IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received November 30, 2021, accepted January 14, 2022, date of publication January 19, 2022, date of current version January 27, 2022.

Digital Object Identifier 10.1109/ACCESS.2022.3144631

Automatic Collision-Free Trajectory Generation for Collaborative Robotic Car-Painting

K. ZBISS[1], A. KACEM[1], MARIO SANTILLO[2], (Member, IEEE), AND A. MOHAMMADI[1], (Member, IEEE)

[1]Department of Electrical and Computer Engineering, University of Michigan–Dearborn, Dearborn, MI 48128, USA
[2]Ford Motor Company, Dearborn, MI 48126, USA

Corresponding author: A. Mohammadi (amohmmad@umich.edu)

This work was supported by Ford Motor Company (Automatic Optimal Multi-Robot Task Allocation) under Grant 389760.

ABSTRACT This paper investigates the problem of collaborative robotic car-painting using a team of industrial manipulators that can be heterogeneous. Given the CAD model of the car, a collection of heterogeneous articulated robotic arms, and their corresponding fixed base positions on the factory floor/ceiling, the objective is to generate a collection of joint trajectories for each robot in a computationally efficient manner such that the car body can be painted by the nozzles attached to the arms while collisions during the painting process are avoided. Our solution to this computationally intensive collaborative coverage path planning relies on decoupling the collision avoidance from the coverage path planning by exploiting the inherent two-dimensional structure of the problem. In particular, our algorithm relies on partitioning the reachable space of the forearms of these robots, projecting the resulting volumes of intersection on the sides and the top of the car body, and performing the coverage planning on the resulting projected volumes. Simulation results on several industrial arms that are collaboratively painting a Ford Motor Company F-150 truck demonstrate the effectiveness of our proposed solution.

INDEX TERMS Computer integrated manufacturing, computational geometry, multi-robot systems, path planning.

I. INTRODUCTION

Collaborative robotics for flexible manufacturing entails the concerted collaboration of a group of robots to achieve a complex objective such as automatically planned assemblies and uncertainty-aware welding/painting/coating which is beyond the capabilities of a single robot [1]–[7]. As opposed to the traditional highly repetitive industrial processes, a major goal in collaborative robotics is to address the need of flexible manufacturing systems for agile customization of production lines. In the context of flexible manufacturing-based car-painting, for instance, if one or more robotic arms become dysfunctional and there is a need for continuing the painting process without any disruptions, the manufacturer might need to replace the non-operational arms with robotic manipulators of a different kind from their inventory. In another scenario, the manufacturer might need to change the make of vehicles in their production line on short notice. In both scenarios, a new collection of trajectories is needed to be planned in the least possible amount of time to address the agile manufacturing requirements.

Robotic spray painting, which is concerned with covering the object surface by means of varnish/paint/ink, utilizes spray guns that are driven by compressed-air. These spray guns atomize the fluid particles and direct them onto the industrial target surfaces [8]–[10]. A drawback of the conventional industrial painting automation systems using robots is the time-consuming nature of trajectory generation for the robotic arms [11], [12]. In order to depart from the time-consuming procedure of direct manual tool path operation, which is based on driving the robot arm by a human operator through a complete spraying cycle, CAD-based methods have been considered as a viable and less time-consuming alternative for generating proper trajectories for the painting robots [9], [13]. In these CAD-based methods, path planning for single industrial robots can be performed using alternatives such as RRT-based algorithms [14], iterative schemes with adaptively changing spray-gun stroke diameters [15], and path-constrained trajectory planning [11], to name a few.

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

In the context of collaborative robotic manipulation, a plethora of modeling, control, and optimization algorithms have been proposed in the literature (see, e.g., the recent survey by [3]). Fuzzy genetic algorithms have been employed in [16] for trajectory planning for two collaborative manipulators, where the robots considered each other as moving obstacles with unknown behaviors. After the offline generation of the robot initial paths using genetic algorithms, an online fuzzy algorithm-based decision maker corrects the robot paths to prevent them from colliding with each other. In a similar approach to [16], Safeea *et al.* [17] achieve collaborative task execution by means of generating a collection of offline computed nominal paths tailored to the industrial task at hand and generating on-the-fly modifications by means of artificial potential fields for collision avoidance. Similarly, Yoon *et al.* [18] use the concept of manipulator shadow space calculations along with modified genetic algorithms for generation of an optimal collision-free sequence for executing *planar tasks*, where the manipulators are assumed to only travel linearly from one point to another and both manipulators start moving at the same time.

In unstructured environments, where the robots might share the same workspace, the issues of timing for task allocation as well as planning the robotic motion sequences become of immense importance [19]–[21]. Furthermore, having to consider other robots as moving obstacles and preventing their collision in an online manner as manifested in [16]–[18] (see, e.g., [22]–[24] for other similar works) requires establishing secure and trustworthy machine-to-machine and possibly machine-to-cloud communication protocols [25], [26] where the robots share their status with each other. On the other hand, in the case of path planning for collaborative robotic car-painting in flexible manufacturing [9]–[11], the painting takes place in the highly structured environment of the manufacturing plant. Therefore, the issues of *offline computational complexity* and *algorithm versatility* across the utilized industrial robotic arms and vehicle makes become more pronounced as opposed to motion planners tailored to unstructured environments, which prevent collision via computationally intensive real-time calculations.

Considering the crucial need of having a versatile and computationally efficient path planning algorithm for collaborative robotic car-painting, this paper investigates the problem of path planning for car-painting using a team of robots that can be heterogeneous. It is well-known that the general centralized approaches to multi-robot path planning problems are NP hard [27], [28] and suffer the curse of dimensionality.

Contributions of the paper. To overcome the curse of dimensionality in collaborative robotic path planning, the contribution of this paper is founded in proposing a computationally efficient, hierarchical, and versatile path planning solution that relies on exploiting the geometrical features of the car-painting problem under study. Our solution to this computationally intensive problem relies on decoupling the collision avoidance objective in the 3D environment from the coverage path planning objective on the 2D surface of the car. We will prove that our proposed algorithm enjoys a polynomial-time computational complexity and is applicable to any types of articulated arms and vehicle make. Furthermore, our solution does not require any communication in-between the robotic arms to prevent collision avoidance.

FIGURE 1. Example of the CAD model of a given car. The depicted vehicle is a Ford Motor Company F-150 truck.

Our proposed algorithm relies on partitioning the reachable space of the forearms of these robots by generation of proper 3D point clouds, using the Quickhull [29] and Khachiyan's [30] algorithms from computational geometry and convex optimization literature for computing the volumes of intersection in-between the reachable spaces of the robot forearms, projecting the resulting volumes on the sides and the top of the car body, and performing coverage planning on the resulting projected volumes using the classical boustrophedon decomposition algorithm [31], [32]. We remark that our idea of partitioning the 3D workspace of the painting robots shares some similarities to the recent work by Kim and Son [2], where a Voronoi diagram-based workspace partition algorithm has been proposed for weak collaboration of unmanned ground vehicles (UGVs) in agricultural robotics. Though the Voronoi diagram-based algorithm in [2] relies on 2D node clustering calculations and covering a 2D agricultural land, our workspace partitioning takes place in a 3D setting because of the need for working with 6-DOF manipulators that operate on 3D objects.

The rest of the paper is organized as follows. We present the problem statement and our solution strategy in Section II. Next, in Section III, we present the details of our 3D workspace calculation algorithms. Thereafter, in Section IV, we present the steps for performing coverage path planning on the surface of the car and assigning cells to the industrial arms. Subsequently, we present our overall algorithm in Section VI by combining the 3D and 2D computational steps and demonstrate our simulation results for collaborative painting of a Ford Motor Company F-150 truck. Finally, we conclude this article with some final remarks and future research directions in Section VII.

