49th IEEE Conference on Decision and Control
December 15-17, 2010
Hilton Atlanta Hotel, Atlanta, GA, USA

A Cell Decomposition Approach to Cooperative Path Planning and Collision Avoidance via Disjunctive Programming

Ashleigh Swingler and Silvia Ferrari

Abstract— **This paper presents a novel approach for planning the minimum-distance path of multiple robotic vehicles with discrete geometries in an obstacle-populated workspace. The approach utilizes approximate cell decomposition to obtain a disjunctive program representation of C-obstacles for obstacles that are not necessarily convex polyhedrons, and robot geometries that are capable of rotating and translating in a Euclidian workspace. In order to produce programs that are computationally tractable, this approach derives a subset of all possible inequality constraints by pruning the connectivity graph based on adjacency relationships between cells, and the principle of optimality. The approach overcomes the limitations of existing approaches by simultaneously planning the paths of multiple robots, subject to any kinodynamic constraints, in environments populated by a large number of non-convex non-polyhedral obstacles. The approach is implemented using readily-available software, such as TOMLAB/CPLEX, and is illustrated here through several numerical simulation examples.**

I. Introduction

Robot path planning typically refers to the problem of determining the shortest trajectory between two robotic configurations in an obstacle-populated environment, while avoiding collisions between the robot's discrete geometry and the obstacles. Classical robotic path planning techniques, such as, cell decomposition [1]–[4], probabilistic roadmap methods [5], [6], and potential field methods [7], [8], take into account the geometries of the robot and the obstacles, while assuming that the robot is a free-flying object, or that it can be modeled by linear dynamics that are often holonomic. Although several of these approaches have been modified to account for nonholonomic dynamic constraints, they continue to suffer from several limitations. For example, cell decomposition is generally not applicable to systems involving multiple robots that must avoid collisions with each other, and it does not allow the incorporation of other classes of constraints, such as control and communication bounds. Mixed-integer linear programming and decentralized control have been used for planning the paths of multiple vehicles represented by point masses and linear dynamic equations, which can be found in [9], [10], and [11]. As shown in [12], [13], the advantages of the disjunctive programming (DP) approach over other robot motion planning techniques are that it can be used to account for nonholonomic and nonlinear robot dynamics, to avoid collisions with other robots, and to incorporate communication and control bounds.

A. Swingler and S. Ferrari are with the Laboratory for Intelligent Systems and Controls (LISC), Department of Mechanical Engineering, Duke University, Durham, NC 27708-0005, {ashleigh.swingler, sferrari}@duke.edu

On the other hand, while they can be used to account for convex polyhedral obstacles via disjunctive inequalities, existing DP approaches do not account for the robot geometry or for the geometry of non-polyhedral non-convex obstacles. Although it has been proposed in [13] that the robot geometry could be considered by augmenting the obstacles' geometries, effectively obtaining C-obstacles, in many applications C-obstacles consist of three-dimensional non-polyhedral objects that cannot be represented by a clause of disjunctive inequalities. Another disadvantage of existing approaches is that, in the presence of many obstacles or long time horizons, the solutions becomes computationally prohibitive. Moreover, existing DP approaches cannot be used to find paths in environments containing concave obstacles or narrow passages, because the corresponding linear boundary constraints can give rise to an infeasible mixed-integer program [14], [15].

The approach presented in this paper utilizes approximate cell decomposition to obtain a feasible mixed-integer program that can be used to overcome the limitations of both classical cell decomposition and DP approaches to path planning. A connectivity tree approach is presented that minimizes the computation time without loss of solution precision, and can be utilized for online and randomized path planning applications. In this approach, approximate cell decomposition is first used to decompose the obstacle-free configuration space of robots and obstacles, of any geometry, into convex rectangloid cells. The union of rectangloid cells and adjacency relationships are then used to generate a connectivity graph that can be pruned and transformed into a connectivity tree based on cell adjacency relationships and the initial configuration of the robot, using the principle of optimality. The connectivity tree specifies a subset of inequality constraints that is guaranteed to contain the optimal path, yet is significantly reduced compared to the set of all possible DP constraints that represent the workspace. The DP is then transformed into a mixed-integer program and solved using readily-available software.

