Workspace Determination of Kinematic Redundant Manipulators using a Ray-Based Method

Angelica Ginnante, Stéphane Caro, Enrico Simetti, François Leborne

Proceedings of the ASME 2023
International Design Engineering Technical Conferences and
Computers and Information in Engineering Conference
IDETC/CIE 2023
August 20-23, 2023, Boston, Massachusetts

IDETC2023-115240

WORKSPACE DETERMINATION OF KINEMATIC REDUNDANT MANIPULATORS USING A RAY-BASED METHOD

Angelica Ginnante[1,2,3], **Stéphane Caro**[1], **Enrico Simetti**[2], **François Leborne**[3]
[1]Nantes Université, École Centrale Nantes, CNRS, LS2N, UMR 6004, F-44000 Nantes, France
[2]University of Genova, DIBRIS, 16145 Genova, Italy
[3]Nimbl'Bot, 7 Avenue de Guitayne, 33610 Canéjan, France
Email: aginnante@nimbl-bot.com, stephane.caro@ls2n.fr, Enrico.Simetti@unige.it, fleborne@nimbl-bot.com

ABSTRACT

Determining the workspace of a robotic manipulator is extremely significant for knowing its abilities and planning the robot application. There exist several techniques for the robot workspace determination. However, these methods usually are affected by computational redundancy, like in the case of Monte Carlo based methods, or their implementation is difficult. Moreover, the workspace analysis of kinematic redundant manipulators is even more complex. This paper introduces a ray-based workspace determination algorithm, easy to implement and not affected by computational redundancy. The proposed method can be applied to any type of serial robot, but it is tested only on spatial kinematic redundant robots. The results show how the approach can clearly determine the boundary of the robot workspace in a short period of time. Finally the time and quality performance of the ray-based method results are compared to the Monte Carlo one demonstrating the improvement of the proposed method.

1 INTRODUCTION

The workspace of a manipulator represents the space where its end-effector can reach any point for at least one orientation [1]. Planning the robot end-effector movements and trajectories requires careful consideration of the workspace analysis. As a result, the workspace of conventional robots has been the subject of numerous scientific studies throughout the past three decades. There exist three main types of methods for the workspace determination, as described in [2]. The first one is the geometric type, which is mainly applied for the workspace identification of planar robots [3]. This approach is intuitive, but it can not accurately describe the workspace of spatial, i.e. non-planar, robots. The second method is the analytic one that employs the kinematic Jacobian matrix [4]. The boundaries of the workspace are generated searching for the rank deficiency of the kinematic Jacobian matrix. By imposing the kinematic Jacobian matrix rank deficiency, a set of equations is obtained and solved to identify the workspace boundaries. This type of approach is complex when applied to kinematic redundant robot [5]. Finally, the third approach is a numerical one, which identifies the workspace boundaries using the forward kinematics of the robot. This type of methods can be applied to any robot and its solution is easily understandable [2]. It is usually employed for the workspace analysis of spatial kinematically redundant robots. The most common method of this third category is the Monte Carlo method [6]. In the Monte Carlo method, a wide number of random robot configurations is generated to determine the robot workspace. However, this method has some drawbacks [2, 5]. The generated workspace can be inaccurate, especially on its boundaries [2], and the real workspace can be different from the one obtained [5]. The coordinate transformation from joint space to workspace in the forward kinematics is nonlinear. This

means that a uniform distribution of coordinate in the joint space does not necessarily lead to a uniform distribution of points in the Cartesian workspace [7]. As a result, some areas in the workspace have a low density of points and other areas have a high density. Unfortunately, the low density of points areas usually corresponds to the boundaries of the workspace [7]. On the contrary, workspace inner areas are characterized by a high density of points, which leads to high computation time.

In [5], the authors employed an improved version of the Monte Carlo method to calculate the robot workspace, called Gaussian Growth. The method consists in generating an amount of starting robot configurations using the classical Monte Carlo method to obtain a seed workspace, which will be inaccurate in some areas. Then, this seed is grown employing a Gaussian, or normal, random distribution, until the workspace is accurately approximated. A similar approach was employed in [8]. In [9], the authors presented another Monte Carlo method combined with the Gaussian distribution plus a Voxel algorithm [10] to analyze the workspace of a nine degrees of freedom kinematic redundant robot. The obtained seed workspace is expanded by setting an accuracy threshold to accurately describe each region. Then, a Voxel algorithm is proposed to compute the volume of the obtained workspace. All these methodologies based on an improved Monte Carlo method are able to identify the manipulator workspace. However, they are still affected by the drawbacks of the Monte Carlo method although to a lesser extent. To describe the workspace of a robot and obtain its volume, computing only the workspace boundaries is enough, but this is not possible with a Monte Carlo based method.

There exist other types of workspace generation algorithm that employs the workspace density and the N-dimensional Euclidean motion group $SE(N)$. In [11], the authors formulate the workspace generation problem for kinematically redundant robots as a diffusion process employing the Euclidean motion group $SE(N)$ to describe the evolution of the workspace density function. The workspace density is a powerful tool in the case of planar serial arms with revolute joints, as shown in [12]. The workspace density based approach can also be used for plotting the reachability map of special situations such as in the case of ball joints [13]. The approach described in [14] is a step forward in the use of the Euclidean motion group $SE(N)$ for the generation of a three dimensional workspace. The authors implement a series of convolutions to reduce computational complexity from a spatial case to a planar one. All these techniques can describe the reachability map, or workspace, of different types of robots. However, their development is complex and hardly intuitive.

The workspace determination algorithm described in this paper is a ray-based method. The ray-based method idea is presented in [15–17] to determine the interference free and wrench closure workspace and for the trajectory verification of cable-driven robots. Compared to other numerical approaches like pointwise or interval-based analysis, the ray-based approaches provide information about the interference free workspace continuity, precisely determining interior regions [17]. Moreover, ray-based methods decrease the computational time with respect to the other approaches [15]. The workspace determination method proposed in this paper takes into account more complex robotic architectures. It is developed for the determination of the whole workspace of redundant manipulators, which are complex to analyze. However, it can be employed with any type of robot. This workspace determination method is simple to develop, avoids the computational redundancy that affects the Monte Carlo based methods and identifies only the boundaries of the robot workspace. The resulting workspace is easy to visualize and obtained in a small amount of time with continuous boundaries. Starting from a set of configurations inside the workspace, the end-effector is moved from its position along several radial directions. When the end-effector reaches the boundary of the workspace and stops, its position is saved. So, this new method is able to quickly identify the boundaries of any robot workspace. In this case, the robots employed to test the workspace determination algorithm are highly kinematic redundant robots [18]. They are perfect candidates as describing their workspace is a complex problem [2].