II. PROBLEM STATEMENT

In this section, we state the problem, present our assumptions, and provide our solution strategy for the overall problem. The details of the solution startegy will be presented in later sections.

Collaborative Robotic Car-Painting Problem (CRCP). Given the CAD model of the car (see Figure 1), a collection of heterogeneous articulated robotic arms, and their corresponding fixed base positions on the factory floor/ceiling, generate a collection of joint trajectories for each robot in a

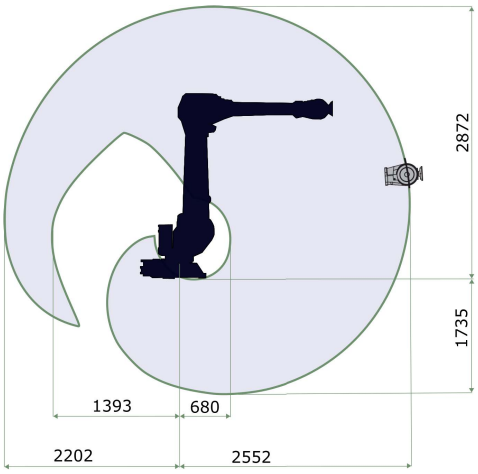

FIGURE 2. Example of an RRR articulated arm. The robot arm in the figure is an ABB IRB 4600 industrial manipulator (recreated from [37]). The unit lengths are in millimeters.

computationally efficient manner such that the car body can be painted by the nozzles attached to the arms while collisions in-between the robots are avoided.

Assumptions on CRCP. We make the following assumptions about the given problem in CRCP.

- A1 The robotic arms are all of RRR articulated type (see, e.g., [33, Chapter 1] for classification of robotic manipulators based on their kinematic structures); namely, their first three joints, which are called the body, shoulder, and elbow, are of revolute kind (see Figure 2).
- A2 The robotic arms are all either floor-mounted or ceiling-mounted with fixed base positions.
- A3 The robotic arms are arranged into two rows that are located on the sides of the car (see Figure 3 for an example).
- A4 The robotic arms do not communicate with each other. In other words, the manipulators are not aware of each other's trajectories.
- A5 For the given industrial robotic arms, there exist inverse kinematics solvers that can generate the required joint positions from the provided end-effector poses (see, e.g., [34], for the inverse kinematics of sixteen fundamental 6-DOF robot manipulators).
- A6 There exist paint deposition modeling for spray painting on automative surfaces (see, e.g., [35] or the model developed by Ford Motor Company in [36]).

Solution strategy for CRCP. Our proposed solution strategy has the following stages, which will be described in detail in later sections.

- Stage 1 Achieving the collision avoidance objective (presented in Section III).
- Stage 2 Achieving the paint coverage objective using the results from Stage 1 (presented in Section IV).
- Stage 3 Using the results from Stage 1 and Stage 2 to create 3D paths for the paint nozzles attached to the robot end-effectors (presented in Section V).

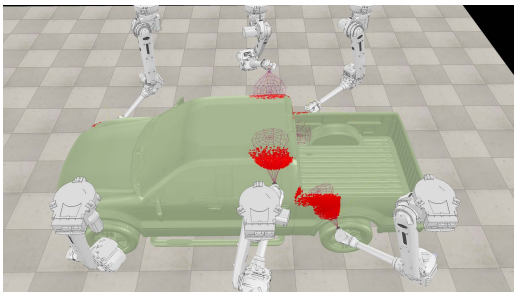

FIGURE 3. Example of the arrangement of two rows of three robotic arms around the vehicle.

III. STAGE 1: ACHIEVING THE COLLISION AVOIDANCE OBJECTIVE

In this section, we will present the 3D workspace calculations for the given robotic arms in order to guarantee non-collision. These calculations unfold in three main steps; namely, approximating the reachable workspace of the robot forearms using 3D point clouds (Section III-A), finding the volumes of intersection in-between the generated 3D point clouds (Section III-B), and fitting proper ellipsoids to the resulting volumes of intersection (Section III-C). The results of this section will later be utilized in the second stage of our algorithm, i.e., Stage 2, to achieve paint coverage objective on the car surface.

A. STEP 1: APPROXIMATING THE REACHABLE WORKSPACE OF ROBOT FOREARMS USING 3D POINT CLOUDS

In order to prevent the collision of the painting robots, we compute the reachable space of the end point of the forearms of these robots [33]. In the computation of the forearm reachable space, we do not consider the wrist of these painting robots since the volume occupied by the wrists is negligible (see Figure 2). Furthermore, as it will be demonstrated later in the paper, we exclude the volumes of intersection in-between the forearm reachable spaces via proper geometric calculations based on convex hull computations and Khachiyan's algorithm. The exclusion of these volumes of intersection ensures that the wrist collision, which might happen on the boundaries of the reachable space of the robot forearms, will not happen (see Figure 2).

In this paper, we are assuming that we are given M painting robots with articulated RRR arms (Assumption A1). We denote the angular position of the j^{th} joint angle of the i^{th} robot by θ_j^i, where $1 \leq i \leq M$ and $1 \leq j \leq 3$. Therefore, the vector of joint angles for the j^{th} robot is given by

$$\boldsymbol{\theta}^i := [\theta_1^i,\ \theta_2^i,\ \theta_3^i]^\top, \quad \theta_{j,\min}^i \leq \theta_j^i \leq \theta_{j,\max}^i, \tag{1}$$

where the constant angles $\theta_{j,\min}^i$ and $\theta_{j,\max}^i$ determine the limits of the robot joint motions (see, e.g., [37] for such limit values for ABB IRB 4600).

We also assume that the base position of these robots are fixed and expressed with respect to the origin of a common coordinate frame that is rigidly attached to the factory

floor/ceiling. It is remarked that robotic arm manufacturers devise flip over capability in some of their robots such as FANUC R-2000iC/220U [38] so that when a work cell is unsuitable for floor or wall mounting, the robot can be installed on the ceiling, and thus giving it the flexibility to fit narrow cell designs. We denote these base positions by

$$\mathbf{b}^i \in \mathbb{R}^3, \quad 1 \leq i \leq M. \tag{2}$$

To investigate the reachable volume by the forearm of each articulated robot, we use the Denavit-Hartenberg parameters associated with the first three joints of these robots along with the forward kinematics maps that relate the joint configurations to the position of the robot forearm tip. In particular, the forearm tip position $\mathbf{t}^i \in \mathbb{R}^3$ of the i^{th} robot with its base located at $\mathbf{b}^i \in \mathbb{R}^3$ on the factory floor/ceiling, $1 \leq i \leq M$, and forward kinematics mapping $T^i(\boldsymbol{\theta}^i) \in \mathbb{R}^{4\times 4}$, which maps the angular position of the first three joints to the orientation and position of the forearm tip, can be computed from

$$\begin{bmatrix} \mathbf{t}^i(\boldsymbol{\theta}^i) \\ 1 \end{bmatrix} = T^i(\boldsymbol{\theta}^i) \begin{bmatrix} 0 \\ 0 \\ 0 \\ 1 \end{bmatrix} + \begin{bmatrix} \mathbf{b}^i \\ 0 \end{bmatrix}, \ 1 \leq i \leq M. \tag{3}$$

In order to keep the algorithms for computing the reachable space of the forearms of the painting robots applicable to any given type of RRR industrial manipulator (see Assumption A1), we will use a sampling-based numerical method to approximate these reachable spaces. Using the sampling-based numerical methods such as the ones in [39]–[41], as opposed to their analytical or geometric counterparts (see, e.g., [42]), will yield 3D point clouds for which a plethora of numerically efficient algorithms such as the Quickhull [29] and Khachiyan's [30] can be invoked. Although we could have employed β-distribution-based sampling for generating the reachable volume of the robot forearm tips as in [39], we found out that uniform sampling of the joint positions for achieving the car-painting objective in CRCP suffices.

