Calculation of the Boundaries and Barriers of the Workspace of a Redundant Serial-parallel Robot using the Inverse Kinematics

Adrián Peidró, Óscar Reinoso, Arturo Gil, José María Marín, Luis Payá and Yerai Berenguer
Systems Engineering and Automation Department, Miguel Hernández University, 03202, Elche, Spain

Keywords: Robot Manipulator, Redundant Robot, Workspace, Inverse Kinematics, Discretization Methods.

Abstract: This paper presents the workspace analysis of a redundant serial-parallel robot. Due to the complexity of the robot, the complex constraints (joint limits and no-interference between the legs of the robot), and the globally serial structure of the robot, a discretization method based on the forward kinematics would be most appropriate to compute the workspace. However, this widely used method can only obtain the external boundaries of the workspace, missing the internal barriers that hinder the motion of the robot, which may exist inside the boundaries. To avoid missing these barriers, we use a discretization method that uses the solution of the inverse kinematic problem of the robot. By studying the feasibility of attaining a desired position and orientation by the different branches of the solution to the inverse kinematics, the proposed discretization method is able to obtain both the external boundaries and the internal barriers of the workspace. Some examples are presented to show the importance of these internal barriers in the motions of the robot inside the workspace.

1 INTRODUCTION

The workspace of a robot manipulator, which is the set of positions and orientations that its end-effector can reach, is very important for designing the robot and planning its movements. There are many methods to calculate the workspace of robots, but most of them can be classified into three main groups (Merlet, 2006): geometrical methods, singularity-based methods and discretization methods.

The geometrical methods are especially useful for parallel robots, in which the end-effector is controlled by two or more kinematic chains actuating in parallel. These methods obtain first the workspace associated with each kinematic chain individually. Such individual workspaces usually are simple geometrical objects, such as annuli (Merlet et al., 1998), spherical shells (Gosselin, 1990) or tori (Liu and Wang, 2014). After obtaining the individual workspaces, the workspace of the complete robot is obtained as the intersection of these individual workspaces. These methods are very fast and efficient, but they are also limited since they are only available for specific robots. Moreover, with these methods it may be very difficult to take into account some restrictions in the calculation of the workspace, such as the existence of joint limits, or the condition that mechanical interferences between different bodies should not occur.

The singularity-based methods begin by writing all the kinematic restrictions of the robot (including joint limits) as a system of equations. Next, the Jacobian matrix of this system with respect to all the involved variables is derived, excluding the variables that define the position and orientation of the end-effector. Then, the boundaries of the workspace can be obtained as the set of configurations of the robot for which the mentioned Jacobian matrix is not full-rank. These configurations that produce the rank deficiency of the Jacobian matrix may be found analytically (Abdel-Malek and Yang, 2006; Abdel-Malek et al., 2000) or numerically (Haug et al., 1996; Bohigas et al., 2012). The main problem of the singularity-based methods is the fact that all the constraints must be written as equalities, which may result in too large systems of equations. Moreover, some restrictions cannot be modeled as equality constraints, like the condition of no-interference between different bodies.

Finally, the discretization methods are very flexible and can easily deal with all types of constraints, although they can be very computer-intensive. The discretization methods consist of discretizing the Cartesian (or joint) space into a regular or randomized grid of nodes, and solving the inverse (or forward) kinematic problem of the robot for each node to obtain the complete configuration of the robot (Macho et al., 2009; Bonev and Ryu, 2001; Pisla et al., 2013;

Peidró, A., Reinoso, Ó., Gil, A., Marín, J., Payá, L. and Berenguer, Y.
Calculation of the Boundaries and Barriers of the Workspace of a Redundant Serial-parallel Robot using the Inverse Kinematics.
DOI: 10.5220/0005984704120420
In *Proceedings of the 13th International Conference on Informatics in Control, Automation and Robotics (ICINCO 2016) - Volume 2*, pages 412-420
ISBN: 978-989-758-198-4
Copyright © 2016 by SCITEPRESS – Science and Technology Publications, Lda. All rights reserved

Cervantes-Sánchez et al., 2000). Then, it is checked if the obtained configuration satisfies all the considered constraints (e.g. joint limits or no-interference), in which case the configuration is stored as a workspace point. Since the forward kinematic problem is usually easier to solve in serial robots than the inverse problem, it is more convenient to discretize the joint space for these robots. On the contrary, the inverse kinematic problem is usually simpler for parallel robots than the forward problem, therefore the Cartesian space is usually discretized for them.

This paper presents the calculation of the workspace of a redundant serial-parallel robot with ten degrees of freedom, using a discretization method and the solution of the inverse kinematic problem. The robot is composed of some parallel mechanisms connected in series, which grants it a globally serial architecture. Both its globally serial structure and its kinematic redundancy make its forward kinematic problem much simpler than its inverse kinematic problem. Thus, a method based on discretizing its joint space and solving the forward kinematics would be more appropriate for calculating its workspace. Nevertheless, this method misses important information regarding the internal structure of the workspace, which is crucial for planning the movements of the robot. For example, Figure 1a shows the workspace that is obtained when sampling the joint space of the robot and solving its forward kinematic problem. This method simply generates positions for the end-effector and "populates" the workspace with these positions. As a result, this method only yields the external boundaries of the workspace. However, a more careful analysis of the workspace, as the one presented in this paper, reveals the existence of internal barriers inside the workspace, as shown in Figure 1b. These internal barriers, which cannot be detected when generating the workspace by using the forward kinematics, are very important for planning the movements of the robot, since they imply motion impediments.

