Received September 9, 2020, accepted September 21, 2020, date of publication September 23, 2020, date of current version October 5, 2020.

Digital Object Identifier 10.1109/ACCESS.2020.3026193

Artificial Intelligence Control in 4D Cylindrical Space for Industrial Robotic Applications

ANDREA de GIORGIO AND LIHUI WANG
Department of Production Engineering, School of Industrial Engineering and Management, KTH Royal Institute of Technology, 114 28 Stockholm, Sweden
Corresponding author: Andrea de Giorgio (andreadg@kth.se)

ABSTRACT This article argues that an efficient artificial intelligence control algorithm needs the built-in symmetries of an industrial robot manipulator to be further characterized and exploited. The product of this enhancement is a four-dimensional (4D) discrete cylindrical grid space that can directly replace complex robot models. A* is chosen for its wide use among such algorithms to study the advantages and disadvantages of steering the robot manipulator within the 4D cylindrical discrete grid. The study shows that this approach makes it possible to control a robot without any specific knowledge of the robot kinematic and dynamic models at planning and execution time. In fact, the robot joint positions for each grid cell are pre-calculated and stored as knowledge, then quickly retrieved by the pathfinding algorithm when needed. The 4D cylindrical discrete space has both the advantages of the configuration space and the three-dimensional Cartesian workspace of the robot. Since path optimization is the core of any search algorithms, including A*, the 4D cylindrical grid provides for a search space that can embed further knowledge in form of cell properties, including the presence of obstacles and volumetric occupancy of the entire industrial robot body for obstacle avoidance applications. The main trade-off is between a limited capacity for pre-computed grid knowledge and the path search speed. This innovative approach encourages the use of search algorithms for industrial robotic applications, opens up to the study of other robot symmetries present in different robot models and lays a foundation for the application of dynamic obstacle avoidance algorithms.

INDEX TERMS Industrial robot, robot control, algorithm, pathfinding, artificial intelligence, cylindrical symmetry, search space, volumetric information, obstacle avoidance.

I. INTRODUCTION

Beside the many benefits of solving problems in high dimensional spaces, which led to the use of redundant industrial robots [1], [2], the complexity of artificial intelligence path or trajectory planning algorithms increases with control variables dimensionality, as the related search space becomes combinatorically harder to explore completely [3]. The application of path or trajectory planning artificial intelligence algorithms pertains to mobile robots as much as industrial robot manipulators. The first category regards robots that can move across a given environment, e.g. terrain, air, water, etc. The industrial robots are typically characterized by one or multiple moving arms tied to a fixed frame and their body parts can even collide one onto another. Humanoid robots are the union of both cases, as they can walk around and use their arms. Despite the different challenges of modelling any of those robots, artificial intelligence control algorithms can generally be applied to all of them, with minor differences. A mobile robot typically moves on a flat surface with obstacles that can be mapped to a simple two-dimensional (2D) geometrical grid [4]. A three-dimensional (3D) coordinate system is necessarily required for robots that move across a 3D space rather than on a surface, making the treatment of their kinematic and dynamic models harder to solve. This is the case for some classes of mobile robots, for flying robots, e.g. quadcopters [5], and for the majority of industrial robot manipulators [6] whose models have to account for the position of each of their complex body parts.

The necessity of a third dimension to describe the industrial robot movements in Cartesian space has already been challenged by many researchers introducing controls in the high-dimensional configuration space [7]–[11]. However, the advantages of using the configuration space are traded off with a greater difficulty in visualizing and operating in such space, to the point that representing other interacting objects

The associate editor coordinating the review of this manuscript and approving it for publication was Francesco Mercaldo.

in the configuration space became a self-standing topic for study [1], [2], [12]–[16].

It is not uncommon to use algorithms for path or trajectory planning that are based on a trade-off between configuration and workspace search. Among these are rapidly-exploring random tree (RRT) [17] or probabilistic roadmaps [18], [19], none of which aims at characterizing the entire robot space in advance because of the high number of possible robot configurations and spatial positions and the high combinatorial complexity that derives from them. This article shows that such an operation is actually possible in a search space that exploits the industrial robot symmetries and opens up to possible new versions of those algorithms. Because the algorithm proposed in this article does not perform a search in the configuration space of the industrial robot but implements a cylindrical coordinate system thanks to the cylindrical symmetries that are most typical of industrial robots. Furthermore, another dimension is added to the cylindrical space to represent the robot end effector (EE) orientations in a hybrid four-dimensional (4D) cylindrical coordinate system. Geometrical implications of the robot configurations, such as the wrist or elbow positions and orientations, are reflected in the structure of the hybrid 4D cylindrical coordinate system. This can be considered a dimensionality reduction of the robot configuration space and it brings both the advantages of searching in a low dimensional physical space and solving problems related to a specific robot configuration, such as singularities [20].

The paper is structured as follows: In section II, some additional literature review is provided. In section III, typical cylindrical symmetries of industrial robots are explained and characterized in the form of reachability planes. In section IV, the reachability planes are acquired (A) and transformed into a 4D cylindrical discrete search space (B), further data such as robot volumetric occupancy information is added to the cells (C) and a pathfinding algorithm is implemented (D). The path is tracked with an industrial controller (E) and the whole project code and some benchmarking values are provided for future reference (F). In section V, there is a general discussion, covering also the current limitations of the method introduced in this article and its study. In section VI, the conclusions revolve around the advantages and disadvantages of the method. Finally, section VII provides insights for further improvements and perfectly reasonable deviations from the beaten track to explore in future work.

II. ADDITIONAL LITERATURE REVIEW

Common path or trajectory planning algorithms in artificial intelligence are A* [21] and its variations such as improved A* [22], double A* [8], hierarchical A* [23], or Dijkstra and its improved versions [24]. These are just a few examples, out of hundreds. A subset of these and other algorithms are more specifically tied to industrial robots, such as the already named double A* [8], non-probabilistic anytime algorithm [25], potential fields [26], [27], probabilistic roadmaps [18], [19] and RRT [17]. Kallman *et al.* [28] proposed the dynamic roadmaps with bidirectional RRT to deal with changing environments. There are also hybrid versions such a cell-based Voronoi roadmap generation algorithm that is searched with A* [10]. Attempts have been made with reinforcement learning [29], [30]. All of these algorithms are in most of the cases based on high-dimensional configuration spaces, corresponding to the robot joint positions. Few are based on Cartesian coordinate systems corresponding to the robot 2D or 3D workspace. With rare exceptions, such as a polar coordinate system approach to mobile robot path planning over a fan-shaped grid [31], no other radial search spaces seem to have been adopted for modeling the movements of industrial robots.

Llopis-Albert *et al.* [32] presented a comparison of the major search based algorithms for trajectory planning. Search algorithms based on multidimensional spaces are mainly tied to the possibility given by a certain robot model of doing the main calculations in its configuration space, also called joint space, with many degrees of freedom (DOF), e.g. six or seven. Kaltsoukalas *et al.* [33] proposed a search algorithm that generates a grid of alternative points based on similar configurations for the path and leads to a desired position and orientation of the end effector of the robot.

III. CYLDINDRICAL SYMMETRIES OF INDUSTRIAL ROBOTS

The most common industrial robot manipulators are serial manipulators [34], composed of a series of rigid links connected by revolute or prismatic joints that for easiness of control are only orthogonal, parallel or intersecting the joint axes. Often the structure resembles an anthropomorphic arm that can be described by a shoulder, an elbow and a wrist. The preferred number of joints is six because at least six DOF are needed to move the end-effector to an arbitrary position and orientation in space. Four DOF robots are used for simpler manufacturing tasks.

The work that has mostly influenced the production of the common structures of robot manipulators available at present time is Donald L. Pieper's 321 kinematic structure [35] providing the closed-form inverse kinematics solution for serial manipulators with six revolute joints and with three consecutive intersecting joints. The symmetries produced by this solution are further exploited in this article, as shown in this section, up to a discrete search space cylindrical configuration that does not require direct and inverse kinematic calculations for online path or trajectory planning.

A. CYLINDRICAL SYMMETRIES OF ABB IRB 1600

An ABB IRB 1600 is used in this article as an example of a typical 6-DOF industrial robot manipulator. It is common to use the first three revolute joints, J_1, J_2 and J_3, to move the end-effector (EE) in space and another three revolute joints, J_4, J_5 and J_6, in a wrist configuration in order to accurately set the EE orientation. See figure 1 for reference.