The 3D point cloud $\mathcal{P}_i$, $1 \leq i \leq M$, associated with the reachable volume of the i^{th} robot forearm tip is given by

$$\mathcal{P}_i = \left\{ \mathbf{t}^i(\boldsymbol{\theta}^k) \right\}_{k=1}^{N_i}, \tag{4}$$

where $\mathbf{t}^i(\boldsymbol{\theta}^k)$ is the position of the forearm tip under configuration vector $\boldsymbol{\theta}^k$ and determined from the forward kinematics mapping in (3), $\boldsymbol{\theta}^k$ is the k^{th} sample picked from the interval $[\theta^i_{1,\min}, \theta^i_{1,\max}] \times [\theta^i_{2,\min}, \theta^i_{2,\max}] \times [\theta^i_{3,\min}, \theta^i_{3,\max}]$, and the number of samples in the point cloud, i.e., N, is given by

$$N_i = \prod_{j=1}^{3} \left(\left\lfloor \frac{\theta^i_{j,\max} - \theta^i_{j,\min}}{\delta_{ij}} \right\rfloor + 1 \right). \tag{5}$$

In Equation (5), $\lfloor \cdot \rfloor$ denotes the floor function, and the constant δ_{ij} determines the distance between the samples in each angular interval. For instance, if $\delta_{i1} = \frac{\pi}{10}$, we mean that the samples from the interval $[\theta^i_{1,\min}, \theta^i_{1,\max}]$ are $\frac{\pi}{10}$ apart from each other. As it will be seen later, the number of points N_i in (5) in each of the forearm point clouds determines the computational complexity of the overall CRCP algorithm. Figure 4 depicts the point clouds that represent the reachable workspace by the forearms of six ABB IRB 4600 manipulators.

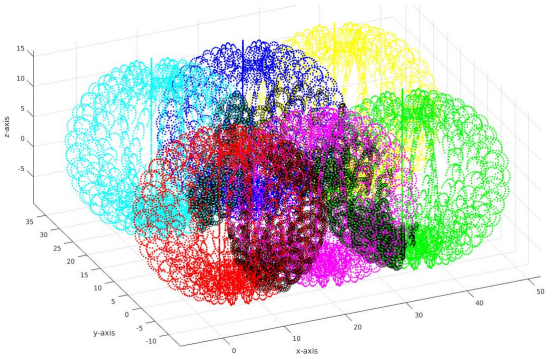

FIGURE 4. Example of the point clouds representing the reachable workspace by the forearms of six ABB IRB 4600 manipulators. The point clouds in black represent areas of possible collisions in-between the robots. The axis units are in decimeters (Note: 1 meter ≡ 10 decimeters).

B. STEP 2: FINDING THE POINT CLOUDS OF INTERSECTION IN-BETWEEN THE GENERATED 3D POINT CLOUDS

In Step 1 of Stage 1, we computed the reachable workspace of the robot forearms using 3D point clouds. In this section, we will compute the point clouds resulting from the intersection of the point clouds from the previous step. Our method is based on using the Quickhull algorithm [29], [43], [44]. For the sake of brevity, we refer the reader to textbooks such as [44, Chapters 3, 4] for the details of this algorithm. Many computational geometry packages have implemented the Quickhull algorithm. For instance, in MATLAB, the function `convhulln` can be used to run the Quickhull algorithm on a given point cloud.

The Quickhull algorithm is an efficient method for computing the convex hull of a finite set of points in n-dimensional spaces. The worst case complexity for computing the convex hull of N points in a 3-dimensional space can be shown to be $O(N \log(N))$. In our case, we need to compute the convex hull $\mathcal{CP}_i \subset \mathbb{R}^3$, $1 \leq i \leq M$, associated with the point clouds $\mathcal{P}_i$ given by (4). In other words, we have

$$\mathcal{CP}_i = \text{conv}(\mathcal{P}_i), \tag{6}$$

where $\text{conv}(\cdot)$ is the convex hull associated with a given set of points. Therefore, the worst case complexity of computing $\mathcal{CP}_i$ is given by $O(N_i \log(N_i))$, where the number of points N_i in the point cloud $\mathcal{P}_i$ is given by (5).

Having computed the convex hulls $\mathcal{CP}_i$, we compute their volumes of intersection using a proper 3D-clipping algorithm [45]. Clipping refers to the process of determining what region of 3D space is contained within another region of 3D space. For instance, one can utilize the extended Sutherland-Hodgman clipping algorithm for computing the

intersection in-between the robotic forearm workspace convex hulls to obtain the convex sets

$$\mathcal{CP}_{ij} := \mathcal{CP}_i \cap \mathcal{CP}_j, \tag{7}$$

which represent the intersection of 3D point clouds of the robotic forearm workspaces. In general, efficient 3D polyhedral intersection algorithms enjoy linear-time complexities (see, e.g., the work by Chan [46]).

Remark 1: The proposed computations in this section results in finding collision-free volumes of intersection $\mathcal{CP}_i - \bigcup_j \mathcal{CP}_{ij}$, where $\mathcal{CP}_i$ in (6) is the convex hull of the reachable workspace associated with the i-th robot and $\bigcup_j \mathcal{CP}_{ij}$ in (7) is the union of the volumes of intersection in-between the i-th robotic forearm and its neighbors.

C. STEP 3: FITTING ELLIPSOIDS TO THE VOLUMES OF INTERSECTION

Having computed the intersection of 3D point clouds of the robotic forearm workspaces in Step 2 of Stage 1, we fit ellipsoids to these volumes of intersection in Step 3. The projection of these ellipsoids on the 2D surface of the car body will later be utilized for achieving the painting coverage objective in the next section.

Consider the point cloud $\mathcal{CP}_{ij} := \{p_k\}_{k=1}^{N_{ij}}$ of N_{ij} points in $\mathbb{R}^3$, where $\mathcal{CP}_{ij}$ is computed from (7). In this section, we will use Khachiyan's algorithm, which is widely used in convex optimization [47], for fitting 3D ellipsoids to the point cloud $\mathcal{CP}_{ij}$. Khachiyan's algorithm [30] is concerned with computing the minimum volume enclosing ellipsoid (MVEE) of the point cloud $\mathcal{CP}_{ij}$ denoted by MVEE($\mathcal{CP}_{ij}$), which is obtained by solving the optimization problem

$$\begin{aligned} &\underset{A,c}{\text{argmin}} \ \log(\det(A)), \\ &\text{subject to} \quad (p_k - c)^\top A(p_k - c) \leq 1, \quad \text{for all } p_k \in \mathcal{CP}_{ij} \end{aligned} \tag{8}$$

where c is the center of MVEE($\mathcal{CP}_{ij}$) and the matrix A provides the shape of the ellipsoid MVEE($\mathcal{CP}_{ij}$). In particular, by computing the singular value decomposition of A as $U \Xi V^\top$, the radii of the ellipsoid are given by $r_k = 1/\sqrt{\xi_k}$, $1 \leq k \leq 3$, where ξ_k is the k-th component of the diagonal of the matrix Ξ. Furthermore, the orientation of the ellipsoid is given by the rotation matrix V.