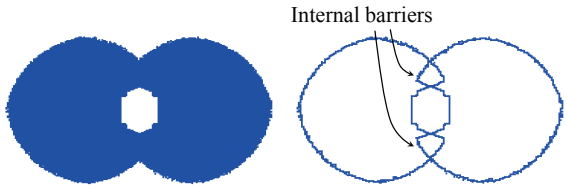

Figure 1: (a) Workspace generated by discretizing the joint space and generating positions solving the forward kinematic problem. This method only yields the external boundaries. (b) Actually, the method based on the forward kinematics misses existing internal barriers that hinder the motion of the robot.

In this paper, we present a discretization method to calculate the workspace of the aforementioned serial-parallel robot, including the internal barriers that lie inside the external boundaries. To this end, the Cartesian space is discretized instead of the joint space, and the inverse kinematic problem is solved instead of the forward kinematics. Although the robot is redundant, the different branches of the solution to its inverse kinematic problem can be exploited to obtain the workspace boundaries and the internal barriers, as this paper shows. After presenting the proposed method, some examples are used to show the importance of the internal barriers of the workspace in the planning of the movements of the robot.

The remainder of this paper is organized as follows. Section 2 presents a ten-degrees-of-freedom serial-parallel robot and reviews the solution to the inverse kinematic problem of this robot. Next, Section 3 presents a method based on the solution of the inverse kinematics to compute the workspace of the robot, including the barriers present inside the workspace. Section 4 presents some examples of the application of the proposed method to calculate different workspaces, and analyzes the effects of the internal barriers on the motion of the robot. Finally, Section 5 concludes this paper.

2 INVERSE KINEMATICS OF A SERIAL-PARALLEL ROBOT

This section reviews the solution of the inverse kinematic problem of the redundant serial-parallel robot shown in Figure 2a. This robot is designed to climb and explore 3D structures, like metallic bridges, to inspect and maintain them. Its feet carry magnets to adhere to the climbed structure. Figure 2b shows an example of the robot performing a transition between two perpendicular beams of a 3D structure.

The robot is biped, and each leg $j \in \{A,B\}$ is composed of two parallel mechanisms connected in series. In each leg j, the i-th parallel mechanism ($i \in \{1,2\}$) has two linear actuators with lengths u_{ij} and v_{ij} (see Figure 2c). Furthermore, each leg is connected to the hip H through one revolute actuated joint, which rotates the leg an angle θ_j with respect to the hip. Thus, the robot has ten degrees of freedom. The joint coordinates associated to these ten degrees of freedom are the lengths of the eight linear actuators and the two rotated angles of the hip, i.e. $\mathbf{q} = [u_{1A}, v_{1A}, u_{2A}, v_{2A}, u_{1B}, v_{1B}, u_{2B}, v_{2B}, \theta_A, \theta_B]^T$.

The geometric design of the robot depends on four parameters, which can be seen in Figure 2: the width b and length h of the central body of the legs,

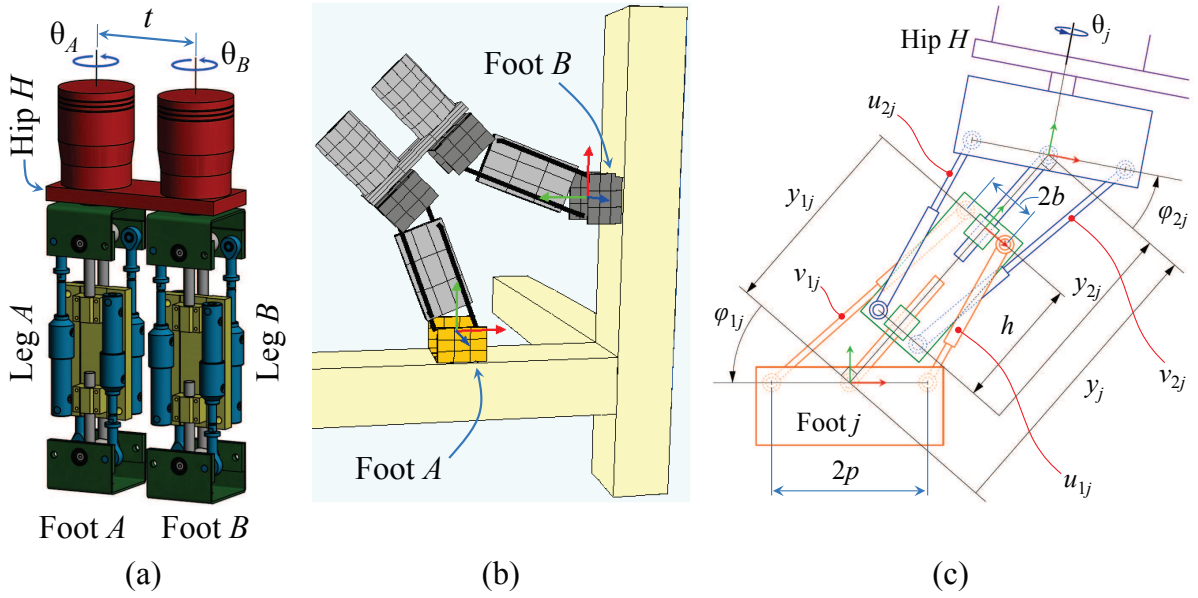

Figure 2: (a) CAD model of the serial-parallel robot studied in this paper. (b) Example of the robot performing a transition between two perpendicular beams in a 3D structure. (c) Detailed view of the kinematics of the leg j of the robot ($j \in \{A,B\}$).

the width p of the feet, and the separation t between the rotations of the hip. Furthermore, in practice the lengths of the linear actuators should be limited, i.e. $u_{ij}, v_{ij} \in [\rho_0, \rho_0 + \Delta\rho]$, where $\rho_0 > 0$ is the minimum length of these actuators and $\Delta\rho > 0$ is their stroke. Thus, the parameters ρ_0 and $\Delta\rho$ will also be regarded as design parameters of the robot.