The paper is outlined as follows. Section 2 introduces the background of the algorithm, namely the kinematic control algorithm used to control the robot and the employed optimization tasks. Section 3 describes the workspace determination method proposed in this paper. Section 4 presents the family of kinematic redundant robots employed to test the new method. Section 5 analyzes the obtained results testing the proposed workspace determination algorithm and compares the performance with the one of a Monte Carlo method. Conclusions and future work are given in section 6.

2 BACKGROUND

This section describes the kinematic control algorithm and optimization tasks employed by the workspace determination algorithm. The chosen kinematic control algorithm is developed to exploit the redundancy of robotic machines and solve several tasks simultaneously. Firstly, the kinematic control algorithm is introduced. Then, two kinematic optimization tasks are described. These tasks prevent the robot from reaching singular configurations where the end-effector is not at the boundary of the workspace.

2.1 Kinematic Control Algorithm

The employed kinematic control algorithm is called Task Priority Inverse Kinematic (TPIK) and is presented in [19–22]. It can find the robot configuration that solves a set of tasks with different priority levels. Moreover, it can activate and deactivate one or more tasks without generating algorithmic discontinuities [19]

to avoid over-constraining the robotic system.

Here, some general definitions are given to briefly describe the TPIK algorithm. This paper deals with serial robotic manipulators, so, the vector named $\mathbf{q} \in \mathbb{R}^n$ represents the joint position vector that describes the arm configuration. The variable n is the number of joints contained in the robot. The joint velocities are grouped in the vector $\dot{\mathbf{q}} \in \mathbb{R}^n$. A control objective is defined to represent one of the robot goals and the associated task state. It corresponds to a scalar variable $x(\mathbf{q})$ computed as a function of the robot configuration $\mathbf{q}$. There exist two types of control objectives, equality and inequality. More details are given in [22]. Each control objective is associated with a feedback reference rate $\dot{\bar{x}}$ that drives $x(\mathbf{q})$ to the desired point x^* with the linked rate $\dot{x}^*$. Each control objective $x(\mathbf{q})$ has an activation function $a^i(x) \in [0,1]$, which states if the control objective is applicable or not at a specific time instant. Finally, a priority level is assigned to each task based on its control objective importance. The tasks with highest priorities are solved first employing the necessary robot degrees of freedom. Then, lower priority tasks are solved if enough degrees of freedom remains. These concepts were fully described in [22]

With the previous definitions, the control action $\mathscr{A}$ taken as input by the TPIK algorithm can be defined as a series of priority levels composed of [22]:

$\dot{\bar{\mathbf{x}}}_k = [\dot{\bar{x}}_{1,k},\ \dot{\bar{x}}_{2,k},\ \ldots\ ,\ \dot{\bar{x}}_{m_k,k}]^\top$ is the vector of all the scalar control objective reference rates, where m_k is the number of control objectives for the k^{th} priority level.

$\mathbf{J}_k$ is the Jacobian matrix associated with the vector $[\dot{x}_{1,k},\ \ldots\ ,\ \dot{x}_{m_k,k}]^\top$ with respect to the joint velocities $\dot{\mathbf{q}}$ for the k^{th} priority level.

$\mathbf{A}_k = \text{diag}(a_{1,k},\ \ldots\ ,\ a_{m_k,k})$ is the activation function diagonal matrix for the k^{th} priority level.

To find the system velocity reference vector $\dot{\bar{\mathbf{q}}}$ that meets all the action priority requirements, the TPIK algorithm solves a sequence of nested minimization problems

$$S_k = \arg\,\mathrm{R}-\min_{\dot{\bar{\mathbf{q}}} \in S_{k-1}} ||\mathbf{A}_k(\dot{\bar{\mathbf{x}}}_k - \mathbf{J}_k\dot{\bar{\mathbf{q}}})||^2, \tag{1}$$

where S_{k-1} is the manifold of all the previous priority level solutions. The solution to the k^{th} priority level is searched in the manifold of all the previous priority level solutions. The notation $\mathrm{R}-\min$ highlights that each minimization is performed through specific regularized space projections to implement priorities among the tasks defined in [19]. Moreover, the TPIK algorithm uses regularized space projection to implement priorities among all the tasks. One of the main advantages of the TPIK algorithm is the use of the activation functions for inequality control objectives to avoid blocking degrees of freedom that could be used by lower priority level tasks. The complete explanation of the TPIK algorithm solution mechanism can be found in [19].

2.2 Optimization tasks

To perform the ray-based workspace determination, the TPIK algorithm employs a task related to the robot end-effector velocity. However, the TPIK algorithm also includes two kinematic optimization tasks to avoid blocking in singular configurations. These task are based on the kinematic indices: dexterity and manipulability. These indices are related to the kinematic Jacobian matrix $\mathbf{J}_e$ for the robot end-effector velocity

$$\mathbf{t} = \begin{bmatrix} \dot{\mathbf{p}} \\ \boldsymbol{\omega} \end{bmatrix} = \mathbf{J}_e(\mathbf{q})\dot{\mathbf{q}} = \begin{bmatrix} \mathbf{J}_l(\mathbf{q}) \\ \mathbf{J}_a(\mathbf{q}) \end{bmatrix} \dot{\mathbf{q}}, \tag{2}$$

where $\mathbf{t} = \left[\dot{\mathbf{p}}^\top, \boldsymbol{\omega}^\top\right]^\top \in \mathbb{R}^6$ is the robot end-effector twist, with $\dot{\mathbf{p}} \in \mathbb{R}^3$ and $\boldsymbol{\omega} \in \mathbb{R}^3$ the linear and angular velocity vectors of the end-effector, respectively. Since the kinematic Jacobian matrix $\mathbf{J}_e$ contains non-homogeneous terms, i.e. linear and angular ones, it needs to be weighted before computing the kinematic performance indices. This weighing is done using the characteristic length L introduced in [23] to solve the absence of dimensional homogeneity. The computation of L is proposed in [24]. Then, the rows associated to the linear velocities of the end-effector in $\mathbf{J}_e$ are divided by L for the revolute joints. The weighted kinematic Jacobian matrix is written as $\mathbf{J}_w$.