The cylindrical symmetry exploited in this article regards robot manipulators with a revolute joint J_1. For the ABB

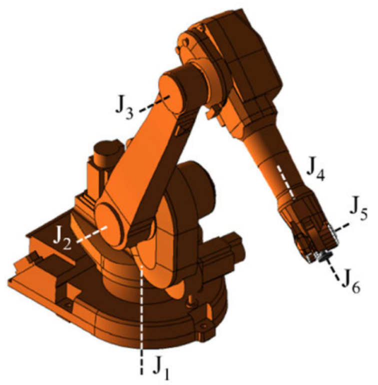

FIGURE 1. 3D view of an ABB IRB 1600 with joints J_2 and J_3 forming an elbow and joints J_4, J_5 and J_6 forming a wrist. Joint J_1 has a key role in producing the industrial robot symmetries exploited in this research. The dashed lines indicate the rotation axes for each joint.

IRB 1600, J_1 can rotate ± 180 degrees from the setup pose. Note that what follows in this article is not applicable if the first joint of a robot manipulator is not of a revolute type. However, other symmetries might apply and this article poses the bases for a broader procedural approach to path or trajectory planning in presence of symmetries.

For the same robot under analysis, J_2 and J_3 are of a revolute type and aligned over the same plane. In particular, J_2 and J_3 jointly function as an elbow for the robot manipulator: a combination of the two angular values leads to any reachable positions in polar coordinates over their plane. Each position can be reached in two configurations, namely with inward or outward elbow configurations. For the purpose of simplifying the demonstration use case of this article, only inward elbow configurations are considered.

J_4, J_5 and J_6 are mostly dedicated to defining the EE orientation, as they can only tweak the EE position, with respect to J_1, J_2 and J_3 that have the greater influence on it.

The cylindrical symmetry is produced by locking the robot EE with a certain orientation on the Cartesian plane xz with $y = 0$ and discretely shifting the EE position over this plane using mostly the shifts of J_2 and J_3 either in the inward or outward configuration. The output of this operation can be called reachability plane for a given EE orientation. All the joint values for each EE position on each reachability plane are thus pre-collected offline. See section IV (A) for more details about how these values are collected for the experimental demo presented in this article.

A knowledge-base is created and filled with all the mapping values between predefined cylindrical coordinates EE positions and orientations and the respective robot joint values. This operation allows to replace the direct and inverse kinematic calculations necessary for the trajectory planning with a space search in discrete cylindrical coordinates.

Some particular J_2-J_3-inward or -outward reachability planes can be obtained by locking J_1, J_4 and J_6 to value zero. For shifting values of J_2, J_3 and J_5, the xz plane with $y = 0$ of the robot Cartesian space can be called CENTRAL, UP and DOWN, based on the wrist configuration. See section IV (A)

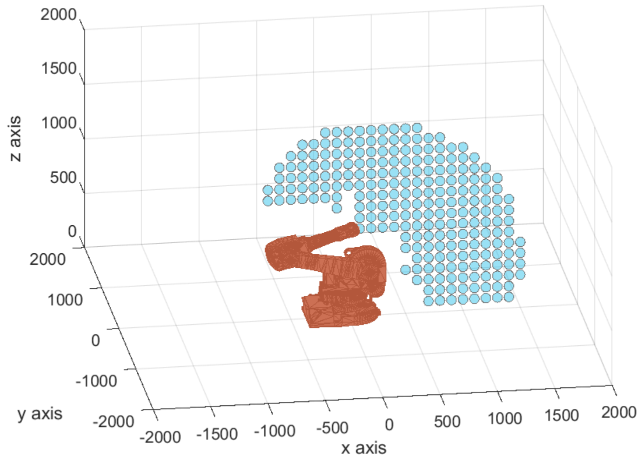

FIGURE 2. 3D view of the reachability plane xz (with $y = 0$) of the ABB IRB 1600 for elbow INWARD and wrist orientation CENTER. The intersection between the robot base plane and the rotation axis of revolute joint J_1 is the origin of the coordinate system.

for more details about these wrist configurations. Alternatively, any values for all the joints can be used to characterize all the positions on the xz plane with $y = 0$ in any other specific wrist orientations.

This is the case of the reachability planes with wrist configurations LEFT and RIGHT. See section IV (A) for more details about these wrist configurations. It is always an advantage for the calculations and for less robot energy consumption if some joint values can be kept constant during the planar shift, as the joint motors do not need to move. The fact that fewer joint movements are necessary to shift discretely the EE position on a grid given a fixed EE orientation is further exploited in this method as a trajectory planning pre-optimization.

In terms of cylindrical coordinates, each reachability plane xz with $y = 0$ can be cylindrically rotated around axis z by varying the values of J1 of a certain offset within the robot reachability limits. Thus, each EE position can be expressed in 3D cylindrical coordinates (φ, z_{cyl}, r) with $\varphi = J_{1,XZ} + J_{1,offset}$, $z_{cyl} = z_{Cart}$ and $r = x$ derived from the 2D Cartesian coordinates on the plane xz with $y = 0$ and the offset of J_1. The discrete cylindrical coordinates are shown in figure 8. It is interesting to note that each cylindrical (3D) position can be reached by up to two points of each reachability plane xz with $y = 0$ because they rotate around the z axis that lies in their center. A radius can thus be negative. This is not a problem because the pathfinding algorithm takes care of the 4D coordinates associated with each of the points and evaluates them in their different robot orientations. See section IV (B) for more details.

The maximum and minimum robot joint values directly affect the discrete grid space limits. It can be seen from the reachability planes in figures 2-7.

IV. FOUR-DIMENSIONAL CYLINDRICAL PATHFINDING

The method presented in this article couples the cylindrical space created around the industrial robot symmetries

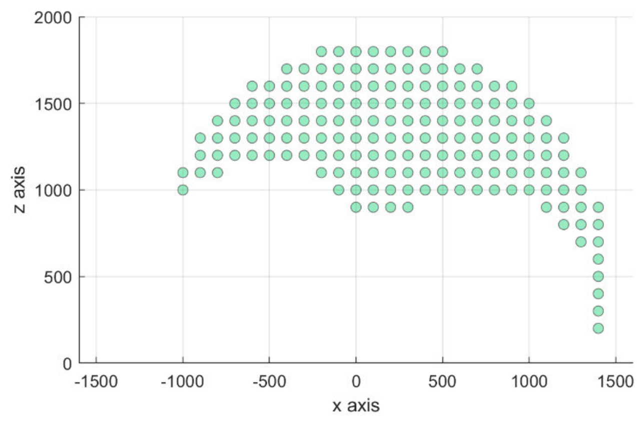

FIGURE 3. 2D view of the reachability plane *xz* (with $y = 0$) of the ABB IRB 1600 for elbow INWARD and wrist orientation UP.

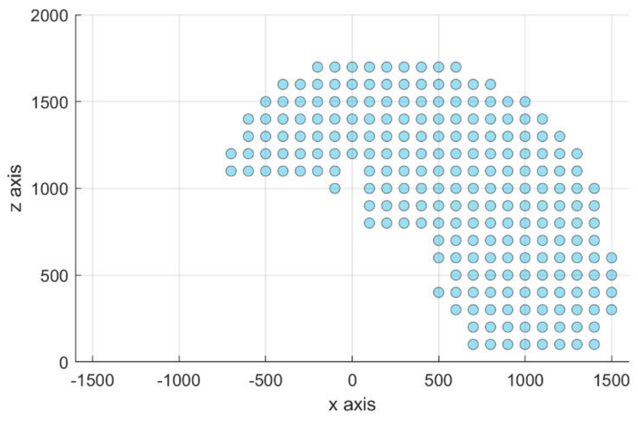

FIGURE 4. 2D view of the reachability plane *xz* (with $y = 0$) of the ABB IRB 1600 for elbow INWARD and wrist orientation CENTER.

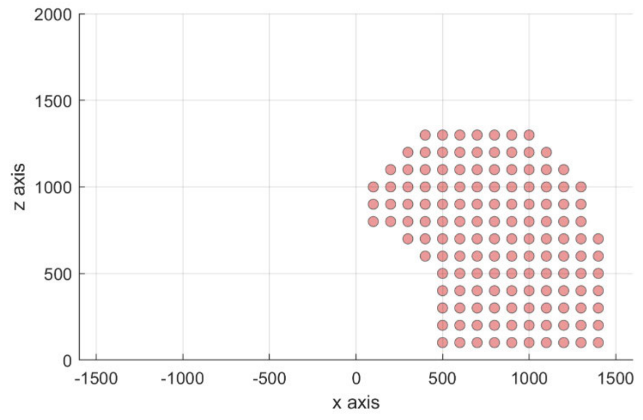

FIGURE 5. 2D view of the reachability plane *xz* (with $y = 0$) of the ABB IRB 1600 for elbow INWARD and wrist orientation DOWN.