The inverse kinematic problem of this robot was solved in (Peidro et al., 2015), and the solution will be summarized here since we will use it to compute the workspace of the studied robot. The inverse kinematic problem consists of finding the values of the joint coordinates $\mathbf{q}$ necessary to attain a desired position and orientation for the foot B relative to the foot A, given by the following homogeneous transform matrix ${}^A\mathbf{T}_B$:

$$ {}^A\mathbf{T}_B = \begin{bmatrix} r_{11} & r_{12} & r_{13} & p_x \\ r_{21} & r_{22} & r_{23} & p_y \\ r_{31} & r_{32} & r_{33} & p_z \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{1} $$

where r_{ij} are the components of the rotation matrix that defines the orientation of the foot B with respect to the foot A and $\mathbf{p} = [p_x, p_y, p_z]^T$ is the position vector of the foot B relative to the foot A.

As stated above, the inverse kinematic problem consists of calculating the vector $\mathbf{q}$ of the joint coordinates from the relative position and orientation between the feet of the robot, given by the matrix ${}^A\mathbf{T}_B$. However, as demonstrated in (Peidro et al., 2015), the inverse kinematic problem becomes easier by defining a set of *intermediate joint coordinates* $\tilde{\mathbf{q}} = [\varphi_{1A}, \varphi_{2A}, \varphi_{1B}, \varphi_{2B}, y_{1A}, y_A, y_{1B}, y_B, \theta_A, \theta_B]^T$, which can be solved more easily in terms of ${}^A\mathbf{T}_B$ (the geometrical meaning of each intermediate joint coordinate is shown in Figure 2c). Then, it can be shown that after solving $\tilde{\mathbf{q}}$, the intermediate joint coordinates unequivocally determine the joint coordinates $\mathbf{q}$. The solution for the intermediate joint coordinates depends on the desired orientation, distinguishing two cases (Peidro et al., 2015).

2.1 Case 1: $r_{33}^2 \neq 1$

Assume that the four intermediate joint coordinates $\{\varphi_{1B}, y_B, y_{1A}, y_{1B}\}$ are known. In that case, it can be shown that the remaining six intermediate joint coordinates $\{\varphi_{1A}, \varphi_{2A}, \varphi_{2B}, y_A, \theta_A, \theta_B\}$ can be calculated in terms of $\{\varphi_{1B}, y_B\}$ by performing the following operations. First, φ_{2B} is calculated as follows:

$$ s_B = \sigma_1 \frac{r_{32}}{\sqrt{1 - r_{33}^2}}, \quad c_B = \sigma_1 \frac{-r_{31}}{\sqrt{1 - r_{33}^2}} \tag{2} $$

$$ \varphi_{2B} = \varphi_{1B} - \text{atan2}(s_B, c_B) \tag{3} $$

where $\sigma_1 \in \{-1, 1\}$ and $\text{atan2}(\sin(\psi), \cos(\psi))$ is the inverse tangent function that uses the sine and the cosine of an angle ψ to calculate the angle in the correct quadrant. Next, θ_A is calculated as follows:

$$ s_{\theta_A} = -\frac{r_{31} y_B s_{\varphi_{1B}} + r_{32} y_B c_{\varphi_{1B}} + p_z}{t} \tag{4} $$

$$ c_{\theta_A} = \sigma_2 \sqrt{1 - s_{\theta_A}^2} \tag{5} $$

$$ \theta_A = \text{atan2}\,(s_{\theta_A}, c_{\theta_A}) \tag{6} $$

where $\sigma_2 \in \{-1,1\}$, $s_{\varphi_{1B}} = \sin(\varphi_{1B})$, and $c_{\varphi_{1B}} = \cos(\varphi_{1B})$. Following, θ_B can be calculated as:

$$s_{\theta_B} = r_{33}s_{\theta_A} + c_{\theta_A}(c_B r_{31} - s_B r_{32}) \quad (7)$$

$$c_{\theta_B} = r_{33}c_{\theta_A} - s_{\theta_A}(c_B r_{31} - s_B r_{32}) \quad (8)$$

$$\theta_B = \text{atan2}\,(s_{\theta_B}, c_{\theta_B}) \quad (9)$$

Finally, the remaining intermediate joint coordinates $\{\varphi_{1A}, \varphi_{2A}, y_A\}$ can be calculated as follows:

$$s_A = s_B r_{11} + c_B r_{12}, \quad c_A = s_B r_{21} + c_B r_{22} \quad (10)$$

$$\Omega_1 = r_{11}y_B s_{\varphi_{1B}} + r_{12}y_B c_{\varphi_{1B}} + p_x - tc_{\theta_A}c_A \quad (11)$$

$$\Omega_2 = r_{21}y_B s_{\varphi_{1B}} + r_{22}y_B c_{\varphi_{1B}} + p_y + tc_{\theta_A}s_A \quad (12)$$

$$y_A = \sqrt{\Omega_1^2 + \Omega_2^2} \quad (13)$$

$$\varphi_{1A} = \text{atan2}\,(\Omega_1, \Omega_2) \quad (14)$$

$$\varphi_{2A} = \varphi_{1A} - \text{atan2}\,(s_A, c_A) \quad (15)$$

2.2 Case 2: $r_{33}^2 = 1$

In this case, we assume that the values of the five intermediate joint coordinates $\{\varphi_{1B}, \varphi_{2B}, y_B, y_{1A}, y_{1B}\}$ are known. Then, by defining $s_B = \sin(\varphi_{1B} - \varphi_{2B})$ and $c_B = \cos(\varphi_{1B} - \varphi_{2B})$, we can still compute the values of the remaining five intermediate joint coordinates $\{\varphi_{1A}, \varphi_{2A}, y_A, \theta_A, \theta_B\}$ using Eqs. (4) to (15).