The paper is organized as follows. The problem formulation and assumptions are described in Section II. The classical cell decomposition approach and disjunctive programming approach to path planning are reviewed in Section III, and the novel methodology is presented in Section IV. The numerical results, presented in Section V, illustrate that this methodology can be applied to account for rotations of discrete geometries, representing the robot, in order to avoid collisions with concave obstacles and other moving robots, while minimizing the distance traveled.

II. Problem Formulation and Assumptions

The path planning problem considered in this paper is to find the shortest paths of N mobile robots with geometries, $\mathcal{A}_i$, that consists of a compact (i.e., closed and bounded) subset of a two-dimensional Euclidian workspace denoted by $\mathcal{W} \subset \mathbb{R}^2$. The robots must simultaneously avoid mutual collisions, i.e., $\mathcal{A}_i \cap \mathcal{A}_j = \emptyset$ for all $i, j = 1, \ldots, N$, $i \neq j$, and collisions with n fixed obstacles $\mathcal{B}_1, \ldots, \mathcal{B}_n$, whose geometries and positions are estimated from prior sensor measurements and, in this paper, are assumed known *a priori*. The approach can be easily extended to higher-dimensional workspaces, and to online path planning, using an approximate dynamic programming approach [16]. The i^{th} robot configuration, $\mathbf{q}_i$, specifies the position and orientation of a moving Cartesian frame $\mathcal{F}_{\mathcal{A}_i}$, embedded in $\mathcal{A}_i$, with respect to a fixed Cartesian frame $\mathcal{F}_{\mathcal{W}}$. It is assumed that the robots' paths can be planned in concert, based on a set of given and fixed initial and final goal configurations, $Q = \{\mathbf{q}_{0_i}, \mathbf{q}_{f_i} \mid i \in \mathcal{I}_{\mathcal{A}}\}$, where $\mathcal{I}_{\mathcal{A}}$ is the set of unique identifiers representing the N robots.

In this problem formulation, the robot state or configuration is defined as $\mathbf{q}_i = [x_i \quad y_i \quad \theta_i]^T \in \mathbb{R}^3$, where (x_i, y_i) and θ_i are the coordinates and orientation of $\mathcal{F}_{\mathcal{A}_i}$ with respect to $\mathcal{F}_{\mathcal{W}}$, respectively. The robot's rotation θ, defined in Fig. 1, is restricted to the range $\Theta = [\theta^{min}, \theta^{max}]$. Without loss of generality, the method is presented for $\mathcal{C} = \mathcal{W} \times \Theta$, where $\mathcal{W} \subset \mathbb{R}^2$.

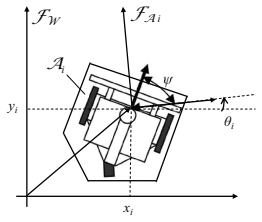

Fig. 1. Robot geometry and reference frames.

It is assumed that minimal distance can be represented by a convex cost function modeled by the square of the L_2-norm,

$$J = \int_0^T \sum_{i=1}^{N} \left(\mathbf{q}_i(t) - \mathbf{q}_i(t-1)\right)^2 \tag{1}$$

Based on the above assumptions, the coordinated geometric path planning problem can be formulated as a mixed-integer linear program (MILP) that can be readily solved using the TOMLAB/CPLEX software, using the approach presented in the following sections. As will be shown in a separate paper, this problem formulation can also be extended to account for control and communication bounds, and for optimizing fuel efficiency subject to nonlinear vehicle dynamics.