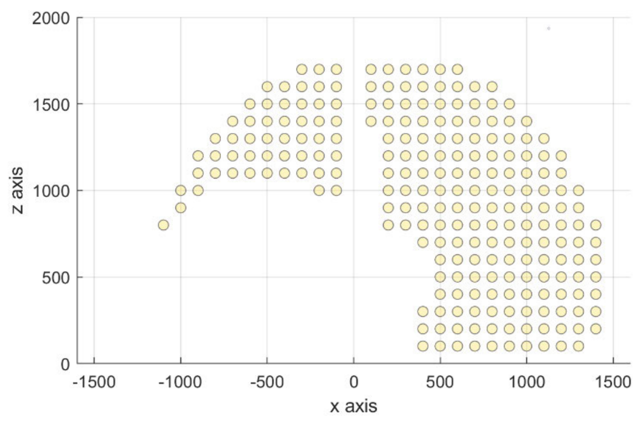

FIGURE 6. 2D view of the reachability plane *xz* (with $y = 0$) of the ABB IRB 1600 for elbow INWARD and wrist orientation LEFT.

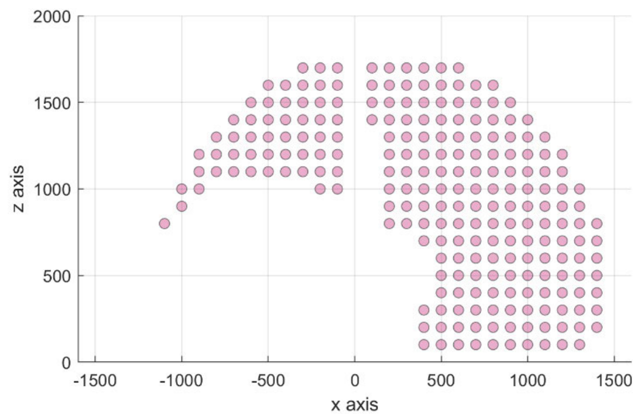

FIGURE 7. 2D view of the reachability plane *xz* (with $y = 0$) of the ABB IRB 1600 for elbow INWARD and wrist orientation RIGHT.

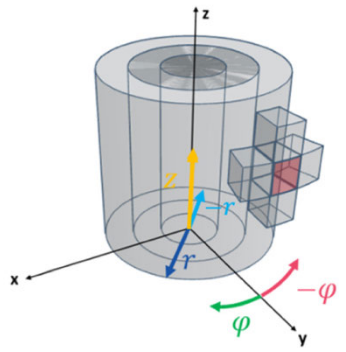

FIGURE 8. Cylindrical coordinate system and a few cylindrical grid cells. A cell is identified by its center position in cylindrical coordinates.

with an artificial intelligence pathfinding algorithm. This section covers all the steps toward a successful implementation of a pathfinding algorithm in 4D cylindrical space. The steps covered in this section go from the practical acquisition of the robot joint values for the knowledge-base (A), to the definition of the fourth dimension (B), the definition of robot volume occupancy for collision avoidance purposes (C), the discrete grid search with A* (D), the path tracking with a robot controller (E) and some benchmarking values (E).

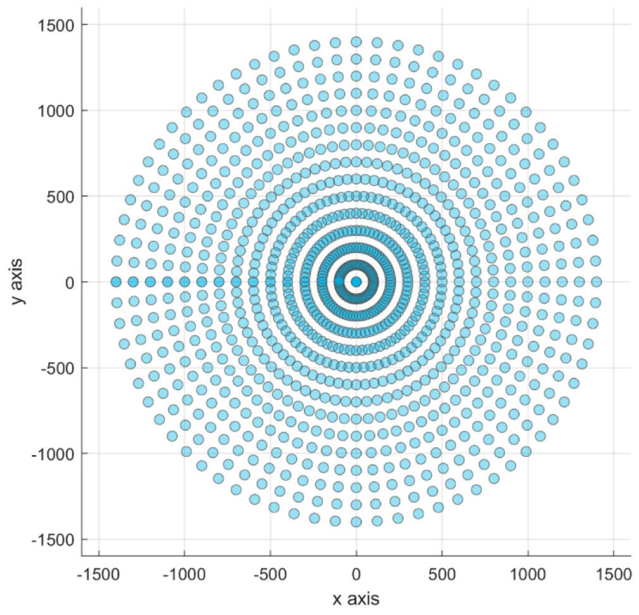

FIGURE 9. A Cartesian plane *xy* showing the polar grid for any height *z* in the cylindrical grid. The density of the polar grid is variable because there is a constant number of points per radius. The grid overlaps for angles $\varphi = \pm\pi$. Despite the density not being ideal, this configuration allows the *xz* planes to rotate around the z axis.

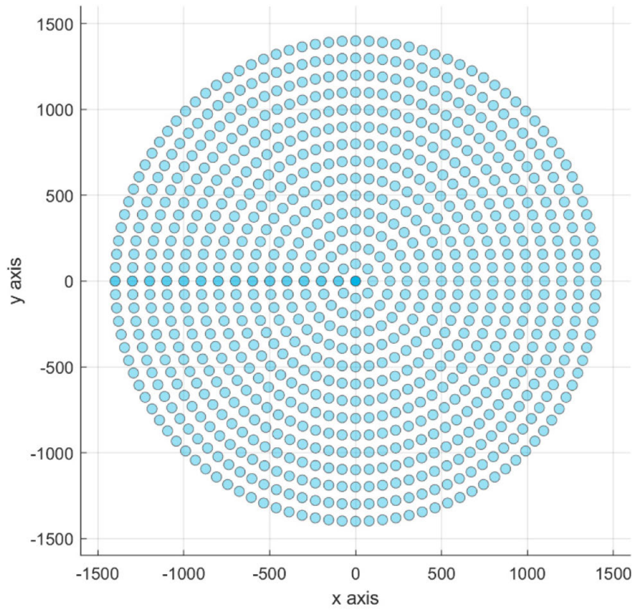

FIGURE 10. A Cartesian plane *xy* showing the polar grid for any height *z* in the cylindrical grid. The density of the polar grid is constant because there is a variable amount of points per radius. The grid overlaps for angles $\varphi = \pm\pi$. This configuration has an ideal density but it does not allow the *xz* planes to rotate around the z axis without a loss of data points.

A. ACQUISITION OF ROBOT JOINT VALUES

The data points necessary to characterize each reachability plane for a certain wrist configuration are sampled directly from RobotStudio®, the simulation software developed by ABB. This step could be simulated in other mathematical software, e.g. Matlab®, using the robot kinematic model. Using ABB RobotStudio® allows the researchers to be sure of the accuracy of the robot model built in the software by the robot producer and therefore to obtain accurate joint values.

The authors have tested the approach on five 2D reachability planes with inward J2-J3-elbow and wrist orientations CENTRAL, UP, DOWN, LEFT and RIGHT. Each plane is characterized by a finite number of EE positions within the robot reachability in that particular orientation, see figures 3-7. The positions are distanced by 100mm forming a discrete 2D Cartesian grid, a choice that is purely made by the experimenters depending on the industrial application requirements, in this case to simply test some collision-free trajectories in space. Thus, after sampling all the robot joints for all the EE positions on a certain reachability plane, the resulting 3D cylindrical space is discretized.

The procedure produced 226 coordinates for the CENTRAL plane (see figure 4), 169 coordinates for the UP plane (see figure 3), 140 coordinates for the DOWN plane (see figure 5), 228 coordinates for the LEFT plane (see figure 6) and 228 coordinates for the RIGHT plane (see figure 7). The joint limits are reported in table I.