Dexterity The dexterity $\eta(\mathbf{J}_w)$ characterizes the kinematic performance of a manipulator in a given configuration [25]. In [22], the analytical expression of η is computed using the Frobenius norm of $\mathbf{J}_w$

$$\eta(\mathbf{J}_w) = \frac{m}{\gamma_1(\mathbf{J}_w)\,\gamma_2(\mathbf{J}_w)}, \tag{3}$$

where m, which represents the number of rows of $\mathbf{J}_w$, is the dimension of the task space and

$$\gamma_1 = \sqrt{\text{trace}(\mathbf{J}_w\mathbf{J}_w^\top)} \text{ and } \gamma_2 = \sqrt{\text{trace}[(\mathbf{J}_w\mathbf{J}_w^\top)^{-1}]}. \tag{4}$$

The index η is bounded between 0 and 1. The higher η, the better the robot dexterity. The robot reaches an isotropic posture when $\eta = 1$. The smaller η, the worse the robot dexterity and the closer to a singularity. Moreover, η can be defined as the ratio between the smallest and the highest singular values of $\mathbf{J}_w$ indicating how close the manipulability hyper-ellipsoid is to being a hyper-sphere [26]. The derivative of the dexterity as a function of the joint variables is described in [22],

$$\frac{\partial \eta}{\partial q_i} = -\eta\left(\frac{\partial \gamma_1}{\partial q_i}\frac{1}{\gamma_1} + \frac{1}{\gamma_2}\frac{\partial \gamma_2}{\partial q_i}\right), \tag{5}$$

where

$$\begin{cases} \frac{\partial \gamma_1}{\partial q_i} = \frac{1}{\gamma_1} \text{trace} \left\{ \mathbf{J}_w \frac{\partial \mathbf{J}_w^\top}{\partial q_i} \right\}, \\ \frac{\partial \gamma_2}{\partial q_i} = \frac{1}{\gamma_2} \text{trace} \left\{ -\mathbf{J}_w \frac{\partial \mathbf{J}_w^\top}{\partial q_i} (\mathbf{J}_w \mathbf{J}_w^\top)^2 \right\}. \end{cases} \tag{6}$$

The dexterity Jacobian matrix $\mathbf{J}_\eta$ as a function of the joint variables is

$$\mathbf{J}_\eta = \left[\frac{\partial \eta}{\partial q_1} \ \cdots \ \frac{\partial \eta}{\partial q_n} \right]. \tag{7}$$

Manipulability The manipulability is an index that measures the kinematic abilities of the robotic system through its weighted Jacobian matrix $\mathbf{J}_w$ [27]. The manipulability of a manipulator is defined as

$$\mu = \sqrt{\det(\mathbf{J}_w \mathbf{J}_w^\top)}, \tag{8}$$

and amounts to the product of all the singular values of $\mathbf{J}_w$. The higher the manipulability value, the larger the manipulability hyper-ellipsoid and the better the kinematic performance of the mechanism [28]. It should be noted that the manipulator reaches a kinematic singularity when μ reaches zero. The derivative of the manipulability as a function of the joint variables is explained in [29] and used in [30]:

$$\frac{\partial \mu}{\partial q_i} = \mu \ \text{trace} \left\{ \frac{\partial \mathbf{J}_w}{\partial q_i} \mathbf{J}_w^+ \right\}. \tag{9}$$

The manipulability Jacobian matrix $\mathbf{J}_\mu$ as a function of the joint variables is

$$\mathbf{J}_\mu = \left[\frac{\partial \mu}{\partial q_1} \ \cdots \ \frac{\partial \mu}{\partial q_n} \right]. \tag{10}$$

3 WORKSPACE DETERMINATION METHOD

This section describes the workspace determination procedure proposed and applied in this paper. The core idea is to identify the workspace boundaries rapidly without losing time collecting points inside it. After initializing the robot in a starting configuration, the end-effector is moved along a set of linear displacement vectors with a target twist $\mathbf{t}_t$ employing the TPIK algorithm. These vectors radiate from the end-effector initial position along different directions. To avoid the robot getting stuck in a singular configuration inside the workspace, two tasks for the kinematic optimization, based on dexterity η and manipulability μ, are included in the TPIK algorithm. If the rates of η and μ, i.e. $\dot{\eta}$ and $\dot{\mu}$, are over the desired threshold $\varepsilon_{\text{kinematic}}$, it means that the TPIK algorithm is still optimizing the robot configuration and the end-effector position is not saved.

Then, when $\dot{\eta}$ and $\dot{\mu}$ are lower than $\varepsilon_{\text{kinematic}}$ and the 2-norm of the end-effector velocity $||\mathbf{t}||_2$ goes under a selected threshold $\varepsilon_{\text{velocity}}$, it means that the end-effector has reached the workspace boundary and its position is saved. Afterwards, the robot is set again to the starting configuration and the end-effector is moved along another vector. After the end-effector has been moved along all the displacement vectors, the robot is initialized in a new starting configuration and moved along all the vectors again. This process is repeated several times starting from different configurations to obtain enough points to describe the boundary of the whole workspace. Algorithm 1 sums up the workspace determination procedure. Table 1 collects the tasks employed by the TPIK algorithm for the workspace generation and their hierarchical priority levels. The algorithm is developed in C++.