Khachiyan's algorithm is based on solving the Lagrangian dual problem of (8) by using a conditional gradient ascent scheme [30] (see also [48]). As it is demonstrated in [30], the original Khachiyan algorithm for computing MVEEs can be solved using $O(N_{ij}^{3.5} \ln(\frac{N_{ij}}{\varepsilon}))$ arithmetic operations to a relative accuracy of ε in the volume. In this paper, we are using the implementation of Khachiyan's algorithm in [49].

IV. STAGE 2: ACHIEVING THE PAINT COVERAGE OBJECTIVE

In this section, we will utilize the results from the previous section in order to generate the nozzle footprint profile for achieving the paint coverage objective. Our calculations for this stage unfold in three main steps; namely, projection of the fitted ellipsoids from Stage 1-Step 3 onto the grid map representation of the sides and top of the vehicle CAD model (Section IV-A), automatic decomposition of the resulting grid maps into cells by utilizing the Morse critical points of the obtained map from the projection step and assigning back-and-forth (boustrophedon-like motions) for covering the cells by the motion of the applicator footprint (Section IV-B).

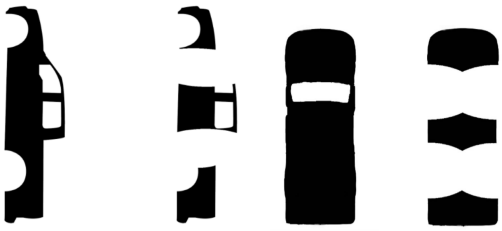

FIGURE 5. Example of projection of the fitted ellipsoids from Stage 1-Step 3 onto the grid map representation of the sides and top of the vehicle CAD model.

A. STEP 1: PROJECTION OF THE FITTED ELLIPSOIDS ONTO THE CAR CAD MODEL

As the first step for achieving the paint coverage objective, we project the fitted ellipsoids from Stage 1-Step 3 onto the grid map representation of the sides and top of the vehicle CAD model to obtain images like the one depicted in Figure 5. In order to obtain the grid map of the sides and the top of the vehicle CAD model, one can employ standard image processing algorithms such as the Moore-Neighbor Tracing algorithm for tracing given contours and detecting the boundaries of objects in a given image [50].

B. STEP 2: DECOMPOSITION OF THE VEHICLE GRID MAP INTO CELLS USING THE MORSE CRITICAL POINTS

After projecting the fitted ellipsoids from Stage 1-Step 3 onto the grid map of the sides and the top of the vehicle CAD model, we use the algorithm due to Acar *et al.* [51] for automatically decomposing the car grid maps, where the volumes of intersection have been projected onto them, by utilizing the Morse critical points of the obtained map from the previous step. The Morse critical point-based decomposition [51] is an extension of the exact cellular decomposition due to Barraquand and Latombe [52].

The exact cellular decomposition represents a robot's free space (here, the nozzle footprint profile) by dividing it into regions with simple structure such that the union of the regions is equal to the free space. The extension due to Acar *et al.* [51] relies on using the critical points of Morse functions for indicating the location of cell boundaries. Since we are using the projection of the fitted ellipsoids on the vehicle grid map, these projections have non-degenerate critical points. Having M_c critical points on a planar grid map, the algorithm due to Acar *et al.* [51] can decompose the surface into cells with complexity $O(M_c \log(M_c))$. There are numerous implementations of this algorithm online. We are using

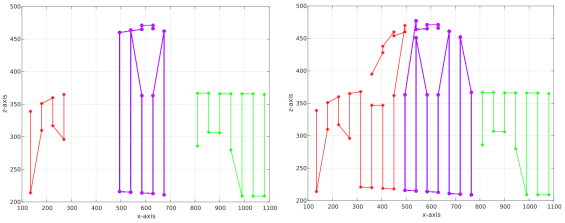

FIGURE 6. Example of assigning back-and-forth (boustrophedon-like) motions on the side of the vehicle for the applicator/nozzle footprint on the grid maps obtained from Stage 2-Step 1. Each colored boustrophedon path belongs to one of the three robots working on one side of the car (see Assumption A3).

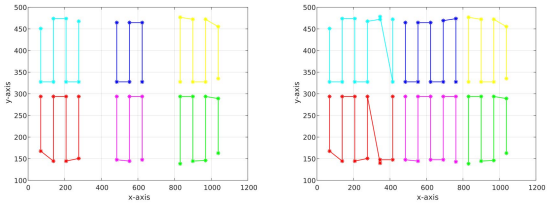

FIGURE 7. Example of assigning back-and-forth (boustrophedon-like) motions on the top of the vehicle for the applicator/nozzle footprint on the grid maps obtained from Stage 2-Step 1. Each colored boustrophedon path belongs to one of the six robots working on two sides of the car (see Assumption A3).

the MATLAB implementation by Horvath [53] for achieving the cellular decomposition.

After the computations associated with the cellular decomposition, we assign back-and-forth (boustrophedon-like) motions for covering the obtained cells by the motion of the applicator/nozzle footprint using the prominent Boustrophedon Algorithm due to Choset [32]. The result of generating such boustrophedon-like paths for 6 robots are depicted in Figures 6 and 7. As it can be seen from these two figures, the left plots in Figures 6 and 7 contain the trajectories of the applicator/nozzle footprints that are obtained from assigning boustrophedon-like paths to the areas on the sides and top of the car that do not overlap with the projection of the ellipsoids obtained from Stage 1-Step 3. Indeed, these ellipsoids are approximating the volumes of intersection in-between the workspaces of the robotic forearms. The projection of these ellipsoids onto the grid map representation of the vehicle CAD model partitions the sides and top of the car into patches with void areas (see Figure 5 for further details). The right plots in Figures 6 and 7 contain the completed nozzle/applicator footprint paths where the boustrophedon-like paths are completed by creating proper boustrophedon-like trajectories for the void areas and assigning them to the appropriate robot. For instance, consider the red-colored paths in Figures 6 and 7. After partitioning the sides and the top of the car using the projection of the fitted ellipsoids, the algorithm generates the red boustrophedon-like paths in the left plots. Next, boustrophedon-like paths are created for the projected ellipsoid areas in-between the red and the magenta-colored paths. These newly generated paths are then utilized to complete the red-colored applicator/nozzle footprint.

V. STAGE 3: TASK ASSIGNMENTS AND GENERATION OF ROBOTIC JOINT TRAJECTORIES

In this section, we will assign the painting tasks to the robots by utilizing the obtained boustrophedon-like motions from Stage 2-Step 2 (see Section V-A). Furthermore, we will create 3D paths from the obtained boustrophedon-like applicator footprints for the paint nozzles attached to the robot end-effectors and utilize these 3D paths to generate the desired joint trajectories for each of these robots (see Section V-B).

A. STEP 1: ASSIGNING ROBOTIC TASKS

Having obtained the boustrophedon-like paths of the applicator footprints on the sides and the top of the car, we compute the centroid of the obtained paths, which depends on the number of the painting robots. For instance, in the case of six robots, we have six centroids associated with the sides of the car and six centroids associated with the top of the car. We then compute the Euclidean distance in-between the centroids of the boustrophedon-like paths and the centroids of the collision-free volumes of intersection $\mathcal{CP}_i - \bigcup_j \mathcal{CP}_{ij}$, where $\mathcal{CP}_i$ in (6) is the convex hull of the reachable workspace associated with the i-th robot and $\bigcup_j \mathcal{CP}_{ij}$ in (7) is the union of the volumes of intersection in-between the i-th robotic forearm and its neighbors.