The density of the cylindrical grid depends on a choice that is specific to the cylindrical shape: how many discrete values are selected per radius. The only choice that allows the *xz* planes to be angularly rotated around the *z* axis and avoid data points loss in a cylindrical 3D space is to keep the number of coordinates per radius constant. This produces a variable density in the 3D cylindrical grid (see figure 9). Namely, coordinates in radiuses near the *z* axis are denser than coordinates in the larger radiuses. The criteria for choosing the number of points per radius is to evaluate the distance of two consecutive coordinates at the farthest radius and approximate it to the Cartesian distance used for the planar coordinates such that the resulting angle will divide a circumference into $\varphi - 1$ sectors and produce φ shifted planes/coordinates. The result of this calculation for the ABB IRB 1600 with a Cartesian distance of 100mm produces an optimal value of 72 points/planes per radius or 72 φ coordinates at the farthest radius of 1500mm from the *z* axis. From the point of view of the search algorithm, moving in a Cartesian grid or a cylindrical grid with constant number of points per radius (variable density) is the same, which is also the reason why this choice is a good trade-off.

The choice of optimizing the number of points per radius has the advantage to keep the cylindrical grid as much constant as possible. This would inevitably break the continuity of the radial planes. Thus, making the search more difficult with an incoherent system of coordinates, e.g. there would be φ jumps from one radius to another, as shown in figure 10. This alternative has indeed been used at least in an article presenting a polar coordinate system approach to mobile robot path planning over a fan-shaped grid [31], but the authors have opted for the variable density approach in this work.

72 points per radius correspond to a 5 degrees rotation of each reachability plane. Thus, 72 times the original samples

TABLE 1. Joint limits (angles expressed in degrees) for each reachability plane.

Reachability plane	J1 limits (deg)		J2 limits (deg)		J3 limits (deg)		J4 limits (deg)		J5 limits (deg)		J6 limits (deg)	
	Min	Max	Min	Max	Min	Max	Min	Max	Min	Max	Min	Max
CENTER	0	0	-88.88	95.68	-84.63	65	0	0	-113.52	112.93	0	0
UP	0	0	-89.75	103.14	-84.44	59.81	0	0	-114.78	52.12	0	0
DOWN	0	0	-66.47	98.07	-76.23	64.10	0	0	-32.89	114.39	0	0
LEFT	-33.02	33.02	-88.75	94.54	-81.79	67.24	-122.88	-58.80	-113.91	-67.30	-137.44	158.45
RIGHT	-33.02	33.02	-88.75	94.54	-81.79	67.24	58.80	122.88	-113.91	-67.30	-158.45	137.44

are automatically produced by shifting only the value of J_1 of multiples of 5 degrees for all the coordinates involving a reachability plane rotation. The cylindrical 3D discrete space can be counted in 16272 coordinates for the CENTRAL plane, 12168 coordinates for the UP plane, 10080 coordinates for the DOWN plane, 16416 coordinates for the LEFT plane and 16416 coordinates for the RIGHT plane. For a total of 71352 3D cylindrical coordinates, with corresponding sampled J_1, J_2, J_3, J_4, J_5 and J_6 values stored in the knowledge-base.

Working in the cylindrical space is rather easier than working in the configuration space because three dimensions can be entirely visualized within the same graph and the transformation from Cartesian to cylindrical coordinates is simpler than applying direct and inverse kinematics.

A note needs to be made for the LEFT and RIGHT reachability planes: J_1 is already rotated of a variable quantity along the plane xz with $y = 0$ to compensate for the angular rotation of the EE. This quantity has a maximum value of 33.22° over the plane which needs to be removed from the maximum rotation of J_1 around the z axis. That breaks open the cylindrical grid for $J_1 = \pm 146.78°$ otherwise closed at $J_1 = \pm 180°$ with overlapping coordinates.

B. THE FOURTH DIMENSION

Three dimensions are too few and six are too many. In order to navigate each cylindrical space resulting from each wrist orientation, a fourth dimension called "orientation" (abbr. "o") is added to the discrete 3D cylindrical space. Each cylindrical space is thus given a discrete numerical coordinate, in the choice of the authors, from 1 to 5, respectively corresponding to wrist orientations CENTRAL, UP, DOWN, LEFT and RIGHT. The discrete 4D cylindrical space is expressed by discrete coordinates (o, φ, z, r) where o is a conventional number assigned to a certain wrist orientation (e.g. wrist orientation CENTER corresponds to $o = 1$), φ represents the number of positive or negative 5-degree angular shifts from the y axis (see figure 8), z corresponds to the same height in Cartesian coordinates discretized as number of unit distances (in this case 100mm) from the origin, and r corresponds to the discrete, positive or negative, radius discretized as number of unit distances (in this case 100mm) from the origin (see figures 8 and 9). The passage from continuous 3D Cartesian coordinates to 3D cylindrical coordinates and vice versa is a simple math operation that makes use of sine and cosine.

From Cartesian to cylindrical coordinates:

$$\begin{cases} r^2 = x^2 + y^2 \\ \tan\varphi = \dfrac{y}{x} \\ z = z \end{cases} \tag{1}$$

Note that the inverse of the tangent function has an infinite number of solutions but these can be restricted to a unique solution if the quadrant where the point is located is known. Such rules are commonly presented in math books and it is left as a task for the reader to find them. From cylindrical to Cartesian coordinates the transformation is easier as sine and cosine give unique solutions:

$$\begin{cases} x = r\cos\varphi \\ y = r\sin\varphi \\ z = z \end{cases} \tag{2}$$

The only further requirement is to discretize the cylindrical coordinates obtained with the first available discrete values in the defined grid and to specify the wrist orientation as a forth cylindrical coordinate, according to the orientation values defined for the application. The range of the grid directly affects the computational time (longer for a finer grid), which means that a trade-off must be chosen to fulfill the operative needs while keeping the computations reasonably fast. This new coordinate system allows the problem of planning the path or trajectory starting from a specific configuration and arriving at another specific configuration to be solved automatically by the search algorithm. All the transitional configurations either exist in the search space or are not possible. This automatically solves also the problem of possible singularity points because they are not stored in the search space, i.e. among the discrete robot coordinates of each configuration plane, without the cost of using the full robot six-dimensional (or higher) configuration space. The assumption made in the discrete search space is that all robot joint movements can be continued between adjacent cells. This is possible because the short spatial distance between cells always allows the robot configuration to be kept during the joint movements necessary to shift the EE from cell to cell and no singularity points are encountered.

C. ROBOT VOLUMETRIC OCCUPANCY

The volume occupied by the robot body at each of all 4D coordinates can be entirely calculated offline with a

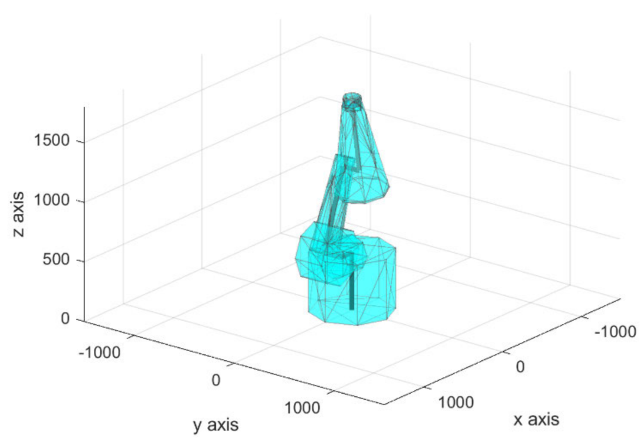

FIGURE 11. Skeleton tracking the DH notation for an ABB IRB 1600 in black and truncated cone bounding boxes representing the volumetric occupancy of the industrial robot.

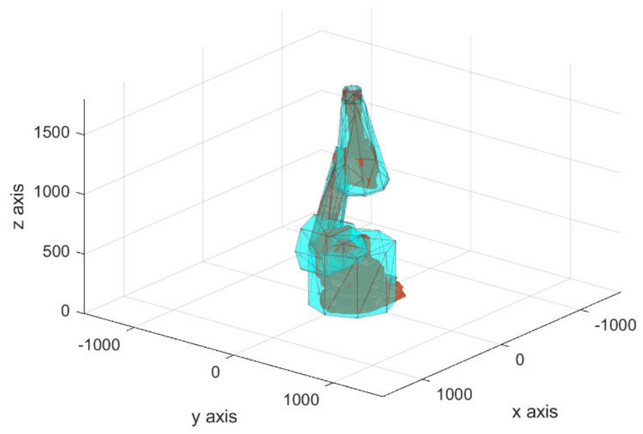

FIGURE 12. Truncated cone bounding boxes enclosing the moving parts of the industrial robot ABB IRB 1600.

computer-aided design (CAD) model of the robot or some bounding boxes around its parts and stored within the knowledge-base for online search recovery. This means that if static obstacles are present in the robot workspace, all the affected cells are removed from the search space. Not just the cells that are directly intersecting the obstacles, but also all of those for which the robot body collides with the obstacles.

