatenated quadratic equations. Finally, the quartic equations that the developed procedure may encounter are solved using Salzer's algorithm (*34*). It is important to highlight that it is not necessary to prepare for the appearance of cubic equations, because the solutions of the IKP of open-chain robotic systems always come in pairs (*3*).

A list of all the operations required to solve the polynomial equations previously shown is compiled in Table 6. The last column of Table 6 contains the computational cost, in clock cycles, of every type of polynomial equation, assuming that the micro-

Table 6. Computational cost required to solve different types of polynomial equations on an ARM Cortex-M4 CPU (*35*, *36*).

Type		Operations					Cost [Cycles]
		adml	div	sqrt	trig	atan	
Linear		1	1	0	0	0	15
Quadratic		7	2	1	0	0	49
Bi-Quadratic		9	2	3	0	0	79
Quartic	min	68	4	3	0	0	166
	max	80	5	5	1	1	282

Table 7. Computational cost, in clock cycles, for a microcontroller with an ARM Cortex-M4 CPU (*35*, *36*).

Operation	Identifier	Cost [Cycles]
Addition or Multiplication	adml	1
Division	div	14
Square root	sqrt	14
Trigonometric operation	trig	29
Arctangent (atan2)	atan	33

controller in charge of the robot's control has an ARM Cortex-M4 CPU (*35*) (*36*). Table 7 breaks down the cost in clock cycles of each operation for the selected microcontroller. If the robot has a control system with a different type of CPU, the user can change the cost of every operation in the initial settings of the procedure, in order to reflect the real computation times on the robot's CPU.

In Table 6, the quartic polynomial equation encompasses two rows, identified as "min" and "max", because Salzer's algorithm can take different paths, depending on the type of roots of the analyzed polynomial equation (*34*). In this case the computational cost is the average between the minimum cost for the quartic equation and its maximum, which is equal to 224 clock cycles for an ARM Cortex-M4 CPU.

Using the costs shown in the last column of Table 6, the procedure selects the Groebner Bases whose highest degree equation has the lower computational cost, thus quickly discarding all the bases that would require a larger amount of time and resources to be solved. It is possible that more than one basis passes this first selection criterion, so the second criterion is applied to all these remaining bases.

The second classification criterion consists on selecting the bases with the lower accumulated computational cost from all their equations. The accumulated cost of each basis is equal to the sum of the computational cost of all the polynomial equations that compose that basis, using again Table 6 to evaluate these costs. This second criterion is only passed by those bases with the lowest accumulated computational cost.

Once again, more than one basis can progress beyond the second criterion, in which case the third one is applied. This third classification criterion searches for the basis with the lowest accumulated cost from all its equation's coefficients. This last accumulated cost is quantified by adding all the clock cycles required to calculate all the coefficients in all the basis' equations. Therefore, after previously discarding all the bases whose equation types would require larger amounts of time and resources to be solved, the procedure narrows down the list of Groebner Bases to those which effectively have the least computation time.

If after these three classification criteria, there is still more than one candidate for the optimal lex order, then any of the orders related with these remaining bases may be selected, because all the bases that pass the three criteria will surely present similar computation times.

Table 8. Selection of the lex order for the Hexapod's IKM. After each step of the selection process, the discarded orders are marked in *italic*. A value of "-" indicates that the lex order was discarded in a previous step. The first classification criterion is to search for those lex orders whose basis' highest degree equation present the least computation time. The second one seeks for the bases with the lesser accumulated cost of all their equations, while the last one searches for the lowest amount of time required to compute all the basis' coefficients. All these costs, expressed in clock cycles, were calculated for a microcontroller with an ARM Cortex-M4 CPU (*35*, *36*). The selected lex order is marked in **bold**.

	Classification Criteria		
Order N°	**Highest Deg. [Cycles]**	**Acc. Cost [Cycles]**	**Coefficients [Cycles]**
1	*79*	-	-
2	*224*	-	-
3	*79*	-	-
4	**49**	**158**	**103**
5	*224*	-	-
6	49	158	*201*

Continuing with the case of the hexapod's leg, Table 8 compiles the results obtained when applying the presented three-step classification criterion to the set of six Groebner Bases calculated in the fourth step of the procedure. The lex orders 1, 2, 3 and 5 were discarded after the first step of the selection process, leaving only orders 4 and 6 active. Both remaining orders passed successfully the second step, but in the third one, lex order 6 was eliminated, setting lex order 4 as the chosen one.

As it can be seen in Table 8, lex order 4 is the selected monomial order of the final Groebner Basis employed for the synthesis of the IKM of the hexapod's leg. The output of this fifth step is the Groebner Basis related to the selected lex order, which for the case of the hexapod's leg is the set of polynomial equations presented in Equations (7) to (12):

$$-p_x^2+[p_x^2+p_y^2]c_1^2=0 \tag{7}$$

$$-p_y c_1+p_x s_1=0 \tag{8}$$

$$\begin{aligned} p_x^5+p_x p_y^4+p_x p_z^4-29360p_x p_z^2-26224(p_x^3+p_x p_y^2)+\\ +52684800p_x+2(p_x^3 p_y^2+p_x^3 p_z^2+p_x p_y^2 p_z^2)+\\ +[1644160(p_x^2+p_y^2)-112(p_x^4+p_y^4+p_x^2 p_z^2+p_y^2 p_z^2)-224p_x^2 p_y^2]c_1+\\ +162817600p_x c_3^2=0 \end{aligned} \tag{9}$$