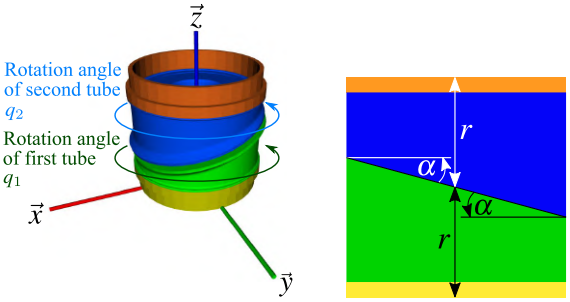
ALGORITHM 1. WORKSPACE DETERMINATION

Require: A number of starting configurations and a set of displacement vectors along different directions. The thresholds $\varepsilon_{\text{kinematic}}$ and $\varepsilon_{\text{velocity}}$ for $\dot{\eta}$ and $\dot{\mu}$ and end-effector twist $\mathbf{t}$, respectively. The target end-effector twist $\mathbf{t}_t$.

```
for i := 1 → number of starting configurations do
    for k := 1 → number of displacement vectors do
        Initialize robot in i^th configuration.
        while η̇ and μ̇ > ε_kinematic do
            while End-effector twist ||t||_2 > ε_velocity do
                Move end-effector along k^th vector with t_t.
            end while
        end while
        Save end-effector position.
    end for
end for
```

TABLE 1. DETAILS ABOUT TASK NAME, CATEGORY, TYPE AND HIERARCHY LEVEL IN END-EFFECTOR VELOCITY ACTION $\mathscr{A}$, SYMBOL (E) FOR EQUALITY CONTROL OBJECTIVE AND (I) FOR INEQUALITY.

Task name	Category	Type	$\mathscr{A}$
End-Effector Velocity	action oriented	E	1$^{\text{nd}}$
Dexterity	optimization	I	2$^{\text{rd}}$
Manipulability	optimization	I	2$^{\text{rd}}$



(a) 3D view and rotation angles (b) Front view

FIGURE 1. NB-MODULE REPRESENTATION IN HOME POSE WITH ROTATION ANGLES [22]

4 FAMILY OF ROBOTS UNDER STUDY

This section proposes the family of robots employed to test the workspace determination algorithm. First, the mechanism that composes and actuates the robots is described. Then, the three robots under analysis are presented.

4.1 Mechanism description

The mechanism, called NB-module, developed and patented by the company Nimbl'Bot [31] is used as a case study in this paper. Here, the NB-module is presented. It is formed of two closed kinematic chains, one internal and one external, actuated by two motors. So, it is a two degrees of freedom mechanism. The internal kinematic chain is passive and ensures strength of the entire design. The external kinematic chain is the active one, actuated by the two motors and shown Fig. 1a. It is composed of the yellow and orange platforms and the blue and green hollow cylinders, both cut by an oblique plane. The green tube rotates around the vertical axis of the yellow platform when it is actuated by the first motor. The angle of the first motor is called q_1. The blue tube rotates around the vertical axis of the orange platform when it is actuated by the second motor. The angle of the second motor is called q_2. The rotation of these two tubes is completely independent and continuous. Figure 1 shows the design parameters of the NB-module. The variable r indicates half height of the NB-module, set to 0.07 m for the rest of this paper. The variable α represents the slope of the oblique plane that cuts the two tubes, set to 15° for the rest of this paper. For more details, the NB-modules is described in [22, 32].

4.2 Description of the robots under analysis

A number of NB-modules can be attached serially to generate different kinematic redundant manipulators. In this paper, the workspace of three different robots is analyzed. The first robot, called NB-R1, is shown in Fig. 2. It is composed of six NB-modules serially attached and has twelve degrees of freedom. The second robot, shown in Fig. 3, is called NB-R2. It has ten NB-modules and can be divided in three main regions: the shoulder (three NB-modules), the elbow (four NB-modules) and the wrist (three NB-modules). In total, it has twenty degrees of freedom. These three regions are connected by two links l_1 and l_2 of length equal to 0.2 m. The third robot, represented in Fig. 4, is organized as the second one, i.e., with ten NB-modules and two links. It is called NB-R3. So, it has twenty degrees of freedom too. However, two angular offsets β_1 and β_2 are inserted between the link l_1 and the first elbow NB-module and between the second link l_2 and the first wrist NB-module. The length of both the links l_1 and l_2 is equal to 0.2 m and the the offsets β_1 and β_2 equal to -45°. Table 2 summarizes the robot dimensions.

TABLE 2. ROBOT DESIGN DETAILS

Design parameter	Value
NB-module half height r	0.07 m
NB-module slope α	15°
Link lengths l_1 and l_2	0.2 m
Offsets β_1 and β_2	-45°

5 RESULTS AND DISCUSSION

This section describes the test to determine the robots workspace and discusses the obtained results. Firstly, the main features of the test are presented. Then, the results are analysed.

5.1 Test description

Before applying the workspace determination algorithm, a rule to initialize the robot has to be defined. In this case, a uniform random generation is employed to obtain the starting configurations. Considering one NB-module, the values of the motors position q_1 and q_2 are randomly generated. These values are applied to all the NB-modules included in one robot. So, each NB-module in the robot is initialized in the same configuration. After several tests, this type of initialization was selected because it generated the most uniform point distribution on the workspace boundaries. If all the NB-modules are randomly initialized with different values, the generated points were not uniformly distributed.

Then, as described in Section 3, the robot end-effector is moved along a set of displacement vectors, in total 14 unit vectors. The first six vectors are oriented along the positive and negative direction of axes $\vec{x}$, $\vec{y}$ and $\vec{z}$, respectively,

$$\vec{v}_{1,2} = \begin{bmatrix} \pm 1 \\ 0 \\ 0 \end{bmatrix}, \ \vec{v}_{3,4} = \begin{bmatrix} 0 \\ \pm 1 \\ 0 \end{bmatrix}, \ \vec{v}_{5,6} = \begin{bmatrix} 0 \\ 0 \\ \pm 1 \end{bmatrix}. \qquad (11)$$

FIGURE 2. ROBOT NB-R1 FORMED OF SIX NB-MODULES

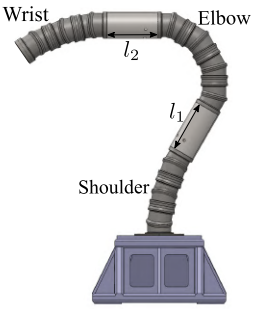


FIGURE 3. ROBOT NB-R2 FORMED OF TEN NB-MODULES PLUS TWO LINKS l_1 AND l_2

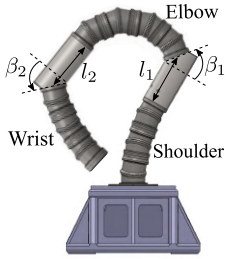


FIGURE 4. ROBOT NB-R3 FORMED OF TEN NB-MODULES PLUS TWO LINKS l_1 AND l_2 AND TWO OFFSETS β_1 AND β_2

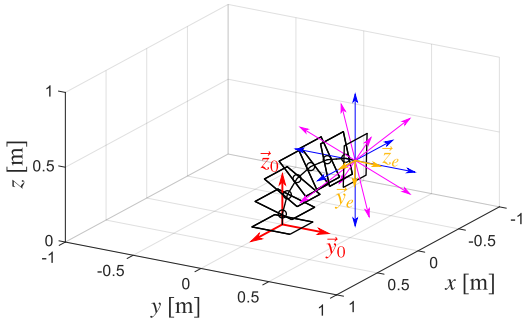


FIGURE 5. REPRESENTATION OF THE NB-R1 AND DISPLACEMENT VECTORS APPLIED TO THE ROBOT END-EFFECTOR POSITION. $\mathscr{F}_0$ IS THE BASE FRAME AND $\mathscr{F}_E$ IS THE END-EFFCTOR FRAME.