The quality of the bounding boxes depends from a trade-off between computation time and accuracy needed during the offline scan of the robot workspace. Static obstacles avoidance does not require a quick rescan of the scene as much as dynamic obstacle avoidance does. The choice is obviously dependent on the industrial robotic application. For the demo in this article, only static obstacles are considered and the bounding boxes are truncated cones centered on a robot skeleton produced as a linear tracking of the cardinal points in the Denavit and Hartenberg (DH) notation for the ABB IRB 1600 robot (see figures 11-12).

D. A* PATHFINDING ALGORITHM

The goal of this work is to move the robot manipulator with discrete path searches on a 4D cylindrical grid, which is a 3D cylindrical grid enhanced by a 4th coordinate relative to a discrete orientation of the robot wrist. An optimal and collision-free path search in discrete 4D cylindrical space is performed using, among several possible algorithms presented in the introduction of this article, a common artificial intelligence search algorithm called A*. It is intended that the novelty of this section is to show how one of these algorithms, more or less efficient depending on the application requirements, can be used for the pathfinding in the 4D cylindrical grid proposed by the paper. In this case, the aim of the A* algorithm is to find the shortest path between two nodes of a given search space - a starting node and a target node - as long as there is a path. In technical terms, it achieves both optimality and completeness. The former is because the path is shortest possible, and the latter because if the solution exists the algorithm is guaranteed to find it.

The optimal decision of the A* algorithm is based on the minimization of a cost function that is split in two components for each node n: $h(n)$ and $g(n)$. The value $h(n) = d(n_s, n)$ is the exact cost of the path from the starting node n_s to node n, using equation (3) to calculate it, because the path is traversed node by node until n is reached. The other value $g(n) = d(n, n_t)$ is an estimated cost from node n to the target node n_t, using the same equation (3) to calculate it, because the rest of path has not been found and traversed yet. Both values are thus calculated using the same cost function (3) that in two- or three-dimensional maps is usually as simple as a Euclidean distance between two nodes. In the four-dimensional cylindrical map introduced by this article, the cost function proposed and tested is a pseudo-distance between two nodes $n_1(o_1, \varphi_1, z_1, r_1)$ and $n_2(o_2, \varphi_2, z_2, r_2)$:

$$d(n_1, n_2) = w(o_1, o_2) + \sqrt{(r_2 - r_1)^2 + \left(\frac{2\pi R}{n_\varphi}\right)^2 (\varphi_2 - \varphi_1)^2 + (z_2 - z_1)^2} \tag{3}$$

where

$$w(o_1, o_2) = \begin{cases} 1 & if\ o_1 = o_2 \\ 0 & else \end{cases} \tag{4}$$

is the pseudo-distance between two orientation coordinates,

$$R = \begin{cases} 0 & if\ r_1 \geq 0\ and\ r_2 \leq 0 \\ 0 & if\ r_1 \leq 0\ and\ r_2 > 0 \\ (|r_1|, |r_2|) & else \end{cases} \tag{5}$$

is the radius for which the shift between φ_1 and φ_2 is calculated, and n_φ is the density of angles φ at radius $R \neq 0$. Note that r_1 and r_2 can be either positive or negative because the reachability planes xz with $y = 0$ are rotated around the z axis around the coordinate $x = 0$.

The search algorithm creates two list of nodes: open and closed. On initialization, the starting node is added to the open list and all the nodes in the search space that are affected by obstacles are added to the closed list. A maximum number of iterations N is defined to avoid an infinite search.

The following steps are repeated until a path is found:

1) If the maximum number of iterations N is not reached, the node n on the open list with the lowest cost

$$f(n) = h(n) + g(n) \tag{6}$$

is selected as the current node.

2) The current node n is moved into the closed list and it is checked if that is the target node. In that case, the search is stopped because the path has been found.

3) All the adjacent nodes to n are considered as possible next nodes to be placed into the open list. Nodes that are in the closed list are ignored in order not to run into obstacles or run in circles.

4) If the possible next nodes are not in the open list, they are added to it and their costs calculated. If they are already in the open list, it is checked if the current path is shorter than the path for which they were added to the open list before. For each positive case, the node cost is recalculated and the predecessor node redefined as n.

Once the target node is reached, the overall path is calculated backwards from the target node to the starting node, recalling each predecessor node from the closed list.

E. PATH TRACKING WITH ROBOT CONTROLLER

The joint positions stored as data points for each cell of the discrete grid constitute the main information needed by the robot controller to track the path found by the search algorithm. Eventually transitions from cell to cell can be marked with specific maximum speeds or this can be calculated overall based on specific industrial robot application requirements. In any case, any robot controllers can generally move the robot from one position, expressed in joint coordinates, to another position with a constant-speed command. This exploits trapezoidal velocities and tends to maintain the maximum speed between consecutives joint shifts. For the ABB IRB 1600 controller, this operation is done through a specific MOVEJ (move joints) instruction in RAPID code. In the demo implemented for this article the velocity is maintained constant. Instructions are sent from a simulation environment, e.g. Matlab®, to the robot controller through TCP/IP protocol and executed in real time. ABB RobotStudio® allows the user to visualize the robot movements in a simulation environment instead of sending the commands to the real robot. This facilitates testing and deployment of code without risking to damage an industrial robot. The code used for the ABB IRB 1600 controller in RobotStudio® can be found in the repository indicated in section IV (F).

F. CODE AND BENCHMARKING

Other than the joint limits reported in table I and five reachability planes (see figures 3-7), some test trajectories are executed with A* and reported in this section with the purpose of providing benchmarking data for future implementations of this algorithm or its variations. In figure 13 it is possible to see the whole trajectory tracked by the ABB IRB 1600 in Matlab®. It starts from coordinates

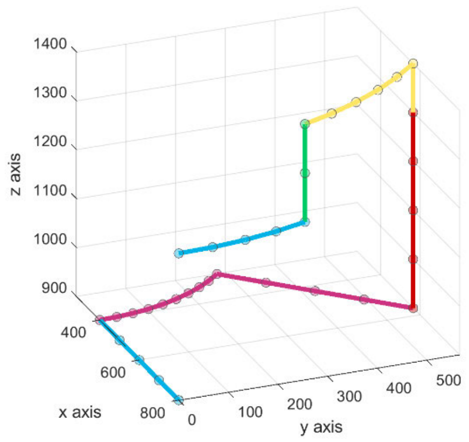

FIGURE 13. Test EE trajectory visualized in Matlab®. The robot moves accordingly to the plan described in section IV (F) across the cylindrical grid. Colors indicate the wrist orientation CENTER (blue), UP (green), DOWN (red), LEFT (pink) and RIGHT (yellow).

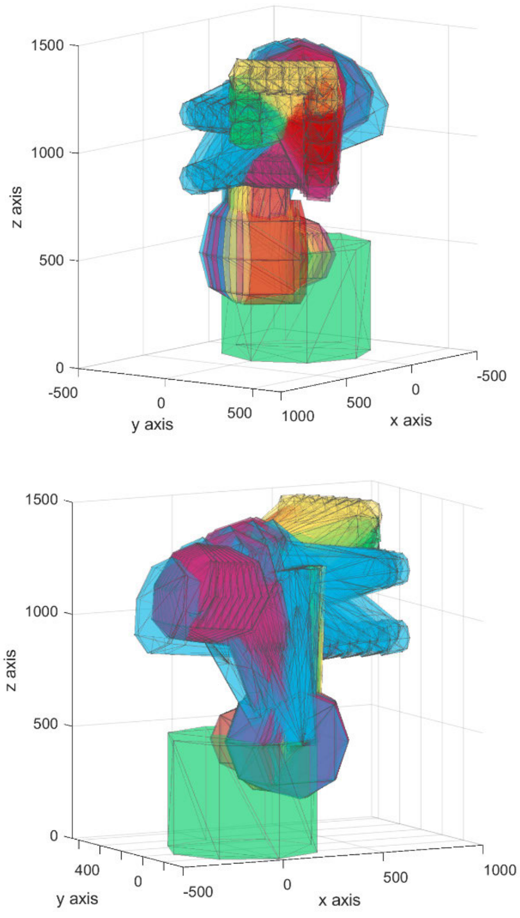

FIGURE 14. Truncated cone bounding boxes volumetric occupancy of the ABB IRB 1600 industrial robot for the test trajectory of section IV (F) visualized in Matlab®. Colors indicate the wrist orientation CENTER (blue), UP (green), DOWN (red), LEFT (pink) and RIGHT (yellow). The volume is rotated around the z axis and displayed in two perspectives for visualization purposes.