III. Background

A. Review of Cell Decomposition for Robot Path Planning

Cell decomposition is a well-known obstacle avoidance method that decomposes the obstacle-free robot configuration space into a finite collection of non-overlapping convex polygons, known as cells, within which a robot path is easily generated. Let the configuration space $\mathcal{C}$ denote the space of all possible robot configurations. A C-obstacle is a subset of $\mathcal{C}$ that causes collisions between the i^{th} robot and at least one obstacle in $\mathcal{W}$, i.e., $\mathcal{CB}_j \equiv \{\mathbf{q}_i \in \mathcal{C} \mid \mathcal{A}_i(\mathbf{q}_i) \cap \mathcal{B}_j \neq \emptyset\}$, where $\mathcal{A}_i(\mathbf{q}_i)$ denotes the subset of $\mathcal{W}$ occupied by the platform geometry $\mathcal{A}_i$ when the robot is in the configuration $\mathbf{q}_i$ [17]. Then, the union $\bigcup_{j=1}^{n} \mathcal{CB}_j$ is the *C-obstacle region*, and the obstacle-free robot configuration space is defined as

$$\mathcal{C}_{free} \equiv \mathcal{C} \setminus \bigcup_{j=1}^{n} \mathcal{CB}_j = \{\mathbf{q}_i \in \mathcal{C} \mid \mathcal{A}(\mathbf{q}_i) \cap (\bigcup_{j=1}^{n} \mathcal{B}_j) = \varnothing\} \tag{2}$$

In classical cell decomposition, the union of the cells representing $\mathcal{C}_{free}$ is obtained by implementing a line-sweeping algorithm, and by constructing a one-dimensional representation of the free-space geometry known as a connectivity graph. The connectivity graph can then be searched for the shortest path between the two cells containing the desired initial and final robot configurations.

One advantage of cell decomposition over existing mixed-integer programming methods is that it guarantees collision avoidance between a robot with any discrete geometry $\mathcal{A}_i$, and obstacles of any shape, that are not necessarily convex. Using an approach known as *approximate-and-decompose* [1], an approximate cell decomposition of $\mathcal{C}_{free}$ can be obtained for robot geometries that are not restricted to planar objects, three-dimensional convex polytopes, or polyhedral objects [2]–[4]. In this approach, cells of a predefined rectangloid shape are used to decompose the bounding and bounded approximations of the obstacles, until the connectivity of $\mathcal{C}_{free}$ is properly represented [1]. By requiring all cells to have the same rectangloid shape, this approach simplifies the implementation and reduces the sensitivity to numerical approximations, achieving a precision that can be made arbitrarily small at the expense of running time. Therefore, in the approximate cell decomposition case, there is a tradeoff between running time and solution completeness. Then, the union of the cells that are strictly outside the C-obstacle region is used to construct a connectivity graph, which represents the adjacency relationships between the cells.

The disadvantages of cell decomposition are that it is computationally intensive in high-dimensional configuration spaces (e.g. robot manipulators), and that it does not typically allow the user to incorporate other motion constraints, such as, nonholonomic dynamics, or communication constraints. Also, it is not directly applicable to cooperative networks, in which the path of one robot is influenced by that of the other agents in the network. By combining approximate cell decomposition with disjunctive programming, as shown in this paper, these limitations can be overcome, while retaining the classic features of completeness and obstacle avoidance.

B. Review of Disjunctive Programming for Robot Path Planning

Disjunctive programming (DP) includes logical constraints in discrete and continuous optimization problems [18] by

representing constraints as a conjunction $\bigwedge$ = AND of r clauses, with each clause being a disjunction of m_ℓ inequalities. Therefore, a cost function $f(\mathbf{x})$ can be minimized as follows,

$$\begin{aligned} &\min f(\mathbf{x}) \\ &\text{subject to} \quad \bigwedge_{\ell=1\ldots r} \left(\bigvee_{j=1\ldots m_\ell} C_{\ell j}(\mathbf{x}) \leq 0 \right) \end{aligned} \tag{3}$$

with respect to a vector $\mathbf{x}$ of decision variables that satisfy one or more of the m_ℓ inequalities. As reviewed in [13], [15], if the cost function and the constraints are linear (or quadratic) with respect to $\mathbf{x}$, then the DP can be formulated as a mixed-integer linear (or quadratic) program using m_ℓ binary slack variables whose sum is less than or equal to $(m_\ell - 1)$, such that at least one inequality is satisfied in each clause.