2.3 Feasible Regions

The solutions described in subsections 2.1 and 2.2 assume that some of the intermediate joint coordinates are known, and compute the remaining intermediate joint coordinates in terms of the former. This is due to the kinematic redundancy of the robot: fixing the position and orientation of one foot with respect to the other foot only provides six independent equations (three for the position and three for the orientation) to calculate ten unknown intermediate joint coordinates. Thus, six independent equations only allow us to calculate six intermediate joint coordinates in terms of the remaining four, as described in subsection 2.1. Moreover, when the desired orientation between the feet satisfies $r_{33}^2 = 1$, φ_{2B} cannot be determined from the equations and its value must also be assumed to be known (subsection 2.2).

According to the solutions described in the previous subsections, the solution of the inverse kinematic problem for a given desired position and orientation is a four-dimensional set in the ten-dimensional space of the intermediate joint coordinates (or a five-dimensional set if the desired orientation satisfies $r_{33}^2 = 1$). This is because the ten intermediate joint coordinates have been solved in terms of four (or five) of the intermediate joint coordinates [called *decision variables* in (Peidro et al., 2015)], whose values can be freely decided. In contrast, in non-redundant robots the solution to the inverse kinematics is a set of isolated points in the joint space (i.e., a zero-dimensional set). Due to the high dimensionality of the solution sets in this robot, it is not possible to represent graphically the complete solutions to the inverse kinematics.

However, as demonstrated in (Peidro et al., 2015), it is possible to reduce the dimensionality of these solution sets without losing relevant information, obtaining an alternative and more compact representation of the solutions of the inverse kinematic problem. This reduction of dimensionality is based on the fact that not all the decision variables (i.e. the four or five intermediate joint coordinates in terms of which the remaining unknowns are calculated) are equally important to determine the posture of the robot. In fact, it can be shown that after fixing the values of the decision variables $\{\varphi_{1B}, y_B\}$ (and also φ_{2B}, if $r_{33}^2 = 1$), the overall posture of the robot is determined, and the remaining decision variables $\{y_{1A}, y_{1B}\}$ only produce small internal motions of the central bodies of the legs along the legs, which do not affect the overall posture of the robot. Since the overall posture of the robot only depends on $\{\varphi_{1B}, y_B\}$ (and also φ_{2B}, if $r_{33}^2 = 1$), we can focus only on these two (or three) variables to analyze the solutions of the inverse kinematic problem, and use this information to discard the pairs $\{\varphi_{1B}, y_B\}$ that yield a forbidden posture in which the legs of the robot interfere, for example.

Exploiting the previous idea regarding the possibility of reducing the dimension of the solution sets, (Peidro et al., 2015) developed a Monte Carlo algorithm to obtain the regions R_f of the φ_{1B}-y_B plane (or the regions of the φ_{1B}-φ_{2B}-y_B space, if $r_{33}^2 = 1$) which allow the robot to attain a desired position and orientation ${}^A\mathbf{T}_B$ satisfying the joint limits, and they called these regions *feasible regions*. Any point of these feasible regions yields a valid solution of the inverse kinematic problem, satisfying the condition that the lengths of the eight linear actuators of the robot should be between the joint limits $[\rho_0, \rho_0 + \Delta\rho]$. In this paper, we will impose an additional condition to calculate the feasible regions: in order for a pair $\{\varphi_{1B}, y_B\}$ (or triplet $\{\varphi_{1B}, \varphi_{2B}, y_B\}$, if $r_{33}^2 = 1$) to belong to the feasible regions R_f, the posture corresponding to that pair (or triplet) should not produce mechanical interferences between the legs of the robot. To check this condition of no-interference between the legs, the Separating Axis Theorem will be used (Ericson, 2004). This will be explained in Section 4 in more detail.

According to subsection 2.1, the solution to the inverse kinematics depends on two binary variables $\sigma_1, \sigma_2 \in \{-1, 1\}$ when the desired orientation satisfies $r_{33}^2 \neq 1$. Each combination of these binary variables will yield a different feasible region R_f, which corresponds to a different branch of the solution to the inverse kinematics. Thus, when $r_{33}^2 \neq 1$, the solution to the inverse kinematics will have four different branches, identified by the four possible combinations (σ_1, σ_2) of these binary variables: $(1,1)$, $(1,-1)$, $(-1,1)$, $(-1,-1)$. If the desired orientation satisfies $r_{33}^2 = 1$, then according to subsection 2.2 only the binary variable σ_2 is involved in the solution of the inverse kinematics. Thus, the inverse kinematic problem will only have two different branches in that case, one branch corresponding to $\sigma_2 = 1$ and the other branch corresponding to $\sigma_2 = -1$.

3 OBTAINING THE WORKSPACE USING THE INVERSE KINEMATICS

In this section, we will describe how the solution to the inverse kinematic problem, described in Section 2, can be used to obtain the workspace of the serial-parallel robot studied in this paper, including its external boundaries and internal barriers.

According to the previous section, the solution to the inverse kinematic problem of the serial-parallel robot studied in this paper can be summarized as follows. Given a desired position and orientation between the feet of the robot, encoded by the homogeneous transformation matrix ${}^A\mathbf{T}_B$ defined in Eq. (1), the solution to the inverse kinematics depends on r_{33}:

- If the desired orientation satisfies $r_{33}^2 \neq 1$, then the solution to the inverse kinematics can be represented by a feasible region R_f in the φ_{1B}-y_B plane, such that any point of this region corresponds to a posture that allows the robot to attain the desired position and orientation ${}^A\mathbf{T}_B$ satisfying the joint limits of the linear actuators and guaranteeing that the legs of the robot do not interfere. Moreover, there exist four different branches for the solution, i.e. four different feasible regions R_f corresponding to the four possible combinations of the binary variables σ_1 and σ_2.
- If the desired orientation satisfies $r_{33}^2 = 1$, the solution to the inverse kinematics can be represented by a feasible region R_f in the φ_{1B}-φ_{2B}-y_B space, such that any point of R_f yields a posture that allows the robot to attain the desired position and orientation ${}^A\mathbf{T}_B$ satisfying the joint limits of the linear actuators and guaranteeing that the legs of the robot do not interfere. In this case, there exist two different branches for the solution, i.e. two different feasible regions R_f, one region per each value of the binary variable σ_2.