$$p_x^3+p_x p_y^2+p_x p_z^2-14680p_x-56[p_x^2+p_y^2]c_1+12760p_x s_3=0 \tag{10}$$

$$
\begin{aligned}
&28(p_x^5+p_xp_y^4+p_xp_z^4)+56(p_x^3p_y^2+p_x^3p_z^2+p_xp_y^2p_z^2)+\\
&+200704(p_x^3+p_xp_y^2-p_xp_z^2)-174562304p_x+\\
&+[10304(p_x^4+p_y^4)+20608p_x^2p_y^2-p_x^6-p_y^6-4p_x^2p_y^2p_z^2-3(p_x^4p_y^2+p_x^2p_y^4)+\\
&+7168(p_x^2p_z^2+p_y^2p_z^2)-p_x^2p_z^4-p_y^2p_z^4-2(p_x^4p_z^2+p_y^4p_z^2)-7463680(p_x^2+p_y^2)]c_1+\\
&+[12760(p_x^3p_z+p_xp_z^3+p_xp_y^2p_z)+10003840p_xp_z]c_3+714560[p_x^2p_z+p_y^2p_z]c_1c_3+\\
&+[116(p_x^5+p_xp_y^4+p_xp_z^4)+232(p_x^3p_y^2+p_x^3p_z^2+p_xp_y^2p_z^2)+\\
&+181888(p_xp_z^2-p_x^3-p_xp_y^2)+71300096p_x]c_2=0
\end{aligned}
\tag{11}
$$

$$
\begin{aligned}
&10304(p_x^3p_z+p_xp_y^2p_z)-p_x^5p_z-p_xp_z^5-p_xp_y^4p_z+6234368p_xp_z+\\
&+7168p_xp_z^3-2(p_x^3p_z^3+p_x^3p_y^2p_z+p_xp_y^2p_z^3)+489216[p_x^2p_z+p_y^2p_z]c_1+\\
&+[357280(p_xp_z^2-p_x^3-p_xp_y^2)+280107520p_x]c_3+\\
&+[10003840(p_x^2+p_y^2)-12760(p_x^4+p_y^4+p_x^2p_z^2+p_y^2p_z^2)-25520p_x^2p_y^2]c_1c_3+\\
&+[116(p_x^5+p_xp_y^4+p_xp_z^4)+232(p_x^3p_y^2+p_x^3p_z^2+p_xp_y^2p_z^2)+\\
&+181888(p_xp_z^2-p_x^3-p_xp_y^2)+71300096p_x]s_2=0
\end{aligned}
\tag{12}
$$

As can be seen in the previous equations, the obtained Groebner Basis constitutes a triangular equation system that can be solved in a staggered way (*8*, *29*). This is because the first equation of the system, shown in Equation (7), only depends on one variable, the lesser monomial in the selected lex order. After that equation is solved, the second one has at most two variables, the one that was previously calculated in the first equation and a new one, while the third depends at most on three variables, two computed in the preceding equations and a new one, and so on. Solving this triangular equation system gives the solution for the all the variables of the original polynomial equation system (*8*).

4.3. *Final IKM Algorithm*

The sixth step of this second version of the developed procedure is again similar to the last one of the procedure presented in (*8*). The main objective of this last step is to transform the triangular polynomial equation system of the selected lex order's basis in an algorithm that implements the final IKM.

As was stated before, the solution of the synthesized Groebner Basis will be a set composed either by the variables of prismatic DoFs (q_j), pairs sine-cosine of rotational DoFs (s_i and c_i), or a combination of both. Therefore, to finish the IKM synthesis of the analyzed robot, it is only necessary to compute the final value of the rotational DoFs, using the corresponding expression of the form presented in Equation (13):

$$
q_i = \operatorname{atan2}(s_i, c_i) \tag{13}
$$

The output of this last step is the algorithm of the synthesized IKM, implemented in two different languages: C++, as compiled code ready to be used in the robot's microcontroller, and MATLAB® script (R2017b, MathWorks®, Natick, MA, USA). This IKM may be used directly in the robot's control system, as the generator of the references for the multi-axis control (see Figure 1), or to simulate the robot's behavior.

The developed procedure can be used to synthesize the IKM of a big range of open-chain robots, including all the cartesian robotic systems, SCARA robots, all the non-redundant robotic manipulators that satisfy the in-line wrist condition, and all non-redundant open-chain robots whose IKM should only solve the positioning problem, as is the case of most multi-legged walking robots (*8*).

The performance analyses of the IKM synthesized for the hexapod's leg, as well as the one prepared for the PUMA 560, are presented in Section 5.

5. Performance Analysis

The procedure presented in Section 4 was used to synthesize the IKMs of a leg of the BH3-R walking hexapod and the PUMA 560 manipulator, both presented in Figure 2. As was previously explained, the inputs of the developed procedure are the D-H parameters of the analyzed robot and the movement range of its actuators, while the output is the synthesized IKM, both in C++ and in MATLAB® script. This procedure also selects automatically the optimal monomial order of the final Groebner Basis that will constitute the IKM, so the user does not need to have any further information about the robot.

The performance of the synthesized IKMs were simulated in MATLAB®, testing their ability to solve the IKP in the workspace of their corresponding robotic systems. The solutions obtained by these IKMs were compared with their corresponding reference models, which are the kinematic models calculated by traditional methods. In (*8*) we show the synthesis of the reference models of the PUMA 560 and the hexapod's leg.