This technique has been recently applied to robot motion planning in obstacle populated environments in which the robot can be modeled as a point mass, and the obstacles can be represented by convex polyhedral objects [12], [13], [15], [19]. Under these assumptions, a disjunction of m_ℓ inequality constraints can be used to guarantee that the robot position lies outside a two-dimensional polyhedral obstacle with m_ℓ boundaries. Then, the i^{th} robot is said to avoid all obstacles if its configuration $\mathbf{q}_i$ satisfies the constraints,

$$\bigwedge_{\ell=1,\ldots,r} \bigvee_{l=1,\ldots,m_\ell} \mathbf{a}_{l\ell}^T \mathbf{q}_i(t) > b_{l\ell} \tag{4}$$

at any time $t \in [0,\ T]$, where m_ℓ is the number of sides of $\mathcal{B}_\ell$, and $\mathbf{a}_{l\ell}$ and $b_{l\ell}$ are coefficients representing the obstacle's linear boundaries.

As shown in [12], [13], the advantages of the DP approach over classical motion planning techniques, such as cell decomposition (in its traditional form) and probabilistic roadmap methods, is that it can be used to account for non-holonomic robot dynamics, and to avoid collisions in multi-agent systems. The disadvantages of existing DP approaches are that it cannot account for robot geometry or for the geometries of non-polyhedral non-convex obstacles, and that, in the presence of many obstacles or long time horizons, the solutions becomes computationally prohibitive. Although it has been proposed in [13] that the robot geometry can be considered by augmenting the obstacles' geometries, effectively obtaining C-obstacles, it can be easily shown that in the presence of rotations, C-obstacles are three-dimensional non-polyhedral objects that cannot be represented by a clause of disjunctive inequalities. Moreover, classical DP cannot be used to find paths in environments containing concave obstacles or narrow passages because the corresponding linear boundary constraints can give rise to an infeasible mixed-integer program [14], [15]. The approach presented in the next section utilizes cell decomposition to obtain a feasible mixed-integer program that can be used to overcome the limitations of both classical cell decomposition and DP approaches to path planning. Also, a connectivity tree approach is presented that minimizes the computation time without loss of solution precision, and can be utilized for online and randomized path planning applications.

IV. Methodology

The methodology presented in this paper can be outlined as follows. After computing the three-dimensional C-obstacles in $\mathcal{C} = \mathcal{W} \times \Theta$ for a discrete set of rotation intervals, the obstacle-free configuration space, $\mathcal{C}_{free}$, is decomposed into rectangloids using an approximate-and-decompose approach, described in Section IV-A. From the decomposition, a connectivity tree is formed using the adjacency relationships between the cells, and the initial robot configuration, $\mathbf{q}_{0_i}$ (Section IV-B). Each branch in the connectivity tree represents a collision-free channel connecting $\mathbf{q}_{0_i}$ to $\mathbf{q}_{f_i}$, where each cell can be modeled by a clause in a disjunctive program that represents a set of configurations that the robot *can* occupy to avoid collisions. The approach presented in this paper differs from prior literature in the manner by which the disjunctions are utilized, and in its implementation of approximate cell decomposition in combination with DP. In this paper, a DP clause of conjunctions represents a cell, i.e., a convex set of robot configurations that are allowed, as opposed to using a DP clause of disjunctions to represent a convex set of robot configurations that are not allowed. Finally, the optimal paths of N robots can be determined from the connectivity tree by solving a mixed-integer linear program using the CPLEX optimization software with a TOMLAB/Matlab interface.