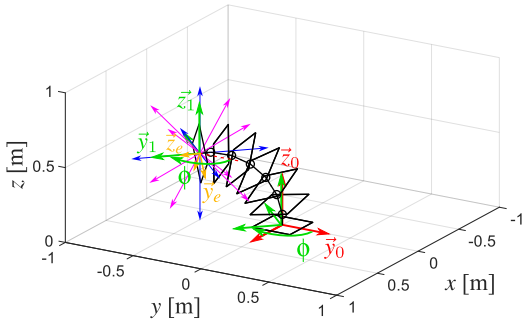


FIGURE 6. REPRESENTATION OF THE NB-R1 WITH $\phi = 135°$ AND DISPLACEMENT VECTORS APPLIED TO THE END-EFFECTOR POSITION ROTATED OF THE SAME ϕ. $\mathscr{F}_0$ IS THE BASE FRAME, $\mathscr{F}_1$ IS THE BASE FRAME ROTATED OF ϕ AROUND $z_0 \equiv z_1$ AND $\mathscr{F}_E$ IS THE END-EFFECTOR FRAME.

TABLE 3. IMPLEMENTATION DETAILS

Variable	Value
$\varepsilon_{\text{kinematic}}$	10^{-6}
$\varepsilon_{\text{velocity}}$	10^{-3} m/s
$\mathbf{t}_t$	0.14 m/s

The other eight vectors are a combination of displacements along all the axis $\vec{x}$, $\vec{y}$ and $\vec{z}$,

$$\vec{v}_7 = \begin{bmatrix} -c \\ -c \\ -c \end{bmatrix}, \ \vec{v}_8 = \begin{bmatrix} c \\ -c \\ -c \end{bmatrix}, \ \vec{v}_9 = \begin{bmatrix} -c \\ c \\ -c \end{bmatrix}, \ \vec{v}_{10} = \begin{bmatrix} c \\ c \\ -c \end{bmatrix}, \\ \vec{v}_{11} = \begin{bmatrix} -c \\ -c \\ c \end{bmatrix}, \ \vec{v}_{12} = \begin{bmatrix} c \\ -c \\ c \end{bmatrix}, \ \vec{v}_{13} = \begin{bmatrix} -c \\ c \\ c \end{bmatrix}, \ \vec{v}_{14} = \begin{bmatrix} c \\ c \\ c \end{bmatrix}, \tag{12}$$

where c is equal to $1/\sqrt{3}$. The displacement vectors applied to the end-effector are shown in Fig. 5. To ensure a uniform point distribution on the whole workspace, the displacement vectors are rotated around the $\vec{z}_1$ axis oriented as the base frame $\vec{z}_0$ axis and applied to the origin of the end-effector frame, as shown in Fig. 6. The rotation angle is equal to the azimuth angle ϕ of the transformation matrix between the base frame to the end-effector frame. The azimuth angle ϕ can be obtained using the notation presented in [32]. Table 3 provides the test implementation details and values.

5.2 Results analysis

The total number of generated points to determine one workspace is 8064. Figure 7 shows three views of the NB-R1 workspace. The colors to the points are assigned on the base of their height along axis z and help the visual identification of the workspace. The graph on the left shows the vertical section, defined by the yz-plane, of the NB-R1 workspace. Then on the right, the upper graph shows a 3D view of the complete NB-R1 workspace and the lower one presents a 3D view of the inner

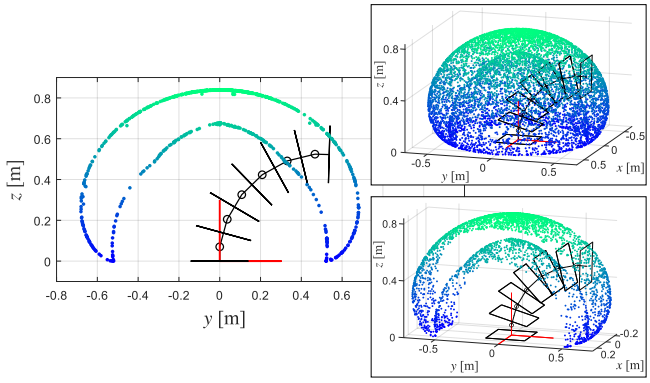


FIGURE 7. GENERATED POINTS DESCRIBING NB-R1 WORKSPACE. GRAPH ON LEFT SHOWS VERTICAL SECTION OF WORKSPACE IN *yz*-PLANE. GRAPH ON UPPER RIGHT SHOWS COMPLETE WORKSPACE. GRAPH ON LOWER RIGHT SHOWS INTERNAL BOUNDARY VIEW.

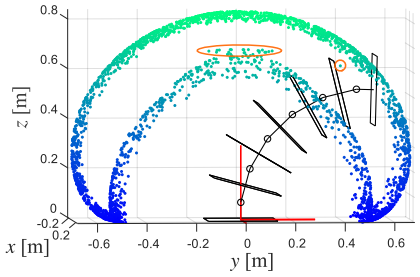


FIGURE 8. INTERNAL BOUNDARY VIEW OF POINTS DESCRIBING NB-R1 WORKSPACE OBTAINED WITHOUT KINEMATIC OPTIMIZATION. ORANGE LINES IDENTIFY POINTS REMAINED STUCK INSIDE WORKSPACE.