$(o, \varphi, z, r) = (1, 37, 13, 24)$ corresponding to wrist orientation CENTER and identified with a blue color of trajectory (figure 13) and volume (figure 14). It then moves to coordinates $(o, \varphi, z, r) = (1, 33, 13, 24)$ keeping the orientation CENTER. It proceeds upward to coordinates $(o, \varphi, z, r) = (2, 33, 15, 24)$ changing also the orientation to UP and the trajectory and volume color to green.

It turns leftward to coordinates $(o, \varphi, z, r) = (4, 28, 15, 24)$ changing also the orientation to LEFT and the trajectory and volume color to yellow. It then moves downward to coordinates $(o, \varphi, z, r) = (3, 28, 10, 24)$ changing also the orientation to DOWN and the trajectory and volume color to red. It continues inward and rightward to coordinates $(o, \varphi, z, r) = (5, 37, 10, 20)$ changing also the orientation to RIGHT and the trajectory and volume color to pink. It finally moves outward to coordinates $(o, \varphi, z, r) = (1, 37, 10, 24)$ changing also the orientation to CENTER and the trajectory and volume color to blue. Each of these changes of coordinates is planned and executed using A*.

An additional single point-to-point trajectory is presented for the purpose of showing the static obstacle avoidance ability of A* in the 4D space. The obstacle defined for this purpose is a cube (see figure 15) that makes it impossible for the robot to directly move from coordinates $(o, \varphi, z, r) = (1, 30, 9, 24)$ to coordinates $(o, \varphi, z, r) = (1, 44, 9, 24)$ in an approximately straight-line trajectory (see figure 15, above). The robot configuration chosen is with wrist orientation CENTER and it appears in the Matlab® simulation in blue color both for the trajectory (see figure 15, above) and the volume occupancy (see figure 15, below). From this example, it can be seen how A* computes the closest alternative trajectory that avoids any collisions between the industrial robot volume and the cube. The simulation of the collisions is entirely done in offline mode and it takes circa 7 hours to verify all the potential robot volume collisions with the predefined static obstacles. The computation of any trajectories avoiding robot volume collisions with static obstacles is done in online mode and it takes seconds (up to a few minutes depending on the distance between starting and ending points) to find a suitable collision-free path.

The code used in this project is available for download from the public GitHub repository located at the following web address: https://github.com/andreadegiorgio/cylindrical-astar.

V. DISCUSSION AND CURRENT LIMITATIONS

The characterization of the industrial robots by their symmetries, together with the necessity of modeling the robot configuration, has led to the creation of an unexplored 4D cylindrical space. Such space holds potential for the use of old and new artificial intelligence algorithms in industrial applications. The overall performance of the search algorithm applied is tied to different properties among different robot models, and trade-offs, including the definition of the discrete grid size. For the choice of the grid size, for example, other works dealing with grid search have presented

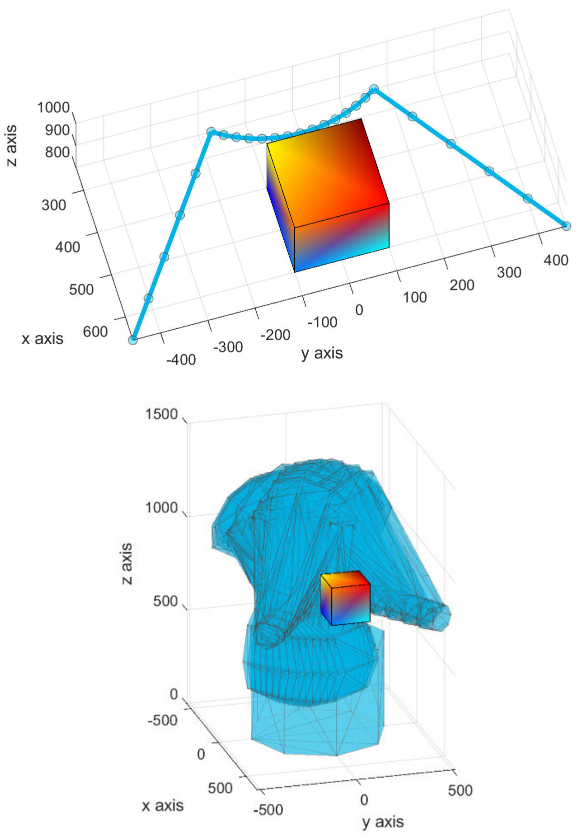

FIGURE 15. Trajectory that avoids an obstacle (above in the figure). The robot volume occupancy (below in the figure) shows how the robot arm does not touch the obstacle during the tracking of the defined trajectory.

methods implementing time step optimizations with further trajectory improvements added only when additional time is available [36].

It is hard to compare this method with online and offline trajectory planning algorithms [37], as it lies in between, with an online search of an offline mapped space. For this reason and because of the lack of benchmarking data, the performance of the search algorithm implemented in this article is not compared to other similar search algorithms, and with complementary or alternative methods. For benchmarking purposes of future implementations of this robot control method, some data points of the version tested in this article with an ABB IRB 1600 have been made available.

A limitation of the approach presented in this article is that the algorithm makes use of discrete information about the robot joint positions, which brings research back to the origins of path or trajectory planning algorithms [38], [39]. Nevertheless, algorithms to plan robot movements have firstly been proposed in discrete domains, for computers to be able to handle the complex numeric computations, and then implemented in continuous domains, thanks to the advances in computational power and mathematical software. Future research needs to address a fine grain, quasi-continuous, definition of the discrete space, thus lifting this limitation.

A distinction needs to be made between path and trajectory planning. A path pertains only to the definition of a feasible spatial path from point A to point B, without any dynamic information on the movement along the path such as time, velocities, accelerations, etc. The latter is referred to as trajectory planning. This article covers the path planning and a limited version of trajectory planning with trapezoidal speeds, tied to the introduction of velocity or acceleration information in the grid cell properties. Adapting existing and more efficient solutions to this new approach is to be considered a further research direction.

Obstacle avoidance and dynamical obstacle avoidance are also limitedly suggested as an example of possible application for the presented approach and are not extensively discussed in this article. While static obstacles can be immediately defined as part of the map searched by a traditional A* algorithm (see figure 15), dynamical obstacles require the use of more advanced algorithms such as D* (Dynamic A*) [40], which are not introduced in this article. On the other hand, the definition of robot volumetric occupancy that comes with the new method can be considered a step forward on the bounding boxes technique and its integration with static or dynamic obstacle avoidance algorithms can be innovative. The truncated cone version of bounding boxes presented in this article is also a novel trade-off between accurate robot models and traditional bounding boxes [41].

VI. CONCLUSION

The aim of this article is to characterize the typical symmetries of an industrial robot manipulator and exploit them to better apply any artificial intelligence control algorithms. A four-dimensional cylindrical discrete grid space is the outcome of this enhancement and it is shown to directly replace complex robot models at planning and runtime. The application of an artificial intelligence pathfinding algorithm, such as A* presented in this article, to the cylindrical grid becomes an innovative procedure that comes with advantages and disadvantages.

Less equations. Among the desired effects, it is possible to steer an industrial robot without any specific knowledge of the robot kinematic or dynamic models at path or trajectory planning and tracking time. It is possible to store additional information in the knowledge-base for each cell of the discrete grid, such as robot body volumetric occupancy for collision avoidance and maximum speeds for trajectory optimization.

A reduced search space. The four-dimensional cylindrical discrete space displays the main advantages of both the configuration space and the 3D workspace of an industrial robot, which improves the state of the art. In fact, the 4D search space is an improvement over the full configuration space explored by other algorithms because it holds the simplicity of searching within a 3D space, cylindrical instead of Cartesian, while having precise knowledge about the robot configuration synthetized in only one additional coordinate.

The main trade-off is between a limited capacity for offline-computed grid information and the path search speed. This study encourages the use of search algorithms for industrial robotic applications, and calls for the research of further exploitable symmetries present in different industrial robots.

No singularities. Among the advantages of the 4D space, singularity points are automatically excluded from the space when the data points are collected. The ability to skip the singularity check in the trajectories constitutes a great simplification of an industrial robot model.