By utilizing the Euclidean distances in-between the centroid of the collision-free workspace volumes and the centroid of the boustrophedon-like paths (see Figure 8), the boustrophedon cells that are the nearest to a given reachable workspace point cloud will be assigned to their respective robotic manipulators. The result of such cell assignment is depicted in Figure 8. In Figure 8, we are using six different colors to represent the six different nozzle/applicator footprint paths assigned to each of the six robots. One collection of these footprint paths correspond to the top of the car. In particular, considering the right plot in Figure 8, these nozzle/applicator footprints on the top of the car are depicted (in a clockwise order) by the red, the magenta, the green, the yellow, the dark blue, and the ceil blue (sky blue) paths, respectively. The other collection of these footprint paths correspond to the sides (the driver's and the passenger's sides) of the car. In particular, in the left plot in Figure 8, the three colored paths correspond to the three nozzle/applicator footprint trajectories of the three robots that are located on the driver's side. In other words, the red, the magenta, and the green-colored paths represent the nozzle/applicator footprint of the three robots that are painting the vehicle on the driver's side. Due to symmetry, we have omitted the plot containing the three nozzle/applicator footprint paths (yellow, dark blue, and ceil blue) on the passenger's side.

It is remarked that due to the design procedure in Stages 1 and 2 of our algorithm when the robots are painting the sides and top of the car, they do not collide with each other.

Remark 2: Our idea of partitioning the 3D workspace of the painting robots resembles the recent work by Kim and

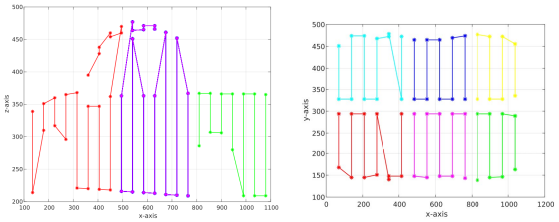

FIGURE 8. Example of assignment of painting tasks to the robots, which is done based on computing the Euclidean distances in-between the centroids of the reachable workspace point clouds of the robots and the centroids of the boustrophedon-like motions obtained from Stage 2-Step 2. Each colored collection of paths corresponds to one robot. For instance, the red paths on the side and the top of the car are assigned to a single robot.

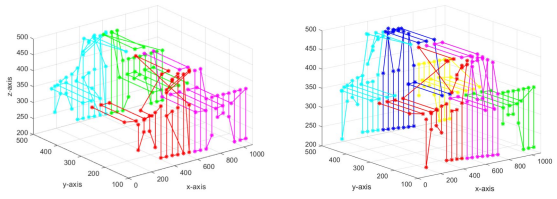

FIGURE 9. Example of generating 3D boustrophedon-like paths of the paint nozzle 3D footprints for four and six robots.

Son [2]. In that work, the authors propose using a Voronoi diagram-based workspace partition algorithm for weak collaboration of UGVs in the context of agricultural robotics. However, the algorithm in [2] relies on 2D node clustering calculations and covering a 2D agricultural land. On the other hand, our workspace partitioning takes place in a 3D setting (achieved in Stage 1). Subsequently, path planning for the applicator footprint is performed in Stage 2. Finally, based on the computations form the previous stages, robotic task assignment is done in Stage 3.

B. STEP 2: 3D PATH GENERATION FOR THE PAINT NOZZLES AND GENERATION OF JOINT TRAJECTORIES FOR THE ROBOTIC ARMS

In this section we utilize the 2D boustrophedon-like paths from the previous stage to generate 3D boustrophedon-like paths for the paint nozzle footprints. To illustrate this process, let us consider Robot 1 and its associated cells on the side and the top of the car. Each of these cells are covered by 2D boustrophedon-like paths obtained from Stage 2. In order to generate 3D boustrophedon-like paths for the paint nozzle footprint of Robot 1, we simply connect the last point of the boustrophedon-like path on the side cell to the first point of the boustrophedon-like path on the top cell associated with Robot 1 by means of a straight line segment. The result of such process for 6 and 4 robots are depicted in Figure 9.

After obtaining the 3D paths of the footprint of the nozzles, we can use proper on-the-shelf algorithms that can provide us with the paint deposition model and robotic arm joint trajectories obtained from efficient inverse kinematics solvers (see Assumptions A5 and A6).

VI. OVERALL ALGORITHM AND SIMULATION RESULTS

The flowchart associated with the overall algorithm proposed in this paper is depicted in Figure 10. The overall computational complexity of our algorithm is determined by Khachiyan's algorithm in Stage 1-Step 3. In particular, we have a polynomial-time complexity of $O(N_0^{3.5}\ln(\frac{N_0}{\varepsilon}))$, where ε is the relative accuracy of the ellipsoids fitted to the volumes of intersection in-between the reachable workspace of the robotic forearms, and the integer N_0 is proportional to the number of points in point clouds representing the reachable workspace of each robotic forearm, which is given by (5).

For the particular case of collaborative robotic car-painting, the obtained polynomial-time complexity is quite manageable for a modest PC with only one of its cores performing the required computations. For instance, the overall run-time of the algorithm implemented on MATLAB for a group of six ABB IRB 4600 robots on an Intel Core i7 CPU with an N_0 of the order of 10^3 and an ellipsoidal fitting tolerance of $\varepsilon = 0.01$ is less than four minutes. The snapshots of a homogeneous team of six robots painting a Ford Motor Company F-150 truck are depicted in Figure 11.

Our algorithm can be applied to heterogeneous teams of robots as well. This is particularly useful when one or more robotic arms become dysfunctional and there is a need for continuing the painting process without any disruptions, the manufacturer might need to replace the non-operational arms with robotic manipulators of a different kind from their inventory. The snapshots of a heterogeneous team of four robots that are painting a Ford Motor Company F-150 truck are depicted in Figure 12. For the interested reader, a movie clip associated with our numerical simulations accompanies this manuscript. The movie clip contains three painting scenarios: a team of six homogeneous robots (six ABB IRB 4600 robots), a team of four heterogeneous robots (two ABB IRB 4600 and two FANUC R-2000iC robots), and a team of four homogeneous robots (four ABB IRB 4600 robots).

Remark 4: To cope with the existence of multiple inverse kinematic solutions in path planning for serial robotic mechanisms and generating unique continuous paths in the robot joint space from a collection of potential paths, the community has proposed several solutions throughout the years. One of these popular solutions, which is also employed by CoppeliaSim and utilized in our paper, is based on utilizing damped least squares (DLS) numerical inverse kinematics solvers (also known as the Levenberg-Marquardt solvers), which were originally proposed by Wampler [54] (also, see [55]). Given a sequence of desired end-effector positions and orientations, which form a continuous path in the end-effector pose space, and an initial desired joint configuration, the DLS-based inverse kinematics solvers are guaranteed to generate a unique and continuous sequence of joint configurations that closely solve for the inverse kinematics problems associated with the provided path in the end-effector pose space. Another possibility, which is not utilized in this paper, is based on using the notion of aspects

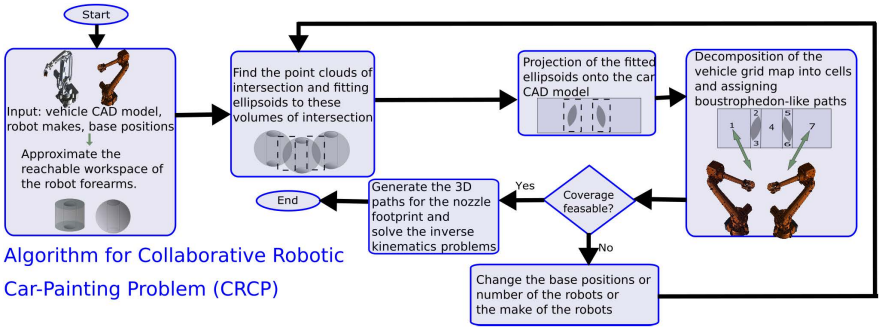

FIGURE 10. The flowchart associated with the proposed algorithm for CRCP.