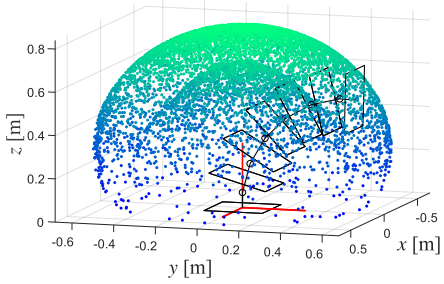


FIGURE 9. POINTS DESCRIBING NB-R1 WORKSPACE OBTAINED STARTING ROBOT JOINTS WITH ALL DIFFERENT RANDOM VALUES

boundaries of the workspace. The NB-R1 workspace boundaries are clearly identified and uniformly distributed. This workspace is symmetric around the z axis. As previously described, to move the robot end-effector until the boundaries, the TPIK algorithm

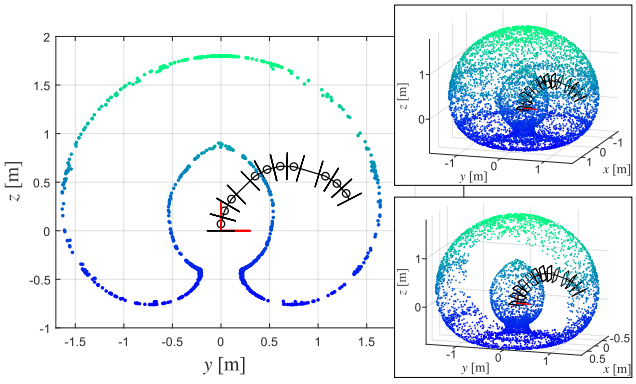


FIGURE 10. GENERATED POINTS DESCRIBING NB-R2 WORKSPACE. GRAPH ON LEFT SHOWS VERTICAL SECTION OF WORKSPACE IN *yz*-PLANE. GRAPH ON UPPER RIGHT SHOWS COMPLETE WORKSPACE. GRAPH ON LOWER RIGHT SHOWS INTERNAL BOUNDARY VIEW.

employs two kinematic optimization tasks to ameliorate the robot kinematic performance and avoid the robot getting stuck in singular configurations. Figure 8 shows the generated points for the NB-R1 workspace deactivating the optimization tasks. The orange lines highlight the points that stopped inside the workspace. The majority of these points are close to the upper area of the internal boundaries. Here, the robot almost reached the workspace boundaries but stopped in a singular configuration. Figure 9 shows the workspace of the NB-R1 when the robot starting configurations are generated initializing all the joints with different random values. As a result, few points describe the workspace lower part, being more concentrated in the upper part. This phenomenon proves the validity of the robot configuration initialization chosen in this paper.

Figure 10 shows three views of the NB-R2 workspace. The graph on the left shows the vertical section, defined by the *yz*-plane, of the NB-R2 workspace. Then on the right, the upper graph shows a 3D view of the complete NB-R2 workspace and the lower one presents a 3D view of the inner boundaries of the workspace. Again, the NB-R2 workspace is clearly identified and symmetric around the z axis. In this case, the total workspace is bigger than the NB-R1 case thank to the higher number of NB-modules and the presence of the two links l_1 and l_2. Finally, Fig. 11 shows three views of the NB-R3 workspace. The graph on the left shows the vertical section, defined by the *yz*-plane, of the NB-R3 workspace. Then on the right, the upper graph shows a 3D view of the complete NB-R3 workspace and the lower one presents a 3D view of the inner boundaries of the workspace. The NB-R3 workspace has a similar same volume of the NB-R2 one since both robots have the same number of module and link lengths. However, the NB-R3 workspace is not symmetric as the two previous cases because of the two offsets β_1 and β_2. The workspace distribution is moved along the positive

TABLE 4. RESULTS FOR RAY-BASED AND MONTE CARLO METHODS IN WORKSPACE DETERMINATION

Robot	Method	Number of points	Total computational time	Result quality
NB-R1	Ray-based	8064	2.1 min	All boundaries detected
	Monte Carlo	500,000	25 min	Lower and internal boundaries missing
NB-R2	Ray-based	8064	9.3 min	All boundaries detected
	Monte Carlo	500,000	26 min	Lower and internal boundaries missing
NB-R3	Ray-based	8064	7.0 min	All boundaries detected
	Monte Carlo	500,000	26 min	Lower and internal boundaries missing

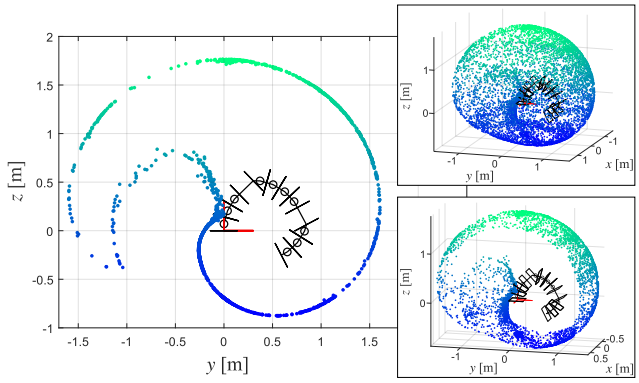


FIGURE 11. GENERATED POINTS DESCRIBING NB-R3 WORKSPACE. GRAPH ON LEFT SHOWS VERTICAL SECTION OF WORKSPACE IN *yz*-PLANE. GRAPH ON UPPER RIGHT SHOWS COMPLETE WORKSPACE. GRAPH ON LOWER RIGHT SHOWS INTERNAL BOUNDARY VIEW.

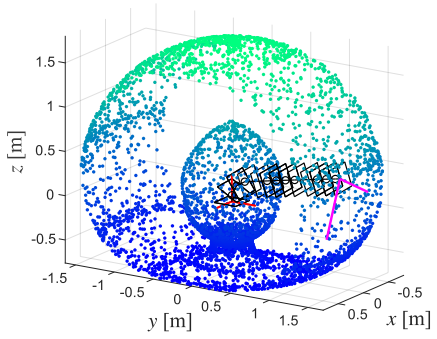


FIGURE 12. ROBOT CONFIGURATION, BLACK, AND LINEAR JACOBIAN MATRIX SINGULAR VECTORS, MAGENTA, FOR NB-R2 IN ONE POINT ON ITS WORKSPACE BOUNDARY

side of axis y since β_1 and β_2 are rotated about the axis x of a negative value. The workspace maintains its symmetry with respect to the yz-plane since there is no offset about the axis y. The NB-R3 workspace boundaries are globally identified. Nevertheless, in this case, the inner boundaries are rougher and not perfectly described. This happens because, despite the presence of the kinematic optimization tasks, the robot reaches singular configurations and stops. This behavior does not appear with the other robots and it is probably due to the presence of β_1 and β_2. More analysis will be done on this point in the future. The behaviors shown in Figs. 8 and 9 were seen also in the case of the robots NB-R2 and NB-R3. However, it is not reported in this paper for the sake of space. Table 4 provides the total time to determine the workspace of the three analyzed robots using the ray-based method.