5.1. *Hexapod's IKM*

Throughout this work we have been using the hexapod's leg as an example for the application of the developed procedure, giving special attention to the automatic selection of the basis' monomial order. In this section we will prove that the selected monomial order is effectively the optimal one. This will be done by analyzing the performance and the computation times of all the Groebner Bases calculated in 4.1, in order to demonstrate that the IKM synthesized from this selected order presents the lesser error and execution time.

Figure 4 presents the comparison between the outputs of the hexapod's leg reference model and the ones obtained by all six IKMs synthesized for this robot, one for each of the relevant lex orders presented in Table 4. This figure has marked in green ("Correctly calculated") all those positions inside the hexapod leg's workspace in which the corresponding IKM obtains the same amount of solutions as the reference model and, for all those solutions, the root mean square error (RMS) in the configuration space is less than $1{\times}10^{-6}$. Marked by red circles are the singularities found by the IKMs which, for the case of the hexapod's leg, are located in the axis defined by the intersection of the planes $X = 0 \cap Y = 0$.

Figure 4 shows that lex orders 1 to 5 correctly calculate all the positions of the robot's workspace, including its singularities. Only lex order 6 has some problems, because it incorrectly computes some false singularities, marked by black circles ("False Singularities"), along the axis defined by the intersection of the planes $Y = 0 \cap Z = 0$. This happens because the basis generated by lex order 6 has several leading terms that heavily depend on the values of p_y and p_z, the position of the leg's tip along

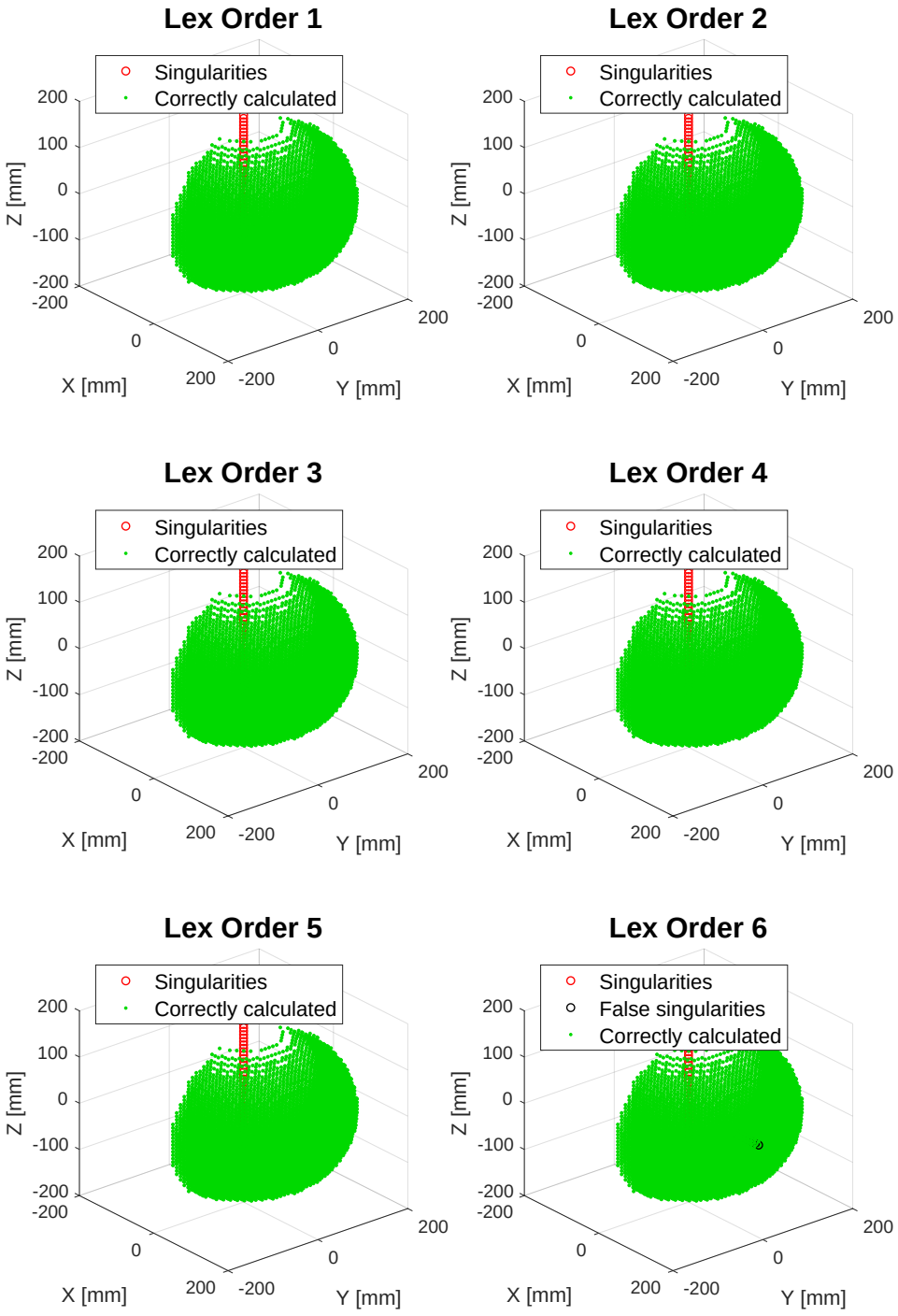

Figure 4. Performance analysis of the six synthesized IKMs for the hexapod's leg, one for each relevant lex order. Marked in green ("Correctly calculated") are all the the positions in which the corresponding IKM obtains the same amount of solutions as the reference model, with an RMS error lesser than 1×10^{-6}. The red circles correspond to the singularities of the leg's mechanism that are correctly identified by each IKM. The data show that lex orders 1 to 5 correctly calculate all the positions of the analyzed workspace, including the singularities of the leg's structure. Only lex order 6 has some problems, because it incorrectly computes some false singularities.