A. Approximate-and-Decompose Approach and Connectivity Trees

In the cell decomposition approach, the free configuration space, $\mathcal{C}_{\text{free}}$, is decomposed into a set of convex, non-overlapping polygons, known as *cells*, denoted by $\mathcal{K}_{\text{void}}$, such that a path between any two configurations in a cell can be easily generated. For special classes of planar objects, and for three-dimensional convex polytopes or polyhedral objects an exact decomposition can be obtained [2]–[4] such that the union of the cells in $\mathcal{K}_{\text{void}}$ is exactly the free configuration space, $\mathcal{C}_{\text{free}}$. When the robots and the obstacles do not fall into one of these special classes, an approximate cell decomposition method can be utilized that decomposes a subset of the free configuration space into non-overlapping predefined rectangloids. As a result, all cells are convex polyhedrals that can be represented by a conjunction of inequalities defined in the configuration space. Approximate cell decomposition is necessary when representing the free configuration space of robots that are capable of rotating. Although for a purely-translating robot the obstacles can be augmented by its geometry, as shown in Fig. 2, to obtain polyhedral C-obstacles in $\mathcal{W}$, for a rotating robot in $\mathbb{R}^2$, the C-obstacles are three-dimensional non-polyhedral compact subsets of $\mathcal{C} = \mathcal{W} \times \Theta$, even when the robot geometry is a convex polyhedral. An example of three-dimensional C-obstacles is shown in Fig. 3 for a workspace. This workspace is revisited in Section V, and can be seen in Figs. 8 and 9.

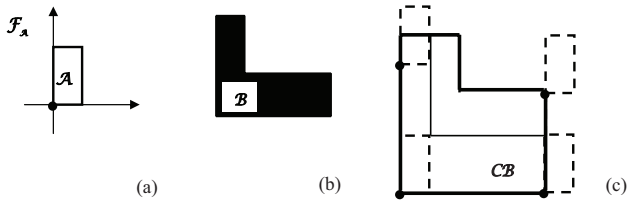

Fig. 2. Example of C-obstacle $\mathcal{CB}$ (c) obtained for a robot geometry $\mathcal{A}$ (a) and an obstacle with geometry $\mathcal{B}$ (b).

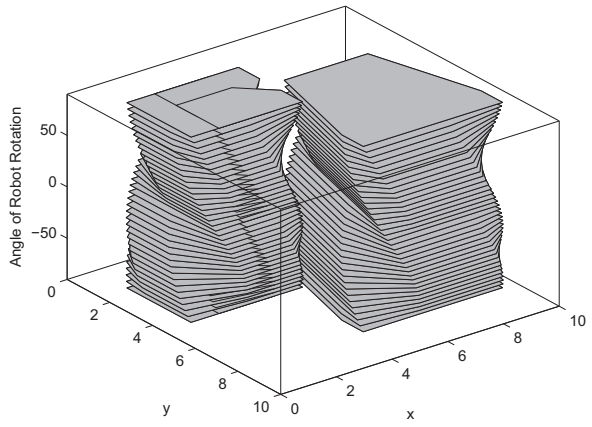

Fig. 3. Representation of a 3D C-Obstacle for a 2D robot $\mathcal{A}$, and 2D obstacles $\mathcal{B}$.

The approximate-and-decompose method is applicable to C-obstacles and robots of any shape, and obtains a decomposition $\mathcal{K}_{\text{void}}$ of obstacle-free disjoint rectangloids whose union is an approximation of $\mathcal{C}_{\text{free}}$. A rectangloid is used to represent a closed region in $\mathcal{C}$, defined as $\kappa = [x_\kappa, x'_\kappa] \times [y_\kappa, y'_\kappa] \times [\theta_\kappa, \theta'_\kappa]$. Due to their predefined shape, the cells' boundaries do not always coincide with the boundaries of C-obstacles. Therefore, a subset of cells, referred to as *mixed*, may contain configurations from $\mathcal{C}_{free}$, as well as from the C-obstacle region. Mixed cells are excluded from the connectivity graph because they may lead to collisions with the obstacles. Their volume can be minimized by approximating C-obstacles as unions of non-overlapping rectangloids, before performing the decomposition [1]. C-obstacle approximations are used to determine which rectangloids to exclude from the connectivity graph.