Figure 12 shows the robot configuration, black, and the linear Jacobian matrix singular vectors, magenta, for the NB-R2 robot in one point on its workspace boundary. The magnitude of the singular vectors tangent to the workspace boundary is different from zero, while the magnitude of the singular vector aligned to the robot configuration is zero. This means that the robot has lost one degree of freedom in the linear task space reaching a singular configuration. The end-effector can no longer move in the zero singular vector direction since it has reached the workspace boundary. The magnitude of at least one singular vector is always equal to zero on the points that compose the workspace boundaries of NB-R1, NB-R2 and NB-R3.

Finally, the performance of the method presented in this paper is compared to the Monte Carlo one. Figure 13 shows the NB-R2 workspace generated using the Monte Carlo method, i.e. generating random configurations. In the Monte Carlo case, $500,000$ random configurations were generated in 26 minutes. The upper external boundaries were correctly identified. However, the lower and internal boundaries are completely ignored and the real robot workspace can not be reconstructed. On the contrary, the proposed ray-based workspace determination algorithm can describe the correct workspace of NB-R2 in less than ten minutes using only 8064 of points. Table 4 compares the results of the ray-based and Monte Carlo methods for each robot. In all the cases, the ray-based performance are better. The computational time of the three robots using the Monte Carlo method

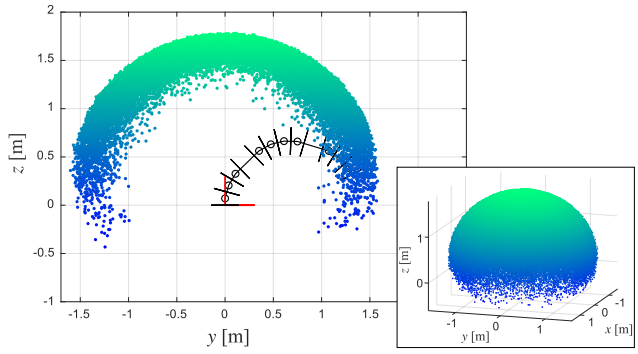


FIGURE 13. POINTS DESCRIBING NB-R2 WORKSPACE GENERATED THROUGH MONTE CARLO METHOD. GRAPH ON LEFT SHOWS VERTICAL SECTION OF WORKSPACE IN *yz*-PLANE. GRAPH ON LOWER RIGHT SHOWS COMPLETE WORKSPACE.

is more or less the same since the process only requires generating random robot configurations and saving the end-effector positions. The NB-R1 and NB-R3 workspace graphs generated with the Monte Carlo method are not reported in this paper for the sake of space.

6 CONCLUSIONS

This paper presented a new algorithm for the workspace determination of robotic manipulators. The workspace determination process was evaluated on three kinematic redundant robots. However, it can be applied also to non-redundant manipulators. The proposed methodology is simple to develop and fast to run, each workspace took less than 10 minutes. It is not affected by computational redundancy, like Monte Carlo based methods, and identifies only the workspace boundaries. The performed tests emphasized the process ability to detect the complete workspace boundaries in a small amount of time. It always took less then ten minutes to produce one workspace. For the smaller robot, e.g. NB-R1, the process lasted almost two minutes. Moreover, the use of the kinematic optimization tasks allowed maintaining better kinematic configurations while moving and avoided getting stuck the manipulator in singular configurations inside the workspace. In the NB-R3 robot case, the generated map identifies the workspace inner boundaries with less accuracy. However, the workspace shape is perfectly identifiable from the obtained results. Furthermore, it is complex to automatize the identification of points that are contained or not inside the computed workspace, although it is easy visually. Finally, the results were compared with the ones obtained with a Monte Carlo method. The proposed ray-based workspace determination algorithm is faster, more accurate and requires less points than the Monte Carlo one.

Future work will address the workspace determination for a specific end-effector orientation. This is important when planning the workpiece placement inside of the robot reachable area. Then, a workspace volume computation will be developed starting from the obtained boundaries to rate the robot abilities. Moreover, it will be necessary to develop a process for the interpolation of all the generated points in a surface. So, identifying the point reachability for the analyzed robot will be a faster process.

ACKNOWLEDGMENT

This work was supported by the ANRT (Association nationale de la recherche et de la technologie) grant CIFRE n° 2020/1051 and Nimbl'bot (`https://nimbl-bot.com/`).

REFERENCES

[1] Cao, Y., Lu, K., Li, X., and Zang, Y., 2011. "Accurate numerical methods for computing 2d and 3d robot workspace". *International Journal of Advanced Robotic Systems,* ***8***(6), p. 76.

[2] Du, Z.-c., Ouyang, G.-Y., Xue, J., and Yao, Y.-b., 2020. "A review on kinematic, workspace, trajectory planning and path planning of hyper-redundant manipulators". In 2020 10th Institute of Electrical and Electronics Engineers International Conference on Cyber Technology in Automation, Control, and Intelligent Systems (CYBER), IEEE, pp. 444–449.

[3] Yahya, S., Moghavvemi, M., and Mohamed, H. A., 2011. "Geometrical approach of planar hyper-redundant manipulators: Inverse kinematics, path planning and workspace". *Simulation Modelling Practice and Theory,* ***19***(1), pp. 406–422.

[4] Bohigas, O., Manubens, M., and Ros, L., 2012. "A complete method for workspace boundary determination on general structure manipulators". *IEEE Transactions on Robotics,* ***28***(5), pp. 993–1006.

[5] Peidró, A., Reinoso, Ó., Gil, A., Marín, J. M., and Payá, L., 2017. "An improved monte carlo method based on gaussian growth to calculate the workspace of robots". *Engineering Applications of Artificial Intelligence,* ***64***, pp. 197–207.