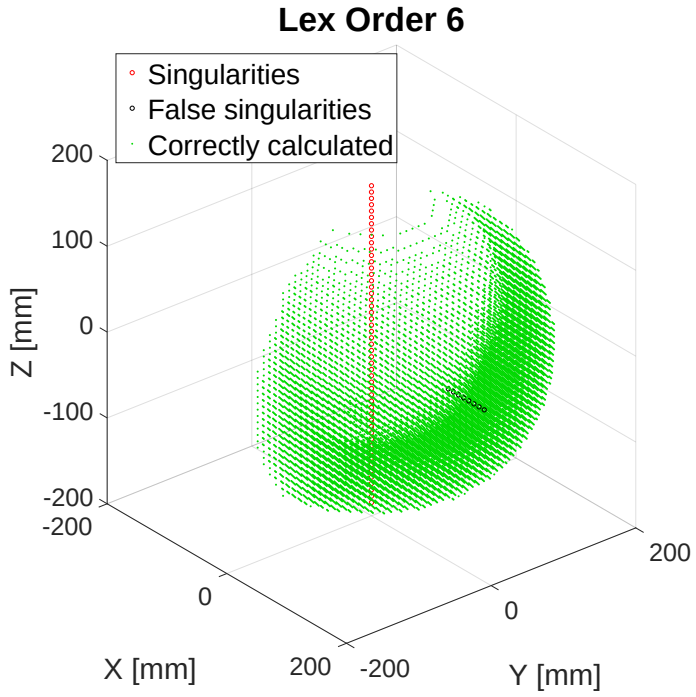

Figure 5. Detailed view of the performance of the hexapod's IKM generated by lex order 6. The difference between this IKM and the ones generated by lex orders 1 to 5, is that the other IKMs do not present those false singularities along the axis defined by the intersection of the planes $Y = 0 \cap Z = 0$.

the axis Y_0 and Z_0, respectively. Therefore, when p_y and p_z are both equal to zero, those leading terms are canceled out, generating an indetermination when the system is solved. Figure 5 presents a detailed view of the IKM generated by this problematic lex order.

These false singularities from the IKM generated by lex order 6 are not a problem, because the selected monomial order is lex order 4, which synthesizes an IKM that correctly calculates all the positions of the robot's workspace. But in the case that the selected monomial order has a problem similar to the one of lex order 6, the procedure's user can indicate that the problematic order is not a valid one. In this case the developed procedure will offer a new IKM, generated by the best monomial order from a list of lex orders that does not contain the problematic one.

In Figures 4 and 5, all the positions marked as "Correctly calculated" have an RMS error lesser than 1×10^{-6} when compared to the ones calculated by the hexapod's reference model. Table 9 contains the maximum and average RMS errors obtained when all the points of the hexapod's workspace are processed by each of the six IKMs synthesized for this robot. As it can be seen in this table, the IKM that presents the least RMS error, both in its maximum value and its average, is the one generated by the selected lex order: number 4.

Regarding the computation cost of the six synthesized IKMs for the hexapod's leg, Table 10 contains a summary of the computation times when those IKMs are presented with all points in the robot's workspace, and compares those times with the ones of the reference model (row named "Ref."). This table shows the maximum ("Max [ms]"), minimum ("Min [ms]") and average ("Avg [ms]") times required to compute all the

Table 9. Maximum and average RMS errors obtained when all the points of the hexapod's workspace are processed by each of the six synthesized IKMs. The selected lex order is marked in **bold**.

Nº	**Avg. RMS Error**	**Max. RMS Error**
1	5.566×10^{-12}	2.981×10^{-8}
2	2.577×10^{-13}	5.345×10^{-10}
3	5.754×10^{-12}	2.981×10^{-8}
4	$\mathbf{4.542\times10^{-16}}$	$\mathbf{1.243\times10^{-14}}$
5	2.561×10^{-13}	5.325×10^{-10}
6	1.177×10^{-14}	1.175×10^{-12}

Table 10. Computation times of the six IKMs generated for the hexapod's leg and its reference model ("Ref." row), when they are presented with all points in the robot's workspace. Column "Avg [ms]" shows the average computation times obtained for all the workspace's points (in milliseconds), while "Min [ms]" and "Max [ms]" present the minimum and maximum registered times, respectively. The selected lex order is marked in **bold**.

Nº	**Min [ms]**	**Avg [ms]**	**Max [ms]**
1	0.226	0.265	0.726
2	0.236	0.285	1.031
3	0.210	0.251	0.748
4	**0.199**	**0.238**	**0.670**
5	0.231	0.281	1.049
6	0.203	0.247	0.787
Ref.	0.033	0.126	0.347

different positions inside the workspace. In this table it can be seen that, as expected, the smallest computation time among all the synthesized IKMs is the one achieved by the IKM related to the selected lex order. This proves that the three-step classification criterion presented in Section 4.2 effectively identifies the optimal monomial order.