New collision-avoidance algorithms. Dynamic obstacle avoidance algorithms can use the present study as a base for new implementations. A disadvantage that can be outlined is the insufficient computational power to speed up any searches in quite rich search spaces, i.e. if the fine grain of the grid increases or complex additional information is added to the cells, e.g. volumetric occupancy information. Luckily, the computational power has proven itself not to be an obstacle for more than a few years, as the technological advancements go quite fast.

Potential for human-robot collaboration. Trajectories based on the cylindrical grid are more understandable by a human operator that is working with industrial robots, because such trajectories are robot-centered and human-oriented with respect to the robot wrist orientation. Whereas trajectories based on the Cartesian space or configuration space are less representative of a collaborative human-robot environment. In fact, they are mainly aligned with objects in the workspace or not aligned with anything that can be used as reference to comprehend the robot intentions or movements.

VII. FUTURE WORK

The algorithm presented in this article opens up for further research directions and other reasonable deviations from the beaten track. A few of those are outlined below with some additional insights from the present study.

Mounting of robot tools. The operation of mounting a tool on top of the EE of the robot and consider the tool central point (TCP) as referring point for the robot joint coordinates stored in the knowledge-base invalidates all the data collected without the tool. Therefore, the introduction of each new tool requires an equivalent amount of work to produce the new offline data for the search algorithm.

Additional wrist orientations. The introduction of new wrist orientations, e.g. UP-LEFT or UP-RIGHT, does not require to change the number of search space dimensions, but the number of orientation coordinates. In fact, every new robot configuration corresponds to the definition of a new reachability plane xz with $y = 0$ and a new orientation coordinate to refer to it.

Additional elbow configuration. The introduction of the outward elbow configuration omitted in this article does not necessarily require changing the number of search space dimensions, but the number of orientation coordinates. In fact, every new robot configuration corresponds to the definition of a new reachability plane xz with $y = 0$ and a new orientation coordinate to refer to it. However, in this case there should be rules on how to transit from INWARD elbow orientation coordinates to OUTWARD elbow orientation coordinates, as the arm can only move from one robot configuration to another in specific coordinates of the grid, where the joint values distance is minimal.

Additional robot DOFs. The search space dimensionality does not depend on the dimensionality of the configuration space. Any additional DOFs for the industrial robot are compatible with the four-dimensional cylindrical discrete grid search algorithm.

Other industrial robot models. The ability to exploit symmetries that are proper of specific industrial robot models makes this algorithm highly dependent on the robot models themselves. Fortunately, the industry has settled over a finite quantity of standard robot models that show symmetries useful for kinematic calculations (wrist, elbow, etc.). The same symmetries can be exploited to create a four-dimensional discrete search space, non-necessarily cylindrical, that works well with this algorithm.

Other pathfinding algorithms. Artificial intelligence as a relatively mature discipline has produced a variety of pathfinding algorithms that can be applied to the four-dimensional discrete search space method proposed in this article. In addition to those, robotics has produced other specific algorithms that are specifically suited for industrial robot movements. Any pathfinding algorithm capable of generating a trackable path in a 4D space can be a suitable candidate for further research.

Analysis of industrial robot controllers. While Matlab® allows a user to be very accurate in producing the via points data for the ABB controller trajectory tracking with joint movements, it is not part of this article's simulation to track the performance of the ABB controller or any other robot controllers. The output trajectory executed in RobotStudio® appears visually fast and smooth, but there are no embedded functionalities in RobotStudio to analyze and export the tracked trajectory when the controller is driven through RAPID code instructions. It would be interesting to analyze in some future work the control method of some industrial controllers, and compare the stability, accuracy and rapidity when trajectory tracking.

REFERENCES

[1] L. Zhao, J. Zhao, H. Liu, and D. Manocha, "Collision-free kinematics for redundant manipulators in dynamic scenes using optimal reciprocal velocity obstacles," 2018, *arXiv:1811.00600*. Accessed: Jul. 1, 2020. [Online]. Available: http://arxiv.org/abs/1811.00600

[2] A. G. J. Kouabon, A. Melingui, J. J. B. M. Ahanda, O. Lakhal, V. Coelen, M. Kom, and R. Merzouki, "A learning framework to inverse kinematics of high DOF redundant manipulators," *Mechanism Mach. Theory*, vol. 153, Nov. 2020, Art. no. 103978, doi: 10.1016/j.mechmachtheory.2020.103978.

[3] S. Alatartsev, S. Stellmacher, and F. Ortmeier, "Robotic task sequencing problem: A survey," *J. Intell. Robot. Syst.*, vol. 80, no. 2, pp. 279–298, Nov. 2015, doi: 10.1007/s10846-015-0190-6.

[4] Z. Aljarboua, "Geometric path planning for general robot manipulators," in *Proc. World Congr. Eng. Comput. Sci.*, 2009, pp. 20–22.

[5] A. Nemati and M. Kumar, "Modeling and control of a single axis tilting quadcopter," in *Proc. Amer. Control Conf.* Piscataway, NJ, USA: Institute of Electrical and Electronics Engineers, 2014, pp. 3077–3082, doi: 10.1109/ACC.2014.6859328.

[6] S. Kucuk and Z. Bingul, "The inverse kinematics solutions of industrial robot manipulators," in *Proc. IEEE Int. Conf. Mechatronics (ICM)*, Jun. 2004, pp. 274–279, doi: 10.1109/icmech.2004.1364451.

[7] M. H. Raibert, "Manipulator control using the configuration space method," *Ind. Robot, Int. J.*, vol. 5, no. 2, pp. 69–73, Feb. 1978, doi: 10.1108/eb004494.

[8] P. Tavares, J. Lima, P. Costa, and A. P. Moreira, "Multiple manipulators path planning using double A*," *Ind. Robot, Int. J.*, vol. 43, no. 6, pp. 657–664, Oct. 2016, doi: 10.1108/IR-01-2016-0006.

[9] T. Rybus, "Obstacle avoidance in space robotics: Review of major challenges and proposed solutions," *Prog. Aerosp. Sci.*, vol. 101, pp. 31–48, Aug. 2018, doi: 10.1016/j.paerosci.2018.07.001.

[10] C. Kohrt, R. Stamp, A. G. Pipe, J. Kiely, and G. Schiedermeier, "An online robot trajectory planning and programming support system for industrial use," *Robot. Comput.-Integr. Manuf.*, vol. 29, no. 1, pp. 71–79, Feb. 2013, doi: 10.1016/j.rcim.2012.07.010.

[11] Y. Yu and V. Lippiello, "6D pose task trajectory tracking for a class of 3D aerial manipulator from differential flatness," *IEEE Access*, vol. 7, pp. 52257–52265, 2019, doi: 10.1109/ACCESS.2019.2910379.

[12] A. M. Kabir, A. Kanyuck, R. K. Malhan, A. V. Shembekar, S. Thakar, B. C. Shah, and S. K. Gupta, "Generation of synchronized configuration space trajectories of multi-robot systems," in *Proc. Int. Conf. Robot. Automat. (ICRA)*. Piscataway, NJ, USA: Institute of Electrical and Electronics Engineers, May 2019, pp. 8683–8690, doi: 10.1109/ICRA.2019.8794275.

[13] M. Bergerman and Y. Xu, "Planning collision-free motions for underactuated manipulators in constrained configuration space," in *Proc. Int. Conf. Robot. Automat.*, 1997, pp. 549–555, doi: 10.1109/robot.1997.620094.

[14] Y. Han, W. Zhao, J. Pan, Z. Ye, R. Yi, and Y.-J. Liu, "A configuration-space decomposition scheme for learning-based collision checking," 2019, *arXiv:1911.08581*. Accessed: Jul. 1, 2020. [Online]. Available: http://arxiv.org/abs/1911.08581

[15] S. Kumar, S. Choudhary, and S. Srinivasa, "Learning configuration space belief model from collision checks for motion planning," 2019, *arXiv:1901.07646*. Accessed: Jul. 1, 2020. [Online]. Available: http://arxiv.org/abs/1901.07646

[16] O. Salzman, K. Solovey, and D. Halperin, "Motion planning for multilink robots by implicit configuration-space tiling," *IEEE Robot. Autom. Lett.*, vol. 1, no. 2, pp. 760–767, Jul. 2016, doi: 10.1109/LRA.2016.2524066.

[17] K. Wei and B. Ren, "A method on dynamic path planning for robotic manipulator autonomous obstacle avoidance based on an improved RRT algorithm," *Sensors*, vol. 18, no. 2, p. 571, Feb. 2018, doi: 10.3390/s18020571.