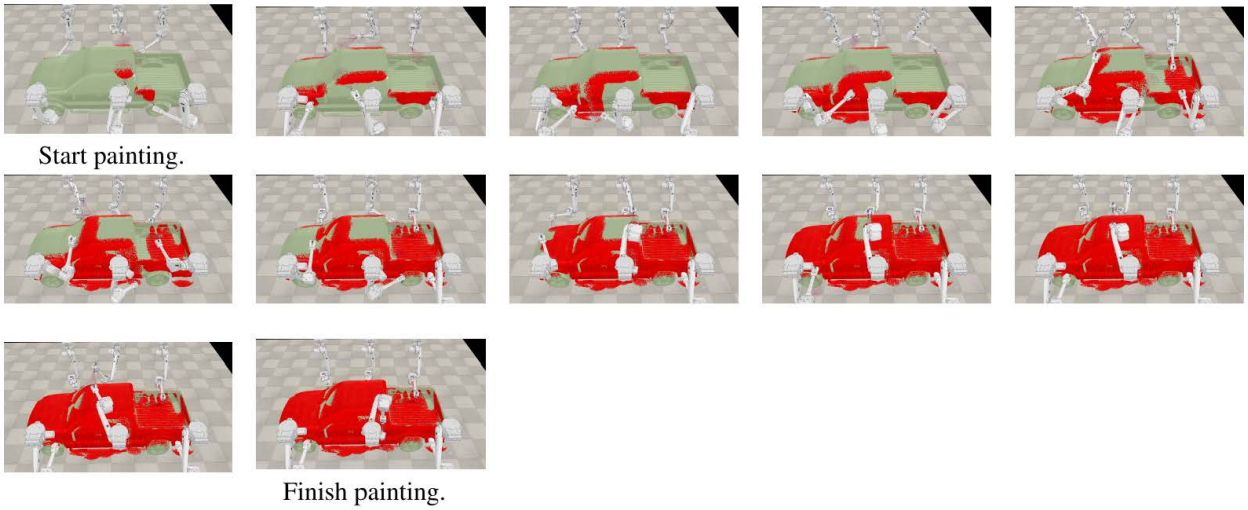

FIGURE 11. Snapshots of six ABB IRB 4600 robots painting an F-150 truck.

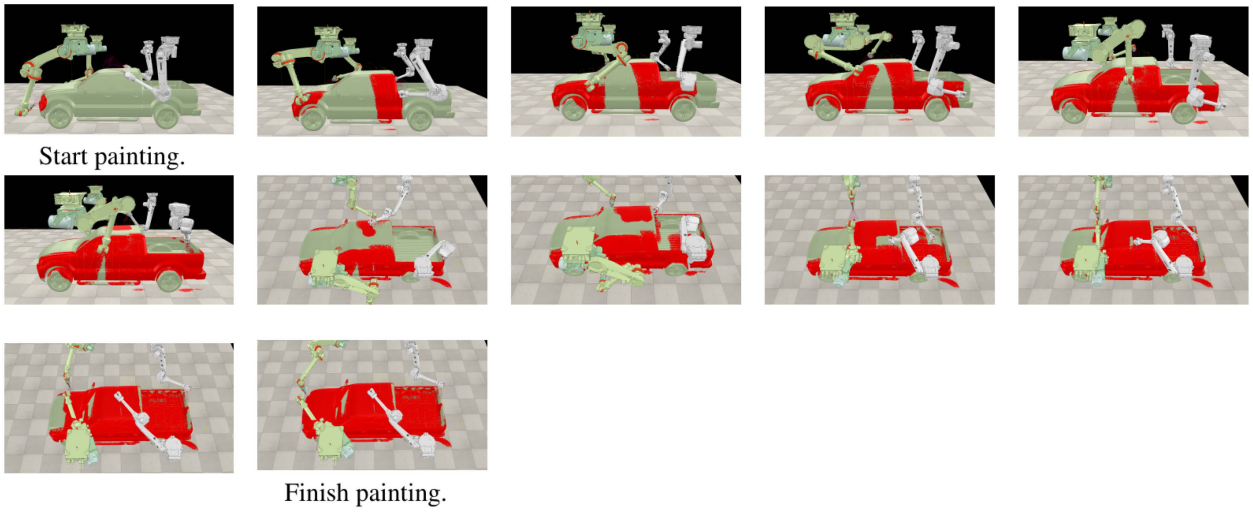

FIGURE 12. Snapshots of two ABB IRB 4600 and two FANUC R-2000iC/220U robots painting an F-150 truck.

originally proposed by Borrel and Liégeois [56] (also, see the recent work by Ferrentino and Chiacchio in [57]). In particular, Borrel and Liégeois [56] introduced the notion of aspects to cope with the existence of multiple inverse kinematic solutions in serial manipulators. The aspects are defined as the maximal singularity-free domains in the robot joint space. For conventional industrial serial robotic arms, the aspects have been found to be the maximal sets in the robot joint space where there exists only one inverse kinematics solution.

VII. CONCLUDING REMARKS

In this paper we proposed an algorithm with polynomial-time complexity for collaborative robotic car-painting using a team of industrial robotic arms. The proposed algorithm relies on partitioning the reachable space of the forearms of these robots using techniques from computational geometry and convex optimization, path planning for the nozzle/applicator footprint on the grid map representation of the car using the classical Morse cellular decomposition and boustrophedon algorithms, and using off-the-shelf inverse kinematics solvers for obtaining the required robotic arm joint trajectories.

Our proposed algorithm is versatile in that it can be applied to any heterogeneous team of industrial arms and any vehicle make. Furthermore, there is no requirement for having the industrial arms to communicate with each other during their operation.

There are some limitations associated with our proposed algorithm. First, in order to apply our proposed algorithm to more delicate surface coating tasks such as stealth coating of fighter jets, we need to adopt non-boustrophedon-like paths for generating smooth nozzle/applicator footprints (see, e.g., the survey paper in [58] for a collection of possible solutions). As another point, we remark that there exist numerically faster implementations of Khachiyan's algorithm in the literature as opposed to the standard implementation that we have utilized in this paper (see, e.g., [48]). Finally, there is a need for extending our approach to the rising problem of interaction with moving targets in the context of collaborative path planning problems for robotic arms.

In future works, the issue of optimal robot base placement, parallelizing the proposed algorithm, and optimizing the manupilability metric of the robotic arms during painting will be investigated.