It is important to highlight that it is impossible to achieve computation times that are lower than the ones of the reference model calculated by traditional methods, because this model is already composed of equations specially crafted for the analyzed robot. The computation times shown for the selected IKM are close enough to the ones of the reference model, while achieving a negligible positioning error, so it can be concluded that the IKM synthesized by our procedure is equivalent to this reference model.

The performance of the final IKM, synthesized with the selected lex order, was also tested following a trajectory that covers three consecutive full swings of the hexapod's leg, each one with a duration of 2.6s. The first swing has the hexapod walking in a diagonal direction that is $-\pi/6$ rad measured from the forward direction. The second swing is walking in the forward direction, while the third one has the robot moving in a $\pi/6$ rad diagonal. Figure 6 presents these tests results, where it can be seen that the outputs given by our IKM allow the hexapod's leg to follow any trajectory, with high precision and a completely negligible positioning error.

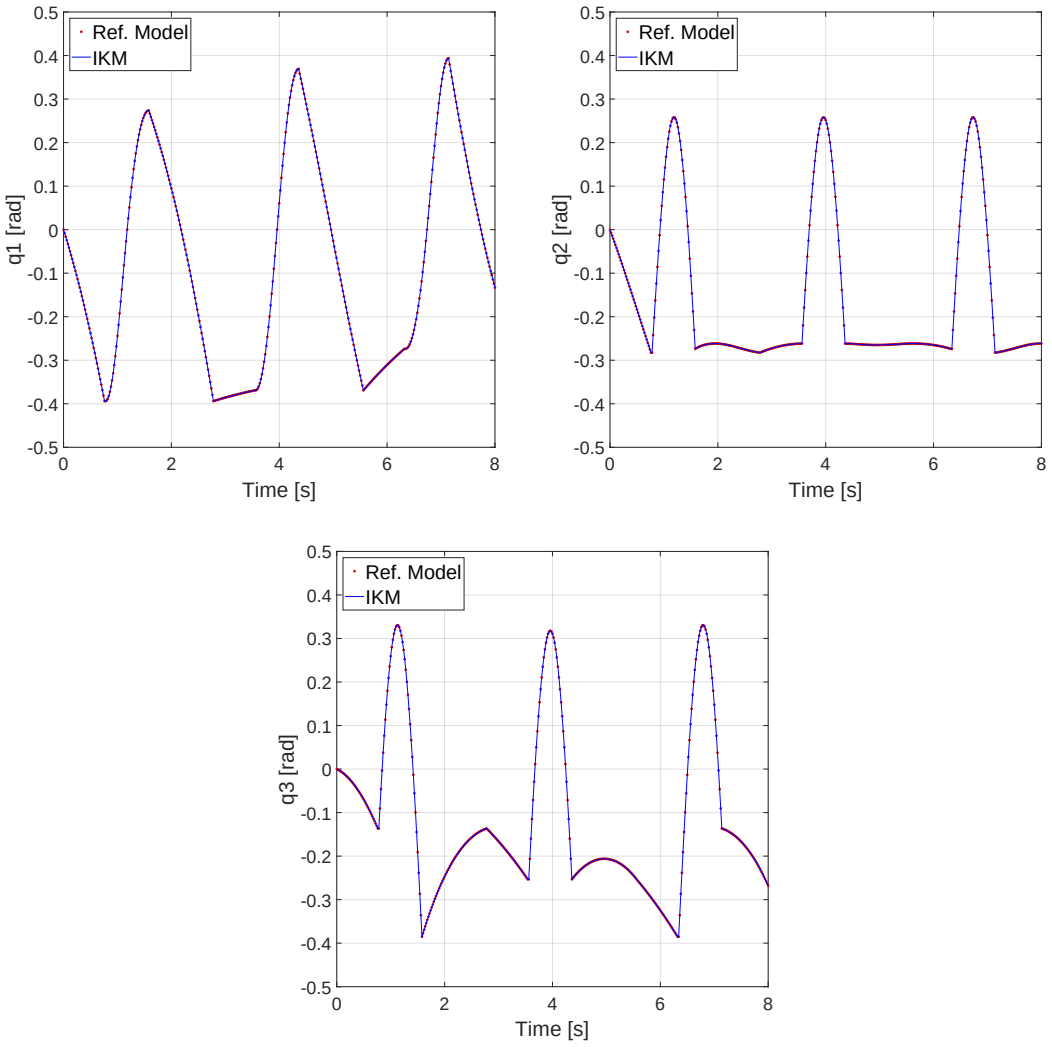

Figure 6. Trajectory tracking analysis for the hexapod's final IKM. The red dots represent the outputs of the reference model, for each of the leg's DoFs, when it is supplied with the predetermined trajectory, while the continuous blue lines are the IKM's outputs for that same trajectory. The top left graph contains the computed outputs for the first DoF (q_1). The data of the second DoF (q_2) is presented in the top right graph, while the bottom one displays the third DoF's outputs (q_3). The data show that the IKM's outputs follow those of the reference model for all three DoFs, with high accuracy and a negligible error along all the desired trajectory

Table 11. Expected values for the trigonometric variables related with the PUMA's rotational DoFs. The columns titled "$\mathbf{E}[|\mathbf{cos}(\mathbf{q_i})|]$" and "$\mathbf{E}[|\mathbf{sin}(\mathbf{q_i})|]$" are the expected values of the trigonometric variables related with a rotational DoF. The relative order of the trigonometric variables related with the rotational DoF is established depending on those expected values. The largest expected value of each DoF is marked in **bold**.