Once the solution to the inverse kinematics is available (including all the branches of this solution), the workspace boundaries and barriers can be obtained using a discretization algorithm explained next.

The workspace of this robot can be defined as the set of positions and orientations that the foot B can attain with respect to the foot A. Such a workspace is a six-dimensional set, since the position and orientation of the foot B relative to the foot A can be represented by three translations $\mathbf{p} = [p_x, p_y, p_z]^T$ and three rotations $\mathbf{r} = [\alpha, \beta, \gamma]^T$ (where α, β and γ are Euler angles). To calculate the workspace, the six-dimensional space of the variables $\{p_x, p_y, p_z, \alpha, \beta, \gamma\}$ is discretized into a regular grid of nodes. For example, we may approximate each axis of this six-dimensional space by n_d nodes regularly distributed between two limits, which yields an overall grid of n_d^6 nodes. Then, for each node of this grid, the inverse kinematic problem if solved to check if the node is attainable by each branch of the solution of the inverse kinematics.

For each branch i of the solution to the inverse kinematics, we create a list WS_i of the nodes that can be reached using that branch. After the algorithm has checked all the nodes of the grid, the list WS_i is an approximation of the workspace associated to the branch i, since it contains all the positions and orientations that can be reached with that branch. Then, the boundaries of the workspace associated to the i-th branch can be approximated by the nodes contained in WS_i which have at least one unreachable neighboring node (i.e., a neighboring node not contained in WS_i). After obtaining the boundaries of the workspaces associated to the different branches, the boundaries of all the branches can be joined to obtain the boundaries and barriers of the complete workspace.

To determine if an arbitrary node of the grid is attainable using a given branch i of the inverse kinematics, the feasible region R_f associated to that branch is calculated using the Monte Carlo algorithm described in (Peidro et al., 2015). If the region R_f is empty, then it is considered that the node is not attainable by the i-th branch, and the node is not included in the list WS_i (see Figure 3). The mentioned Monte Carlo algorithm generates the region R_f by randomly sampling points in the φ_{1B}-y_B plane (or in the φ_{1B}-φ_{2B}-y_B space, if $r_{33}^2 = 1$). If the posture generated by each randomly sampled point satisfies the joint limits of the linear actuators, and if the legs do not interfere, then that point is stored as a point of the feasible region R_f. In this

way, a discrete approximation of the region R_f can be obtained by sampling a large number of points in the φ_{1B}-y_B plane (or in the φ_{1B}-φ_{2B}-y_B space).

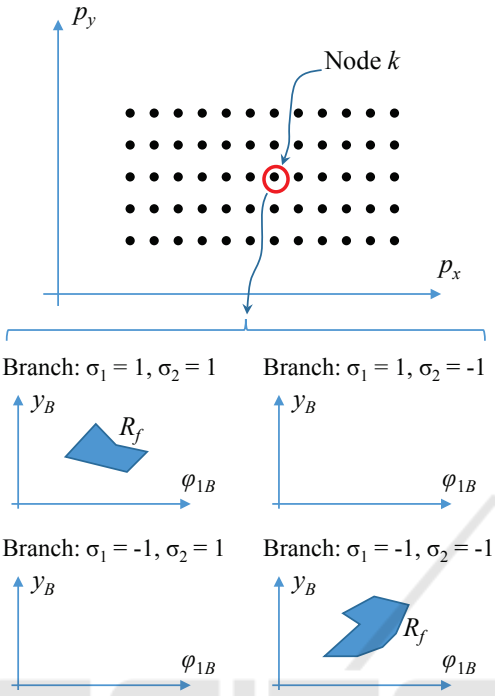

Figure 3: A 2D example that illustrates the process to determine whether a given node k of the potential workspace is reachable or not. For a given node k, the feasible regions R_f are calculated using all the branches of the solution of the inverse kinematic problem. In this case, the node k belongs only to the workspaces of the branches $(1,1)$ and $(-1,-1)$ since these are the only branches leading to non-empty feasible regions R_f.

Note that it is sufficient to find a single point belonging to R_f to guarantee that R_f is not empty and classify the corresponding node of the workspace as attainable. However, checking if R_f is empty (and classifying the corresponding node as unattainable by the i-th branch of the inverse kinematics) is not that easy, since one would need to explore exhaustively the φ_{1B}-y_B plane (or the φ_{1B}-φ_{2B}-y_B space) to guarantee that this plane (space) does not have points satisfying all the constraints (i.e., to guarantee that R_f is empty). Since it is not feasible to perform such an exhaustive search to check if R_f is empty, this problem is practically solved by establishing a maximum number n_a of attempts to generate a point in R_f. Then, the aforementioned Monte Carlo algorithm begins to randomly sample points in the φ_{1B}-y_B plane (or in the φ_{1B}-φ_{2B}-y_B space, if $r_{33}^2 = 1$). If it finds a point that satisfies all the constraints (joint limits and no-interference), the Monte Carlo algorithm stops: the region R_f is not empty (it contains at least one point) and the node is classified as attainable. If, on the contrary, the Monte Carlo algorithm has sampled n_a random points and none of them satisfies the constraints, it is considered that R_f is empty, which means that the corresponding node cannot be attained by the considered branch of the inverse kinematics. Obviously, this method will be more accurate when n_a increases, but the computation will also take more time, so a compromise between precision and computational cost is necessary.