ACKNOWLEDGMENT

The authors would like to sincerely thank Gary Farquhar from Ford Motor Company for his support through the course of the project. *(K. Zbiss and A. Kacem contributed equally to this work.)*

REFERENCES

[1] F. Vicentini, "Collaborative robotics: A survey," *J. Mech. Design*, vol. 143, no. 4, Apr. 2021, Art. no. 040802.
[2] J. Kim and H. I. Son, "A Voronoi diagram-based workspace partition for weak cooperation of multi-robot system in orchard," *IEEE Access*, vol. 8, pp. 20676–20686, 2020.
[3] Z. Feng, G. Hu, Y. Sun, and J. Soon, "An overview of collaborative robotic manipulation in multi-robot systems," *Annu. Rev. Control*, vol. 49, pp. 113–127, Jan. 2020.
[4] F. Zeng, J. Xiao, and H. Liu, "Force/torque sensorless compliant control strategy for assembly tasks using a 6-DOF collaborative robot," *IEEE Access*, vol. 7, pp. 108795–108805, 2019.
[5] A. Perzylo, M. Rickert, B. Kahl, N. Somani, C. Lehmann, A. Kuss, S. Profanter, A. B. Beck, M. Haage, M. R. Hansen, M. T. Nibe, M. A. Roa, O. Sornmo, S. G. Robertz, U. Thomas, G. Veiga, E. A. Topp, I. Kessler, and M. Danzer, "SMErobotics: Smart robots for flexible manufacturing," *IEEE Robot. Autom. Mag.*, vol. 26, no. 1, pp. 78–90, Mar. 2019.
[6] M. P. Polverini, A. M. Zanchettin, and P. Rocco, "A computationally efficient safety assessment for collaborative robotics applications," *Robot. Comput.-Integr. Manuf.*, vol. 46, pp. 25–37, Aug. 2017.
[7] S. Mokaram, J. M. Aitken, U. Martinez-Hernandez, I. Eimontaite, D. Cameron, J. Rolph, I. Gwilt, O. McAree, and J. Law, "A ros-integrated API for the kuka lbr iiwa collaborative robot," *IFAC-Papers Line*, vol. 50, no. 1, pp. 15859–15864, 2017.
[8] S. Moe, J. T. Gravdahl, and K. Y. Pettersen, "Set-based control for autonomous spray painting," *IEEE Trans. Autom. Sci. Eng.*, vol. 15, no. 4, pp. 1785–1796, Oct. 2018.
[9] A. Gasparetto, R. Vidoni, D. Pillan, and E. Saccavini, "Automatic path and trajectory planning for robotic spray painting," in *Proc. 7th Ger. Conf. Robot. (ROBOTIK)*, Munich, Germany: VDE, 2012, pp. 1–6.
[10] G. Biegelbauer, A. Pichler, M. Vincze, C. L. Nielsen, H. J. Andersen, and K. Haeusler, "The inverse approach of FlexPaint [robotic spray painting]," *IEEE Robot. Autom. Mag.*, vol. 12, no. 3, pp. 24–34, Sep. 2005.
[11] G. Trigatti, P. Boscariol, L. Scalera, D. Pillan, and A. Gasparetto, "A new path-constrained trajectory planning strategy for spray painting robots–rev.1," *Int. J. Adv. Manuf. Technol.*, vol. 98, nos. 9–12, pp. 2287–2296, Oct. 2018.
[12] S. C. Akkaladevi, M. Propst, M. Hofmann, L. Hiesmair, M. Ikeda, N. C. Chitturi, and A. Pichler, "Programming-free approaches for human–robot collaboration in assembly tasks," in *Advanced Human-Robot Collaboration in Manufacturing*. Cham, Switzerland: Springer, 2021, pp. 283–317. [Online]. Available: https://doi.org/10.1007/978-3-030-69178-3_12
[13] H. Chen and W. Sheng, "Transformative CAD based industrial robot program generation," *Robot. Comput.-Integr. Manuf.*, vol. 27, no. 5, pp. 942–948, Oct. 2011.
[14] H. J. Zhang, "Path planning of industrial robot based on improved RRT algorithm in complex environments," *IEEE Access*, vol. 6, pp. 53296–53306, 2018.
[15] S. Seriani, A. Cortellessa, S. Belfio, M. Sortino, G. Totis, and P. Gallina, "Automatic path-planning algorithm for realistic decorative robotic painting," *Autom. Construct.*, vol. 56, pp. 67–75, Aug. 2015.
[16] E. A. Merchan-Cruz and A. S. Morris, "Fuzzy-GA-based trajectory planner for robot manipulators sharing a common workspace," *IEEE Trans. Robot.*, vol. 22, no. 4, pp. 613–624, Aug. 2006.
[17] M. Safeea, P. Neto, and R. Bearee, "On-line collision avoidance for collaborative robot manipulators by adjusting off-line generated paths: An industrial use case," *Robot. Auto. Syst.*, vol. 119, pp. 278–288, Sep. 2019.
[18] H. J. Yoon, S. Y. Chung, and M. J. Hwang, "Shadow space modeling and task planning for collision-free cooperation of dual manipulators for planar task," *Int. J. Control, Automat. Syst.*, vol. 17, no. 4, pp. 995–1006, Apr. 2019.
[19] G. A. Korsah, A. Stentz, and M. B. Dias, "A comprehensive taxonomy for multi-robot task allocation," *Int. J. Robot. Res.*, vol. 32, no. 12, pp. 1495–1512, 2013.
[20] D. H. Lee, "Resource-based task allocation for multi-robot systems," *Robot. Auto. Syst.*, vol. 103, pp. 151–161, May 2018.
[21] A. Montano and R. Suarez, "An on-line coordination algorithm for multi-robot systems," in *Proc. IEEE 18th Conf. Emerg. Technol. Factory Autom. (ETFA)*, Sep. 2013, pp. 1–7.
[22] A. Dietrich, T. Wimbock, A. Albu-Schaffer, and G. Hirzinger, "Integration of reactive, torque-based self-collision avoidance into a task hierarchy," *IEEE Trans. Robot.*, vol. 28, no. 6, pp. 1278–1293, Dec. 2012.
[23] A. Yazici, G. Kirlik, O. Parlaktuna, and A. Sipahioglu, "A dynamic path planning approach for multirobot sensor-based coverage considering energy constraints," *IEEE Trans. Cybern.*, vol. 44, no. 3, pp. 305–314, Mar. 2013.
[24] G. Dumonteil, G. Manfredi, M. Devy, A. Confetti, and D. Sidobre, "Reactive planning on a collaborative robot for industrial applications," in *Proc. 12th Int. Conf. Informat. Control, Autom. Robot.*, Jul. 2015, pp. 450–457.
[25] G. Hu, W. P. Tay, and Y. Wen, "Cloud robotics: Architecture, challenges and applications," *IEEE Netw.*, vol. 26, no. 3, pp. 21–28, May/Jun. 2012.
[26] P. P. Ray, "Internet of robotic things: Concept, technologies, and challenges," *IEEE Access*, vol. 4, pp. 9489–9500, 2016.
[27] S. Akella and S. Hutchinson, "Coordinating the motions of multiple robots with specified trajectories," in *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, vol. 1, May 2002, pp. 624–631.
[28] S. S. Chiddarwar and N. Ramesh Babu, "Conflict free coordinated path planning for multiple robots using a dynamic path modification sequence," *Robot. Auto. Syst.*, vol. 59, nos. 7–8, pp. 508–518, Jul. 2011.
[29] C. B. Barber, D. P. Dobkin, and H. Huhdanpaa, "The quickhull algorithm for convex hulls," *ACM Trans. Math. Softw.*, vol. 22, no. 4, pp. 469–483, Dec. 1996.
[30] L. G. Khachiyan, "Rounding of polytopes in the real number model of computation," *Math. Oper. Res.*, vol. 21, no. 2, pp. 307–320, May 1996.