Rotational DoF	$\mathbf{E}[\|\mathbf{cos}(\mathbf{q_i})\|]$	$\mathbf{E}[\|\mathbf{sin}(\mathbf{q_i})\|]$	Relative order
q_1	**0.709**	0.561	$s_1 > c_1$
q_2	0.507	**0.763**	$c_2 > s_2$
q_3	**0.757**	0.511	$s_3 > c_3$

Table 12. Relevant lex orders for the PUMA manipulator.

Nº	Lexicographic Order
1	$s_1 > c_1 > c_2 > s_2 > s_3 > c_3$
2	$s_1 > c_1 > s_3 > c_3 > c_2 > s_2$
3	$c_2 > s_2 > s_1 > c_1 > s_3 > c_3$
4	$c_2 > s_2 > s_3 > c_3 > s_1 > c_1$
5	$s_3 > c_3 > s_1 > c_1 > c_2 > s_2$
6	$s_3 > c_3 > c_2 > s_2 > s_1 > c_1$

5.2. *PUMA's IKM*

To demonstrate the application of the developed procedure to robotic manipulators, it was also used to synthesize the IKM of the PUMA 560 shown in Figure 2. As it was said before, the user just has to give as inputs the D–H parameters of the PUMA, presented in Table 2, and the movement range of its actuators.

When applied to the PUMA robot, the output of the procedure's third step is the polynomial equation system presented in Equation (14), where the input parameters are the three components of the position vector of the manipulator's wrist ($\vec{\mathbf{p}}$), represented by p_x, p_y, and p_z. As was also the case for the hexapod's leg, the six variables of this polynomial equation system that should be solved are s_1, c_1, s_2, c_2, s_3, and c_3.

$$\begin{gathered} -a_2 s_1 c_2 - d_4 s_1 c_2 c_3 + d_4 s_1 s_2 s_3 - a_3 s_1 s_2 c_3 - a_3 s_1 c_2 s_3 - d_2 c_1 - p_x = 0 \\ a_2 c_1 c_2 + d_4 c_1 c_2 c_3 - d_4 c_1 s_2 s_3 + a_3 c_1 s_2 c_3 + a_3 c_1 c_2 s_3 - d_2 s_1 - p_y = 0 \\ d_1 - a_2 s_2 - d_4 s_2 c_3 - d_4 c_2 s_3 + a_3 c_2 c_3 - a_3 s_2 s_3 - p_z = 0 \\ s_1^2 + c_1^2 - 1 = 0 \\ s_2^2 + c_2^2 - 1 = 0 \\ s_3^2 + c_3^2 - 1 = 0 \end{gathered} \tag{14}$$

Table 11 presents the expected values for each of the three PUMA's DoFs ($E[|cos(q_i)|]$ and $E[|sin(q_i)|]$), while Table 12 shows the relevant lex orders for the PUMA manipulator. It is important to highlight that the PUMA has three rotational DoFs, just like the hexapod's leg, but the lex orders presented in Table 12 are not the same that the ones used for the hexapod (see Table 4). This is because in the PUMA's case: $E[|sin(q_2)|] > E[|cos(q_2)|]$, so the two trigonometric variables related with q_2 are ordered as: $c_2 > s_2$.

The results of the three-step selection criterion applied to the relevant lex orders of the PUMA manipulator are presented in Table 13. After applying this selection

Table 13. Selection of the lex order for the PUMA's IKM. After each step of the selection process, the discarded orders are marked in *italic*. The first classification criterion is to search for those lex orders whose basis' highest degree equation present the least computation time. The second one seeks for the bases with the lowest accumulated cost between all their equations, while the last one searches for the lesser amount of time required to compute all the coefficients of the basis. All these costs, expressed in clock cycles, were calculated for a microcontroller with an ARM Cortex-M4 CPU (*35*, *36*). The selected lex order is marked in **bold**.

	Classification Criteria		
Order Nº	**Highest Deg. [Cycles]**	**Acc. Cost [Cycles]**	**Coefficients [Cycles]**
1	37	110	*360*
2	37	110	*405*
3	37	110	47
4	**37**	**110**	**47**
5	37	110	*405*
6	37	110	*65*

Table 14. Computation times of the six IKMs generated for the PUMA 560 and its reference model ("Ref." row), when they are presented with all points in the robot's workspace. Column "Avg [ms]" shows the average computation times obtained for all the workspace's points (in milliseconds), while "Min [ms]" and "Max [ms]" present the minimum and maximum registered times, respectively. The selected lex order is marked in **bold**.

Nº	**Min [ms]**	**Avg [ms]**	**Max [ms]**
1	0.228	0.251	0.597
2	0.213	0.235	0.657
3	0.205	0.231	0.633
4	**0.204**	**0.230**	**0.621**
5	0.210	0.233	0.707
6	0.212	0.239	0.678
Ref.	0.108	0.117	0.275

criterion, the selected lex order for the PUMA is the fourth one, which is the same lex order in which the variables were solved while applying the geometric method used in (*8*).

All six IKM's synthesized for the PUMA correctly calculate all the positions inside the robot's workspace, with an RMS error below 4×10^{-9}. The computation times required by the all the IKMs synthesized for the PUMA manipulator are compiled in Table 14. It can be seen that the IKM that has the least average computation time is the one automatically selected by our three-step criterion: lex order 4.