It should be remarked that the method described in this section to compute the workspace is very computer-intensive if we try to discretize and compute the six-dimensional workspace of the variables $\{p_x, p_y, p_z, \alpha, \beta, \gamma\}$, since the number of nodes to check is n_d^6 (and, for each node, the feasible regions R_f must be obtained for the different branches of the inverse kinematics). Moreover, a six-dimensional workspace cannot be represented graphically. For these reasons, and to decrease the computational cost, we will fix some of these six variables to obtain lower-dimensional workspaces that can be represented graphically and are more easy to understand, such as the constant-orientation workspace (i.e., the set of positions that can be attained by the foot B with respect to the foot A when the relative orientation between the feet is constant). The constant-orientation workspace is very useful when planning some movements which are necessary to explore a 3D structure, such as when performing a transition between different beams (see Figure 2b). In the following section, we will illustrate the algorithm described in the present section using some examples.

4 EXAMPLES

This section presents some examples of the application of the method described in Section 3 to compute different workspaces of the studied robot. For the next examples, the geometric design parameters of the robot are: $b = p = 4$, $h = 16$, $t = 15.6$, $\rho_0 = 19$, and $\Delta\rho = 7.5$ (all in cm). In all the examples of this section, we will discretize the workspace into $n_d = 200$ points in each axis. Moreover, we will sample a maximum of $n_a = 5000$ random points before deciding that the feasible region R_f (of valid postures that yield a given position and orientation between the feet) is empty (n_a was defined in Section 3). In all the examples, we will assume that the foot A is firmly attached to the structure, and we will compute the set of positions that the foot B (which is free to move) can reach.

4.1 Example 1

In this example, we are interested in obtaining a constant-orientation workspace, i.e. the set of attainable points by the foot B when the relative orientation between the feet is constant. More specifically, we will study the workspace obtained when both feet have the same orientation, which means that the rotation submatrix of ${}^{A}\mathbf{T}_{B}$ is the identity matrix. Furthermore, we will be interested only in the intersection of such constant-orientation workspace with the plane $p_z = 0$, to study the planar motions of the robot inside this constant-orientation workspace. Note that the robot needs to perform motions of this type to travel along a beam of a structure, as shown in Figure 4. The desired position and orientation will have the following form for this example:

$$ {}^{A}\mathbf{T}_{B} = \begin{bmatrix} 1 & 0 & 0 & p_x \\ 0 & 1 & 0 & p_y \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{16} $$

To apply the algorithm described in the previous section, we will define a box $\mathcal{B}$ that encloses the workspace in the p_x-p_y plane, and we will discretize this box into a grid of 40000 nodes regularly distributed (200 nodes per each axis). Then, for each node, we will solve the inverse kinematic problem using the matrix of Eq. (16) as the input, to check if each node is attainable by the different branches of the inverse kinematics. The chosen box for this example is $\mathcal{B} = \{(p_x, p_y) : -80 \text{ cm} \leq p_x \leq 80 \text{ cm}, -50 \text{ cm} \leq p_y \leq 50 \text{ cm}\}$. Running the algorithm described in the previous section for these parameters, the workspace shown in Figure 4 is obtained. Note that, since in this case $r_{33}^2 = 1$, according to Section 2.3 the solution to the inverse kinematic problem in this case has two branches: one for $\sigma_2 = 1$ and other for $\sigma_2 = -1$. The workspaces associated to these branches are shown in Figure 4 with different colors.

Note that this workspace is split into the components associated with the two branches of the solution to the inverse kinematic problem. The complete workspace (i.e. the union of the two components) has a void around the foot A, which is necessary to avoid the interferences between the legs. According to Figure 4, the robot cannot move the foot B from one side of the workspace (e.g. the right half the workspace, associated with the branch $\sigma_2 = 1$) to the other side (e.g. the left half, associated with the branch $\sigma_2 = -1$) keeping constant the orientation between the feet, since the workspaces associated with both branches have boundaries in the middle of the workspace, both above and below the foot A. This is

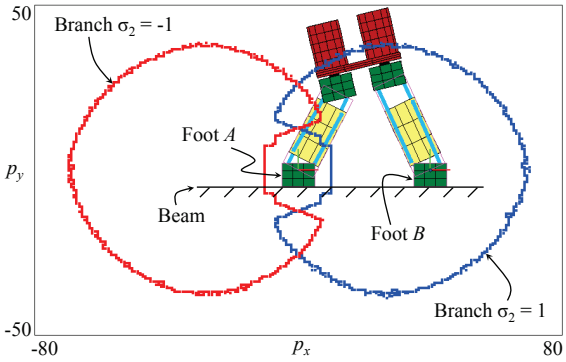

Figure 4: Planar constant-orientation workspace that provides the points at which the foot B can be placed with the same orientation as the foot A. This constant-orientation workspace is useful to plan movements along the direction of the beam. The workspaces associated with the branches $\sigma_2 = -1$ and $\sigma_2 = 1$ are represented in red and blue colors, respectively.

illustrated in Figure 5, where the robot starts at an initial point in the workspace associated with the branch $\sigma_2 = 1$, and tries to describe a trajectory towards the left half of the workspace (see Figure 5a). However, the trajectory cannot be completed because the robot cannot cross the boundaries of the component of the workspace in which it moves. This boundary is originated from the fact that the legs cannot interfere: as Figure 5b shows, when the foot B is close to the mentioned boundary, both legs are about to intersect.