[6] Guan, Y., and Yokoi, K., 2006. "Reachable space generation of a humanoid robot using the monte carlo method". In 2006 IEEE/RSJ International Conference on Intelligent Robots and Systems, IEEE, pp. 1984–1989.

[7] Li, L., Shang, J., Feng, Y., and Yawen, H., 2018. "Research of trajectory planning for articulated industrial robot: a review". *Computer engineering and applications,* ***54***(5), pp. 36–50.

[8] He, B., Zhu, X., and Zhang, D., 2020. "Boundary encryption-based monte carlo learning method for

workspace modeling". *Journal of Computing and Information Science in Engineering,* ***20***(3).

[9] Zhao, Z., He, S., Zhao, Y., Xu, C., Wu, Q., and Xu, Z., 2018. "Workspace analysis for a 9-dof hyper-redundant manipulator based on an improved monte carlo method and voxel algorithm". In 2018 IEEE International Conference on Mechatronics and Automation (ICMA), IEEE, pp. 637–642.

[10] Fernández-Sarría, A., Martínez, L., Velázquez-Martí, B., Sajdak, M., Estornell, J., and Recio, J., 2013. "Different methodologies for calculating crown volumes of platanus hispanica trees using terrestrial laser scanner and a comparison with classical dendrometric measurements". *Computers and electronics in agriculture,* ***90***, pp. 176–185.

[11] Wang, Y., and Chirikjian, G. S., 2004. "Workspace generation of hyper-redundant manipulators as a diffusion process on se (n)". *IEEE Transactions on Robotics and Automation,* ***20***(3), pp. 399–408.

[12] Dong, H., Du, Z., and Chirikjian, G. S., 2013. "Workspace density and inverse kinematics for planar serial revolute manipulators". *Mechanism and Machine Theory,* ***70***, pp. 508–522.

[13] Dong, H., Fan, T., Du, Z., and Chirikjian, G., 2015. "Inverse kinematics of discretely actuated ball-joint manipulators using workspace density". In International Design Engineering Technical Conferences and Computers and Information in Engineering Conference, Vol. 57144, American Society of Mechanical Engineers, p. V05CT08A039.

[14] Han, Y., Pan, J., Xia, M., Zeng, L., and Liu, Y.-J., 2021. "Efficient se (3) reachability map generation via interplanar integration of intra-planar convolutions". In 2021 IEEE International Conference on Robotics and Automation (ICRA), IEEE, pp. 1854–1860.

[15] Abbasnejad, G., Eden, J., and Lau, D., 2019. "Generalized ray-based lattice generation and graph representation of wrench-closure workspace for arbitrary cable-driven robots". *IEEE Transactions on Robotics,* ***35***(1), pp. 147–161.

[16] Zhang, Z., Cheng, H. H., and Lau, D., 2020. "Efficient wrench-closure and interference-free conditions verification for cable-driven parallel robot trajectories using a ray-based method". *IEEE Robotics and Automation Letters,* ***5***(1), pp. 8–15.

[17] Cheng, H. H., and Lau, D., 2022. "Ray-based cable and obstacle interference-free workspace for cable-driven parallel robots". *Mechanism and Machine Theory,* ***172***, p. 104782.

[18] Chirikjian, G. S., and Burdick, J. W., 1994. "A hyper-redundant manipulator". *IEEE Robotics & Automation Magazine,* ***1***(4), pp. 22–29.

[19] Simetti, E., and Casalino, G., 2016. "A novel practical technique to integrate inequality control objectives and task transitions in priority based control". *Journal of Intelligent & Robotic Systems,* ***84***(1-4), pp. 877–902.

[20] Simetti, E., Casalino, G., Aicardi, M., and Wanderlingh, F., 2018. "Task priority control of underwater intervention systems: Theory and applications". *Ocean Engineering,* ***164***, pp. 40–54.

[21] Simetti, E., Casalino, G., Aicardi, M., and Wanderlingh, F., 2019. "A task priority approach to cooperative mobile manipulation: Theory and experiments". *Robotics and Autonomous Systems,* ***122***, p. 103287.

[22] Ginnante, A., Caro, S., Simetti, E., and Leborne, F., 2023. "Kinetostatic optimization for kinematic redundancy planning of nimbl'bot robot". *Journal of Mechanisms and Robotics*, pp. 1–20.

[23] Angeles, J., 1992. "The design of isotropic manipulator architectures in the presence of redundancies". *The International Journal of Robotics Research,* ***11***(3), pp. 196–201.

[24] Khan, W. A., and Angeles, J., 2005. "The kinetostatic optimization of robotic manipulators: The inverse and the direct problems". *Journal of Mechanical Design,* ***128***(1), pp. 168–178.

[25] Angeles, J., and López-Cajún, C. S., 1992. "Kinematic isotropy and the conditioning index of serial robotic manipulators". *The International Journal of Robotics Research,* ***11***(6), pp. 560–571.

[26] Pond, G., and Carretero, J. A., 2006. "Formulating jacobian matrices for the dexterity analysis of parallel manipulators". *Mechanism and Machine Theory,* ***41***(12), pp. 1505–1519.

[27] Yoshikawa, T., 1985. "Manipulability of robotic mechanisms". *The international journal of Robotics Research,* ***4***(2), pp. 3–9.

[28] Angeles, J., 2003. *Fundamentals of robotic mechanical systems: theory, methods, and algorithms*. Springer.

[29] Park, J., 2000. "Analysis and control of kinematically redundant manipulators: An approach based on kinematically decoupled joint space decomposition". *PhD Thesis, POSTECH*.

[30] Marani, G., Kim, J., Yuh, J., and Chung, W. K., 2002. "A real-time approach for singularity avoidance in resolved motion rate control of robotic manipulators". In Proceedings 2002 IEEE International Conference on Robotics and Automation (Cat. No. 02CH37292), Vol. 2, IEEE, pp. 1973–1978.

[31] Dufau, L., March 23, 2021. Articulated robot arm. US patent 10,953,554.

[32] Ginnante, A., Leborne, F., Caro, S., Simetti, E., and Casalino, G., 2021. "Design and kinematic analysis of a novel 2-dof closed-loop mechanism for the actuation of machining robots". In International Design Engineering Technical Conferences and Computers and Information in Engineering Conference, Vol. 85444, American Society of Mechanical Engineers.