6. Conclusions

This work presents the second version of the procedure originally exposed in (*8*), which applies the Groebner Basis theory to synthesize the IKM of non-redundant open-chain robotic systems. The IKM synthesis is done applying a six step systematic methodology, that does not require any special knowledge of the robot's mechanical structure, besides its Denavit-Hartenberg parameters and the movement range of its

actuators. This improved version also automatically selects the best monomial order for the Groebner Basis that will constitute the core of the final IKM, without requiring any further input from the user. The procedure's output is the synthesized IKM, both in C++ and as a MATLAB® script, which may be used directly in the robot's control system or to simulate its behavior.

The performance analysis presented in Section 5 shows that the developed procedure successfully synthesized the IKMs of a non-redundant multi-legged robot, a Lynxmotion's BH3-R hexapod, and a non-redundant manipulator, a PUMA 560, correctly selecting in both cases the optimal monomial order for the Groebner Basis that constitute the core of those IKMs. The synthesized IKMs are totally comparable, both in precision and computation time, with their respective kinematic models calculated by traditional methods. This implies that the developed procedure represents a systematic solution to the Kinematic Problem of non-redundant open-chain robotic systems, one that is independent of the robot's mechanical structure.

Finally, it is important to highlight that, even though in this work the developed procedure was only used to synthesize the IKM of a multi-legged robot and a non-redundant manipulator, it can also be applied to all Cartesian robotic systems and SCARA robots, covering this way a large range of non-redundant open-chain robotic systems.

7. Future Work

The first proposed future work is to extend the procedure to also cover the end effector's orientation, completing this way the full pose computation for all types of non-redundant open-chain robots.

The second proposed future work is the extension to redundant robots, in order to fully cover all the spectrum of open-chain robotic systems. The application of Groebner Basis theory to a redundant robot will surely produce an underdetermined equation system with infinite solutions. Properly solving these types of equation systems will require the application of kinematic restrictions, such as Lagrange multipliers, as is common for redundant robotic systems.

Disclosure statement

The authors declare no conflict of interest.

Funding

This research was partially funded by Plan Nacional de I+D+i, Agencia Estatal de Investigación del Ministerio de Economía, Industria y Competitividad del Gobierno de España, in the project FEDER-CICYT DPI2017-84201-R.

References

(1) Atique, M.U.; Sarker, R.I.; Ahad, A.R. Development of an 8DOF quadruped robot and implementation of Inverse Kinematics using Denavit-Hartenberg convention, *Heliyon* **2018**, *4* (12), e01053.

(2) Flanders, M.; Kavanagh, R.C. Build-A-Robot: Using virtual reality to visualize the Denavit-Hartenberg parameters, *Computer Applications in Engineering Education* **2015**, *23* (6), 846–853.
(3) Fu, K.S.; Gonzalez, R.C.; Lee, C.S.G. *"Robotics: Control, Sensing, Vision and Intelligence"*; CAD/CAM, robotics, and computer vision; McGraw-Hill, 1987.
(4) Özgür, E.; Mezouar, Y. Kinematic modeling and control of a robot arm using unit dual quaternions, *Robotics and Autonomous Systems* **2016**, *77*, 66–73.
(5) Wang, X.; Han, D.; Yu, C.; Zheng, Z. The geometric structure of unit dual quaternion with application in kinematic control, *Journal of Mathematical Analysis and Applications* **2012**, *389*, 1352–1364.
(6) Aydm, Y.; Kucuk, S. Quaternion based inverse kinematics for industrial robot manipulators with euler wrist, In *2006 IEEE International Conference on Mechatronics, ICM*, 2006; pp 581–586.
(7) Barrientos, A.; Álvarez, M.; Hernández, J.D.; del Cerro, J.; Rossi, C. Modelado de Cadenas Cinemáticas mediante Matrices de Desplazamiento. Una alternativa al método de Denavit-Hartenberg, *RIAI - Revista Iberoamericana de Automática e Informática Industrial* **2012**, *9* (4), 371–382.
(8) Guzmán-Giménez, J.; Valera Fernández, Á.; Mata Amela, V.; Díaz-Rodríguez, M.Á. Synthesis of the Inverse Kinematic Model of Non-Redundant Open-Chain Robotic Systems Using Groebner Basis Theory, *Applied Sciences* **2020**, *10* (8), 2781.
(9) Rodriguez, R.; Cardozo, T.; Ardila, D.L.; Cuellar, C.A. A consistent methodology for the development of inverse and direct kinematics of robust industrial robots, *ARPN Journal of Engineering and Applied Sciences* **2018**, *13* (1), 293–301.
(10) Petrescu, R.V.; Aversa, R.; Akash, B.; Bucinell, R.B.; Corchado, J.M.; Berto, F.; Mirsayar, M.; Apicella, A.; Petrescu, F.I. Anthropomorphic Solid Structures n-R Kinematics, *American Journal of Engineering and Applied Sciences* **2017**, *10* (1), 279 – 291.
(11) Chen, S.; Luo, M.; Abdelaziz, O.; Jiang, G. A general analytical algorithm for collaborative robot (cobot) with 6 degree of freedom (DOF), *Proceedings of the 2017 IEEE International Conference on Applied System Innovation: Applied System Innovation for Modern Technology, ICASI 2017* **2017**, 698–701.
(12) Bouzgou, K.; Ahmed-Foitih, Z. Geometric modeling and singularity of 6 DOF Fanuc 200IC robot, In *4th International Conference on Innovative Computing Technology, INTECH 2014 and 3rd International Conference on Future Generation Communication Technologies, FGCT 2014*, aug; IEEE, 2014; pp 208–214.
(13) Manzoor, S.; Ul Islam, R.; Khalid, A.; Samad, A.; Iqbal, J. An open-source multi-DOF articulated robotic educational platform for autonomous object manipulation, *Robotics and Computer-Integrated Manufacturing* **2014**, *30* (3), 351–362.
(14) Mahajan, A.; Singh, H.P.; Sukavanam, N. An unsupervised learning based neural network approach for a robotic manipulator, *International Journal of Information Technology* **2017**, *9* (1), 1–6.
(15) Duka, A.V. Neural Network based Inverse Kinematics Solution for Trajectory Tracking of a Robotic Arm, *Procedia Technology* **2014**, *12*, 20–27.
(16) Toshani, H.; Farrokhi, M. Real-time inverse kinematics of redundant manipulators using neural networks and quadratic programming: A Lyapunov-based approach, *Robotics and Autonomous Systems* **2014**, *62* (6), 766–781.
(17) Rokbani, N.; Alimi, A.M. Inverse kinematics using particle swarm optimization, a statistical analysis, In *Procedia Engineering*, jan; Elsevier, 2013; Vol. 64, pp 1602–1611.
(18) Jiang, G.; Luo, M.; Bai, K.; Chen, S. A Precise Positioning Method for a Puncture Robot Based on a PSO-Optimized BP Neural Network Algorithm, *Applied Sciences* **2017**, *7* (10), 969.
(19) Köker, R. A genetic algorithm approach to a neural-network-based inverse kinematics solution of robotic manipulators based on error minimization, *Information Sciences* **2013**, *222*, 528–543.
(20) Doan, N.C.N.; Lin, W. Optimal robot placement with consideration of redundancy prob-