[18] R. Geraerts and M. H. Overmars, "A comparative study of probabilistic roadmap planners," in *Algorithmic Foundations of Robotics* (Springer Tracts in Advanced Robotics). Berlin, Germany: Springer, 2004, pp. 43–57, doi: 10.1007/978-3-540-45058-0_4.

[19] L. E. Kavraki, P. Svestka, J.-C. Latombe, and M. H. Overmars, "Probabilistic roadmaps for path planning in high-dimensional configuration spaces," *IEEE Trans. Robot. Autom.*, vol. 12, no. 4, pp. 566–580, Aug. 1996, doi: 10.1109/70.508439.

[20] V. D. Tourassis and M. H. Ang, "Identification and analysis of robot manipulator singularities," *Int. J. Robot. Res.*, vol. 11, no. 3, pp. 248–259, Jun. 1992, doi: 10.1177/027836499201100307.

[21] E. Fernandes, P. Costa, J. Lima, and G. Veiga, "Towards an orientation enhanced astar algorithm for robotic navigation," in *Proc. IEEE Int. Conf. Ind. Technol. (ICIT)*. Piscataway, NJ, USA: Institute of Electrical and Electronics Engineers, Mar. 2015, pp. 3320–3325, doi: 10.1109/ICIT.2015.7125590.

[22] J. Yao, C. Lin, X. Xie, A. J. Wang, and C.-C. Hung, "Path planning for virtual human motion using improved A* star algorithm," in *Proc. 7th Int. Conf. Inf. Technol., New Gener.*, 2010, pp. 1154–1158, doi: 10.1109/ITNG.2010.53.

[23] H. Wang, J. Zhou, G. Zheng, and Y. Liang, "HAS: Hierarchical A-star algorithm for big map navigation in special areas," in *Proc. 5th Int. Conf. Digit. Home*. Piscataway, NJ, USA: Institute of Electrical and Electronics Engineers, Nov. 2014, pp. 222–225, doi: 10.1109/ICDH.2014.49.

[24] M. Noto and H. Sato, "A method for the shortest path search by extended Dijkstra algorithm," in *Proc. IEEE Int. Conf. Syst. Man Cybern.*, Oct. 2000, pp. 2316–2320, doi: 10.1109/icsmc.2000.886462.

[25] A. S. Vázquez and A. Adán, "Nonprobabilistic anytime algorithm for high-quality trajectories in high-dimensional spaces," *Robotica*, vol. 30, no. 2, pp. 289–303, Mar. 2012, doi: 10.1017/S0263574711000506.

[26] M. Kelemen, I. Virgala, T. Lipták, L. Miková, F. Filakovský, and V. Bulej, "A novel approach for a inverse kinematics solution of a redundant manipulator," *Appl. Sci.*, vol. 8, no. 11, p. 2229, Nov. 2018, doi: 10.3390/app8112229.

[27] X. Xu, Y. Hu, J. Zhai, L. Li, and P. Guo, "A novel non-collision trajectory planning algorithm based on velocity potential field for robotic manipulator," *Int. J. Adv. Robot. Syst.*, vol. 15, no. 4, pp. 1–13, Jul. 2018, doi: 10.1177/1729881418787075.

[28] M. Kallmann and M. Mataric, "Motion planning using dynamic roadmaps," in *Proc. IEEE Int. Conf. Robot. Automat.*, Jan./May 2004, pp. 4399–4404, doi: 10.1109/robot.2004.1302410.

[29] Y. Ansari, E. Falotico, Y. Mollard, B. Busch, M. Cianchetti, and C. Laschi, "A multiagent reinforcement learning approach for inverse kinematics of high dimensional manipulators with precision positioning," in *Proc. IEEE RAS EMBS Int. Conf. Biomed. Robot. Biomechtron.*, Jun. 2016, pp. 457–463, doi: 10.1109/BIOROB.2016.7523669.

[30] M. Ossenkopf, P. Ennen, R. Vossen, and S. Jeschke, "Reinforcement learning for manipulators without direct obstacle perception in physically constrained environments," *Procedia Manuf.*, vol. 11, pp. 329–337, Jan. 2017, doi: 10.1016/j.promfg.2017.07.115.

[31] T. Li, S. Sun, and Y. Gao, "Fan-shaped grid based global path planning for mobile robot," *ROBOT*, vol. 32, no. 4, pp. 547–552, Aug. 2010, doi: 10.3724/SP.J.1218.2010.00547.

[32] C. Llopis-Albert, F. Rubio, and F. Valero, "Optimization approaches for robot trajectory planning," *Multidisciplinary J. Educ., Social Technol. Sci.*, vol. 5, no. 1, p. 1, Mar. 2018, doi: 10.4995/muse.2018.9867.

[33] K. Kaltsoukalas, S. Makris, and G. Chryssolouris, "On generating the motion of industrial robot manipulators," *Robot. Comput.-Integr. Manuf.*, vol. 32, pp. 65–71, Apr. 2015, doi: 10.1016/j.rcim.2014.10.002.

[34] L. Sciavicco and B. Siciliano, *Modelling and Control of Robot Manipulators*. London, U.K.: Springer, 2000, doi: 10.1007/978-1-4471-0449-0.

[35] D. L. Peiper. (1968). The kinematics of manipulators under computer control. Stanford University, Stanford, CA, USA. [Online]. Available: http://oai.dtic.mil/oai/oai?verb=getRecord&metadataPrefix=html&identifier=AD0680036

[36] P. Reynoso-Mora, W. Chen, and M. Tomizuka, "On the time-optimal trajectory planning and control of robotic manipulators along predefined paths," in *Proc. Amer. Control Conf.*, Jun. 2013, pp. 371–377, doi: 10.1109/acc.2013.6579865.

[37] Z. Shiller, "Off-line and on-line trajectory planning," in *Motion and Operation Planning of Robotic Systems* (Mechanisms and Machine Science). Norwell, MA, USA: Kluwer, 2015, pp. 29–62, doi: 10.1007/978-3-319-14705-5_2.

[38] T. Flash and R. B. Potts, "Communication," *Int. J. Robot. Res.*, vol. 7, no. 5, pp. 48–57, Oct. 1988, doi: 10.1177/027836498800700505.

[39] H. H. Tan and R. B. Potts, "A discrete path/trajectory planner for robotic arms," *J. Austral. Math. Soc. B. Appl. Math.*, vol. 31, no. 1, pp. 1–28, Jul. 1989, doi: 10.1017/s0334270000006457.

[40] A. Stentz, "The focussed D* algorithm for real-time replanning," in *Proc. 14th Int. Joint Conf. Artif. Intell. (IJCAI)*, vol. 2, 1995, pp. 1652–1659. Accessed: Jun. 21, 2020. [Online]. Available: https://dl.acm.org/doi/abs/10.5555/1643031.1643113

[41] B. H. Lee and C. S. G. Lee, "Collision-free motion planning of two robots," *IEEE Trans. Syst., Man, Cybern.*, vol. SMC-17, no. 1, pp. 21–32, Jan. 1987, doi: 10.1109/TSMC.1987.289330.

ANDREA de GIORGIO was born in Naples, Italy, in 1987. He received the B.S. degree in electronic engineering from the University of Naples Federico II, Italy, in 2013, and the M.S. degree in computer science and engineering with major in machine learning from the KTH Royal Institute of Technology, Stockholm, Sweden, in 2015, where he is currently pursuing the Ph.D. degree in production engineering. His research interests include artificial intelligence, machine learning, and deep learning algorithms for industrial collaborative human–machine knowledge processes in real or virtual, augmented, and mixed reality environments.

LIHUI WANG is currently a Professor and the Chair of Sustainable Manufacturing with the KTH Royal Institute of Technology, Sweden. He is actively engaged in various professional activities. He has published nine books and authored in excess of 500 scientific publications. His research interests include cyber-physical systems, cloud manufacturing, real-time monitoring and control, predictive maintenance, human–robot collaborations, adaptive, and sustainable manufacturing systems. He is a Fellow of the Canadian Academy of Engineering, CIRP, SME, and ASME. He is also a Professional Engineer in Canada, the President of the North American Manufacturing Research Institution of SME, and the Chairman of the Swedish Production Academy. He is also the Editor-in-Chief of the *International Journal of Manufacturing Research*, *Robotics and Computer-Integrated Manufacturing*, and the *Journal of Manufacturing Systems*.