To check if the two legs interfere in all the examples of this paper, the Separating Axis Theorem is used (Ericson, 2004). Each leg can be approximated by the union of two *cuboids* (also known as *rectangular parallelepipeds*): one cuboid encloses the foot, and the other cuboid encloses the central body of the leg, including the linear actuators (these cuboids are represented in magenta in Figure 5b). Then, the two legs will interfere if one of the cuboids of one leg intersects one of the cuboids of the other leg. Since the cuboids are convex shapes, the Separating Axis Theorem can easily be used to check if they intersect.

4.2 Example 2

In this example, the objective is to find the planar constant-orientation workspace defined by the following homogeneous transformation matrix between the feet of the robot:

$$ {}^{A}\mathbf{T}_{B} = \begin{bmatrix} 0 & -1 & 0 & p_x \\ 1 & 0 & 0 & p_y \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{17} $$

This orientation is a rotation of 90° about the Z axis, which is necessary to perform a transition between

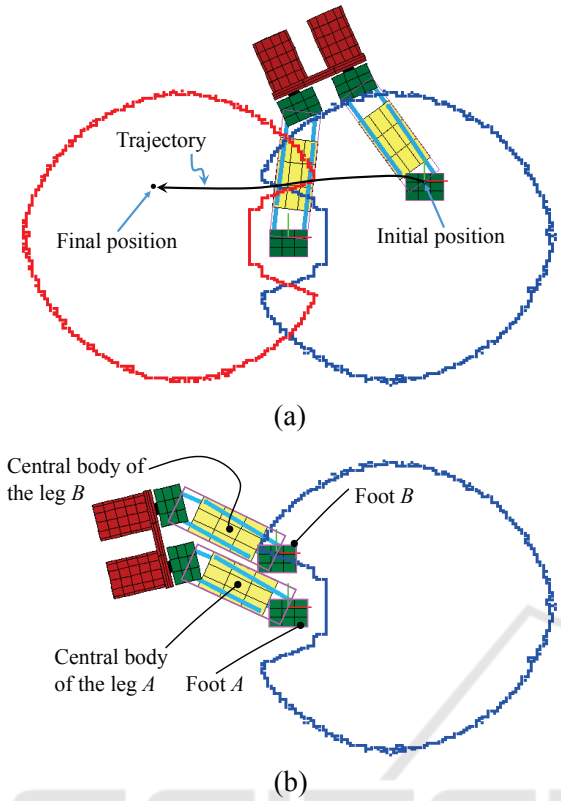

Figure 5: (a) A trajectory between both components of the workspace. (b) The robot cannot reach the left component because it cannot cross a boundary of the component in which it moves (the right component). When approaching the boundary, the legs are about to intersect: the foot B is almost touching the central body of the leg A. To cross the boundary, an interference between these two bodies would be necessary.

two perpendicular beams in a structure, as shown in Figures 2b and 6. Again, we are interested only in the intersection of this constant-orientation workspace with the plane $p_z = 0$, since the motion necessary to perform such a transition is planar.

Next, the algorithm described in Section 3 is applied, discretizing the following box $\mathcal{B}$ into a grid of 40000 nodes (200 nodes per axis): $\mathcal{B} = \{(p_x, p_y) : -40 \text{ cm} \leq p_x \leq 80 \text{ cm}, -40 \text{ cm} \leq p_y \leq 80 \text{ cm}\}$. The resulting workspace is shown in Figure 6. Again, since the desired orientation satisfies $r_{33}^2 = 1$, the solution to the inverse kinematic problem has two branches, one for $\sigma_2 = 1$ (shown in blue in Figure 6) and other for $\sigma_2 = -1$ (shown in red in Figure 6).

In this case, the workspace attainable using the branch $\sigma_2 = -1$ of the solution to the inverse kinematic problem is smaller than the workspace associated with the branch $\sigma_2 = 1$. Note that the workspace associated with the branch $\sigma_2 = -1$ has two components: a big one, which is close to the foot A in Figure

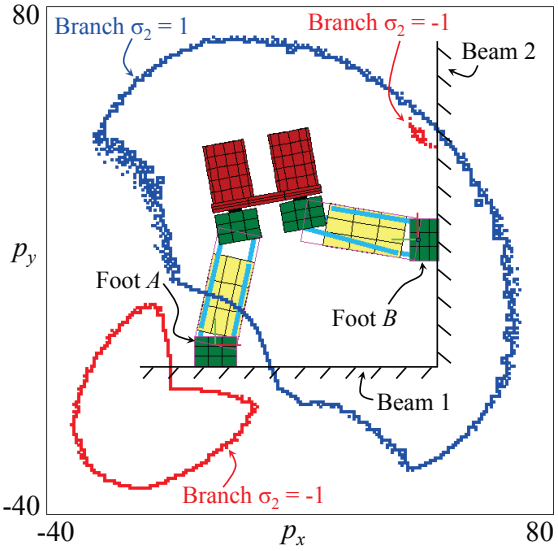

Figure 6: Planar constant-orientation workspace containing all the points of the plane $p_z = 0$ that can be reached with the foot B rotated 90° about the Z axis with respect to the foot A.

6, and a smaller one, which is close to the beam 2 in the same figure. Actually, the shape of the smaller component cannot be appreciated very well in Figure 6. However, a more precise approximation of that component can be obtained if the algorithm described in Section 3 is executed again discretizing a smaller box $\mathcal{B}$ that encloses only the area around the mentioned small component of the workspace, instead of using a big box that contains all the components as shown in Figure 6.