lem for wrist-partitioned 6R articulated robots, *Robotics and Computer-Integrated Manufacturing* **2017**, *48*, 233–242.
(21) Rokbani, N.; Casals, A.; Alimi, A.M.; Azar, A.T.; Vaidyanathan, S. IK-FA, a new heuristic inverse kinematics solver using firefly algorithm, *Studies in Computational Intelligence* **2015**, *575*, 369–385.
(22) Buchberger, B. Gröbner bases and systems theory, *Multidimensional Systems and Signal Processing* **2001**, *12* (3-4), 223–251.
(23) Kendricks, K.D. A kinematic analysis of the gmf a-510 robot: An introduction and application of groebner basis theory, *Journal of Interdisciplinary Mathematics* **2013**, *16* (2-3), 147–169.
(24) Wang, Y.; Hang, L.B.; Yang, T.L. Inverse kinematics analysis of general 6R serial robot mechanism based on groebner base, *Frontiers of Mechanical Engineering in China* **2006**, *1* (1), 115–124.
(25) Rameau, J.F.; Serré, P. Computing mobility condition using Groebner basis, *Mechanism and Machine Theory* **2015**, *91*, 21–38.
(26) Abbasnejad, G.; Carricato, M. Direct Geometrico-static Problem of Underconstrained Cable-Driven Parallel Robots With n Cables, *IEEE Transactions on Robotics* **2015**, *31* (2), 468–478.
(27) Gan, D.; Liao, Q.; Dai, J.S.; Wei, S.; Seneviratne, L.D. Forward displacement analysis of the general 6-6 Stewart mechanism using Gröbner bases, *Mechanism and Machine Theory* **2009**, *44* (9), 1640–1647.
(28) Huang, X.; He, G. Forward kinematics of the general Stewart-Gough platform using Gröbner basis, In *2009 IEEE International Conference on Mechatronics and Automation, ICMA 2009*, 2009; pp 3557–3561.
(29) Uchida, T.; McPhee, J. Using Gröbner bases to generate efficient kinematic solutions for the dynamic simulation of multi-loop mechanisms, *Mechanism and Machine Theory* **2012**, *52*, 144–157.
(30) Cox, D.A.; Little, J.; O'Shea, D. *"Ideals, Varieties, and Algorithms: An Introduction to Computational Algebraic Geometry and Commutative Algebra"*, Fourth ed.; Undergradute Texts in Mathematics series; Springer International Publishing, 2015.
(31) Faugère, J.C. FGb: A library for computing Gröbner bases, In *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*; Springer, Berlin, Heidelberg, 2010; Vol. 6327 LNCS, pp 84–87.
(32) Faugère, J.C. A new efficient algorithm for computing Gröbner bases (F4), *Journal of Pure and Applied Algebra* **1999**, *139* (1), 61–88.
(33) Faugère, J.C.; Gianni, P.; Lazard, D.; Mora, T. Efficient computation of zero-dimensional gröbner bases by change of ordering, *Journal of Symbolic Computation* **1993**, *16* (4), 329–344.
(34) Salzer, H.E. A Note on the Solution of Quartic Equations, *Mathematics of Computation* **1960**, *14* (71), 279–281.
(35) ARM Limited. *Cortex-M4 Technical Reference Manual, Revision r0p0*; Technical Report, 2009. https://developer.arm.com/documentation/ddi0439/b/.
(36) ST Microelectronics. *Getting started with the CORDIC accelerator using STM32CubeG4 MCU Package, AN5325*; Technical Report, 2019. https://www.st.com/en/embedded-software/stm32cubeg4.html.