[31] H. Choset and P. Pignon, "Coverage path planning: The boustrophedon cellular decomposition," in *Field and Service Robotics*, A. Zelinsky, Ed. London, U.K.: Springer, 1998, pp. 203–209.
[32] H. Choset, "Coverage of known spaces: The boustrophedon cellular decomposition," *Auton. Robots*, vol. 9, no. 3, pp. 247–253, 2000.
[33] M. W. Spong, S. Hutchinson, and M. Vidyasagar, *Robot Modeling and Control*. Hoboken, NJ, USA: Wiley, 2020.
[34] S. Kucuck and Z. Bingul, "The inverse kinematics solutions of fundamental robot manipulators with offset wrist," in *Proc. IEEE Int. Conf. Mechatronics (ICM)*, Jul. 2005, pp. 197–202.
[35] D. C. Conner, A. Greenfield, P. N. Atkar, A. A. Rizzi, and H. Choset, "Paint deposition modeling for trajectory planning on automotive surfaces," *IEEE Trans. Autom. Sci. Eng.*, vol. 2, no. 4, pp. 381–392, Oct. 2005.
[36] W. Sheng, N. Xi, M. Song, Y. Chen, and P. MacNeille, "Automated CAD-guided robot path planning for spray painting of compound surfaces," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS)*, Oct. 2000, pp. 1918–1923.
[37] *Robotics: Product Specification: IRB 4600*, document 3HAC032885-001, Revision, AB, ABB Company, 2020.
[38] F. A. Corporation. (2021). *FANUC R-2000iC/220U*. Accessed: Aug. 28, 2021. [Online]. Available: https://www.fanucamerica.com/products/robots/series/r-2000/r-2000ic-220u
[39] Y. Cao, K. Lu, X. Li, and Y. Zang, "Accurate numerical methods for computing 2D and 3D robot workspace," *Int. J. Adv. Robotic Syst.*, vol. 8, no. 6, p. 76, Dec. 2011.
[40] O. Bohigas, M. Manubens, and L. Ros, "A complete method for workspace boundary determination on general structure manipulators," *IEEE Trans. Robot.*, vol. 28, no. 5, pp. 993–1006, Oct. 2012.
[41] Y. Guan, K. Yokoi, and X. Zhang, "Numerical methods for reachable space generation of humanoid robots," *Int. J. Robot. Res.*, vol. 27, no. 8, pp. 935–950, Aug. 2008.
[42] Z. Vafa and S. Dubowsky, "The kinematics and dynamics of space manipulators: The virtual manipulator approach," *Int. J. Robot. Res.*, vol. 9, no. 4, pp. 3–21, Aug. 1990.
[43] T. M. Chan, "Optimal output-sensitive convex hull algorithms in two and three dimensions," *Discrete Comput. Geomtry*, vol. 16, no. 4, pp. 361–368, Apr. 1996.
[44] F. P. Preparata and M. I. Shamos, *Computational Geometry: An Introduction*. Cham, Switzerland: Springer, 2012.
[45] J. D. Foley, A. Van Dam, S. K. Feiner, J. F. Hughes, and R. L. Phillips, *Introduction to Computer Graphics*, vol. 55. Reading, MA, USA: Addison-Wesley, 1994.
[46] T. M. Chan, "A simpler linear-time algorithm for intersecting two convex polyhedra in three dimensions," *Discrete Comput. Geometry*, vol. 56, no. 4, pp. 860–865, Dec. 2016.
[47] Y. Nesterov, "Rounding of convex sets and efficient gradient methods for linear programming problems," *Optim. Methods Softw.*, vol. 23, no. 1, pp. 109–128, Feb. 2008.
[48] M. J. Todd and E. A. Yildirim, "On Khachiyan's algorithm for the computation of minimum-volume enclosing ellipsoids," *Discrete Appl. Math.*, vol. 155, no. 13, pp. 1731–1744, 2007.
[49] N. Moshtagh. (2021). *Minimum Volume Enclosing Ellipsoid*. MATLAB Central File Exchange. Accessed: Jul. 20, 2021. [Online]. Available: https://www.mathworks.com/matlabcentral/fileexchange/9542-minimum-volume-enclosing-ellipsoid
[50] S. Biswas and R. Hazra, "Robust edge detection based on modified Moore-neighbor," *Optik*, vol. 168, pp. 931–943, Sep. 2018.
[51] E. U. Acar, H. Choset, A. A. Rizzi, P. N. Atkar, and D. Hull, "Morse decompositions for coverage tasks," *Int. J. Robot. Res.*, vol. 21, no. 4, pp. 331–344, Apr. 2002.
[52] J. Barraquand and J.-C. Latombe, "Robot motion planning: A distributed representation approach," *Int. J. Robot. Res.*, vol. 10, no. 6, pp. 628–649, 1991.
[53] E. Horvath. (2017). *Coverage Path Planning Algorithms (CPP) on Grid Maps*. Accessed: Aug. 27, 2021. [Online]. Available: https://github.com/horverno/sze-academic-robotics-projects/blob/master/RobotGridCoveragePathPlanning
[54] C. W. Wampler, "Manipulator inverse kinematic solutions based on vector formulations and damped least-squares methods," *IEEE Trans. Syst., Man, Cybern.*, vol. SMC-16, no. 1, pp. 93–101, Jan. 1986.
[55] S. R. Buss, "Introduction to inverse kinematics with jacobian transpose, pseudoinverse and damped least squares methods," *IEEE J. Robot. Automat.*, vol. 17, no. 1, p. 16, Apr. 2004.
[56] P. Borrel and A. Liegeois, "A study of multiple manipulator inverse kinematic solutions with applications to trajectory planning and workspace determination," in *Proc. IEEE Int. Conf. Robot. Autom.*, Apr. 1986, pp. 1180–1185.
[57] E. Ferrentino and P. Chiacchio, "On the optimal resolution of inverse kinematics for redundant manipulators using a topological analysis," *J. Mech. Robot.*, vol. 12, no. 3, Jun. 2020, Art. no. 031002.
[58] E. Galceran and M. Carreras, "A survey on coverage path planning for robotics," *Robot. Auton. Syst.*, vol. 61, no. 12, pp. 1258–1276, Dec. 2013.

K. ZBISS received the bachelor's degree in computer engineering from the Mediterranean University of Technology (MedTech), Tunisia. He is currently pursuing the Ph.D. degree with the Robotic Motion Intelligence Laboratory, Department of Electrical and Computer Engineering, University of Michigan–Dearborn, Dearborn, MI, USA. His current research interests include path planning for painting robots and robotic collision avoidance.

A. KACEM received the bachelor's degree in computer engineering from the Mediterranean University of Technology (MedTech), Tunisia. She is currently pursuing the Ph.D. degree with the Robotic Motion Intelligence Laboratory, Department of Electrical and Computer Engineering, University of Michigan–Dearborn, Dearborn, MI, USA. Her current research interests include path planning for painting robots and robotic collision avoidance.

MARIO SANTILLO (Member, IEEE) received the B.S. degree in aeronautical and mechanical engineering from Rensselaer Polytechnic Institute, in 2003, and the M.S. and Ph.D. degrees in aerospace engineering from the University of Michigan, in 2005 and 2009, respectively. He currently leads the robotics research efforts with the Ford Motor Company's Research and Advanced Engineering Organization, where he focuses on the development and application of advanced control methods and artificial intelligence algorithms to next-generation robotic technologies.

A. MOHAMMADI (Member, IEEE) received the master's degree from the University of Alberta, Edmonton, AB, Canada, in 2011, and the Ph.D. degree in electrical and computer engineering from the University of Toronto, Toronto, ON, Canada, in 2016. During his Ph.D. studies, he collaborated with the Norwegian Centre for Autonomous Marine Operations and Systems (a Centre of Excellence for research in Norway) on locomotion control of ground and swimming snake robots. From 2016 to 2018, he was a Postdoctoral Research Associate with the Locomotor Control Systems Laboratory, The University of Texas at Dallas, Richardson, TX, USA. He is currently an Assistant Professor with the Department of Electrical and Computer Engineering and the Director of the Robotic Motion Intelligence Laboratory, University of Michigan–Dearborn, Dearborn, MI, USA. His current research interests include collaborative robotics, cyber-security of in-vehicle networks, mechatronics, nonlinear control theory, and hybrid dynamical systems.