The posture of the robot shown in Figure 6, with the foot B placed on the beam 2, is obtained using the branch $\sigma_2 = 1$ of the solution to the inverse kinematics. However, according to the same figure, it would be also possible to place the foot B on the beam 2 using the branch $\sigma_2 = -1$, since this branch yields a small component of the workspace near the beam 2 (i.e., the small component described in the previous paragraph). This is checked in Figure 7, which shows a posture of the robot that places the foot B at a point of the smallest of the two workspace components associated with the branch $\sigma_2 = -1$. As this figure shows, using the branch $\sigma_2 = -1$, the foot B can also be effectively placed on the beam 2 with the desired orientation. However, in this posture, the hip of the robot intersects the beams, so this would not be a feasible solution in practice. This unfeasibility can be easily detected by the presented method if we include the condition that no part of the robot should intersect the obstacles of the environment, using a similar procedure to the method used to check if different legs intersect, described in subsection 4.1.

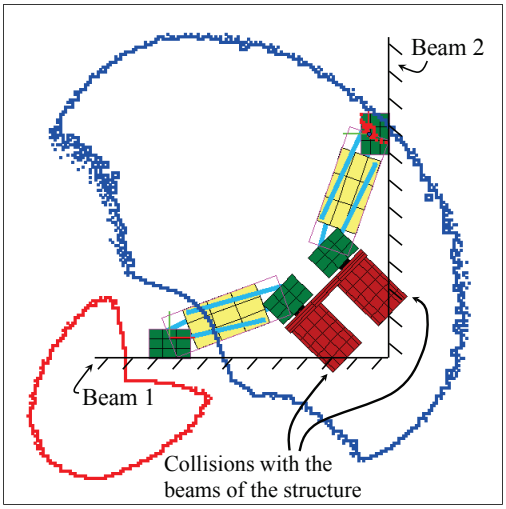

Figure 7: A posture which places the foot B on the beam 2 with the desired orientation, using the branch $\sigma_2 = -1$. In this posture, the robot collides with the structure.

5 CONCLUSIONS

This paper has presented a discretization method to calculate the workspace of a serial-parallel redundant robot using the solution to its inverse kinematic problem. In contrast to the methods based on the forward kinematics, the proposed method is able to obtain both the external boundaries and the internal barriers of the workspace. Using this method, we have analyzed the negative effects of the internal barriers on the movements of the robot inside its workspace.

In the future, we will solve the path planning problem of this robot. We hope that the analysis presented in this paper will be very useful to plan the movements of the robot, since the presented method generates workspace maps that allow the robot to navigate safely through its workspace, knowing the positions at which the actuators attain the joint limits, or the positions where the legs of the robot would collide.

ACKNOWLEDGEMENTS

This work has been supported by the Spanish Ministry of Education through grant FPU13/00413, by the Spanish Ministry of Economy through project DPI2013-41557-P, and by the Generalitat Valenciana through project AICO/2015/021.

REFERENCES

Abdel-Malek, K., and Yang, J. (2006). Workspace boundaries of serial manipulators using manifold stratification. *The International Journal of Advanced Manufacturing Technology*, Vol. 28(11), pp. 1211–1229.

Abdel-Malek, K., Yeh, H.J., and Othman, S. (2000). Interior and exterior boundaries to the workspace of mechanical manipulators. *Robotics and Computer-Integrated Manufacturing*, Vol. 16(5), pp. 365–376.

Bohigas, O., Manubens, M., and Ros, L. (2012). A Complete Method for Workspace Boundary Determination on General Structure Manipulators. *IEEE Transactions on Robotics*, Vol. 28(5), pp. 993–1006.

Bonev, I.A., and Ryu, J. (2001). A new approach to orientation workspace analysis of 6-DOF parallel manipulators. *Mechanism and Machine Theory*, Vol. 36(1), pp. 15–28.

Cervantes-Sánchez, J.J., Hernández-Rodríguez, J.C., and Rendón-Sánchez, J.G. (2000). On the workspace, assembly configurations and singularity curves of the RRRRR-type planar manipulator. *Mechanism and Machine Theory*, Vol. 35(8), pp. 1117–1139.

Ericson, C. (2004). *Real-Time Collision Detection*. CRC Press.

Gosselin, C. (1990). Determination of the Workspace of 6-DOF Parallel Manipulators. *ASME Journal of Mechanical Design*, Vol. 112(3), pp. 331–336.

Haug, E.J., Luh, C.M., Adkins, F.A., and Wang, J.Y. (1996). Numerical Algorithms for Mapping Boundaries of Manipulator Workspaces. *ASME Journal of Mechanical Design*, Vol. 118(2), pp. 228–234.

Liu, X.J., and Wang, J. (2014). *Parallel Kinematics. Type, Kinematics, and Optimal Design*. Springer-Verlag Berlin Heidelberg.

Macho, E., Altuzarra, O., Amezua, E., and Hernandez, A. (2009). Obtaining configuration space and singularity maps for parallel manipulators. *Mechanism and Machine Theory*, Vol. 44(11), pp. 2110–2125.

Merlet, J.P. (1995). Determination of the orientation workspace of parallel manipulators. *Journal of Intelligent and Robotic Systems*, Vol. 13(2), pp. 143–160.

Merlet, J.P. (2006). *Parallel Robots*. Springer Netherlands.

Merlet, J.P., Gosselin, C.M., and Mouly, N. (1998). Workspaces of planar parallel manipulators. *Mechanism and Machine Theory*, Vol. 33(1-2), pp. 7–20.

Peidro, A., Gil, A., Marin, J.M., and Reinoso, O. (2015). Inverse kinematic analysis of a redundant hybrid climbing robot. *International Journal of Advanced Robotic Systems*, 12:163, pp. 1–16.

Pisla, D., Szilaghyi, A., Vaida, C., and Plitea, N. (2013). Kinematics and workspace modeling of a new hybrid robot used in minimally invasive surgery. *Robotics and Computer-Integrated Manufacturing*, Vol. 29(2), pp. 463–474.