Definition 4.1: A *bounding rectangloid approximation* of $\mathcal{CB}_j[\kappa] = \mathcal{CB}_j \cap \kappa$, denoted by $\mathcal{RB}_j[\kappa]$, is a collection of non-overlapping rectangloids $\mathcal{R}_{jv}, v = 1, \ldots, p$, whose union contains $\mathcal{CB}_j[\kappa]$.

The n bounding rectangloid approximations of the C-obstacles are used to calculate $\mathcal{K}_{\text{void}}$ according to the following procedure:

1) Discretize the range of rotations $\Theta = [\theta^{min}, \theta^{max}]$ into ν non-overlapping intervals $\mathcal{I}_u = [\gamma_u, \gamma_{u+1}]$, where $u = 1, \ldots, \nu$, $\gamma_1 = \theta^{min}$ and $\gamma_{v+1} = \theta^{max}$, and such that $\kappa^u = [x_\kappa, x'_\kappa] \times [y_\kappa, y'_\kappa] \times \mathcal{I}_u$.
2) Compute $\mathcal{CB}_j[\kappa^u]$, for $j = 1, \ldots, n$, and for every $u = 1, \ldots, \nu$, by discretizing $\mathcal{I}_u$ into k_u fixed values $\gamma_u + l\Delta\theta$, for $0 \le l \le k_u$, and $\Delta\theta = (\gamma_{u+1} - \gamma_u)/k_u$.
3) For every rotation interval, indexed by $u = 1, \ldots, \nu$, and every obstacle indexed by $j = 1, \ldots, n$, the outer projection of the C-obstacles, compute the outer projection,
$$\mathcal{OCB}_j[\kappa^u] = \{(x, y) \mid \exists \theta \in \mathcal{I}_u : (x, y, \theta) \in \mathcal{CB}_j[\kappa^u]\} \tag{5}$$
and calculate the bounding rectangloid approximation $\mathcal{RB}_j[\kappa^u]$ of $\mathcal{OCB}_j[\kappa^u] \times \mathcal{I}_u$.
4) For every $u = 1, \ldots, \nu$, generate a rectangloid decomposition of the void configuration space,
$$\mathcal{K}^u_{\text{void}} \equiv \kappa^u \setminus \left\{ \bigcup_{j=1}^{n} \mathcal{RB}_j[\kappa^u] \right\} \tag{6}$$
and let $\mathcal{K}_{\text{void}} = \cup_{u=1}^{v} \mathcal{K}^u_{\text{void}}$, denote the approximate cell decomposition of $\mathcal{K}_{\text{void}}$.

After the above decomposition procedure is completed, $\mathcal{K}_{\text{void}}$ is used to obtain a connectivity graph and tree, for the i^{th} robot, based on its initial and goal configurations $\mathbf{q}_{0_i}$ and $\mathbf{q}_{f_i}$. As a first step, a connectivity graph is used to represent the robotic sensor as a point in configuration space, as follows:

Definition 4.2: A *connectivity graph*, $\mathcal{G}$, is a non-directed graph where the nodes represent rectangloid cells in $\mathcal{K}_{\text{void}}$, and two nodes κ_i and κ_j in $\mathcal{G}$ are connected by an arc (κ_i, κ_j) if and only if the corresponding cells are adjacent in $\mathcal{C}_{free}$.

Several approaches can be used to search $\mathcal{G}$ for a channel, a sequence of adjacent cells, connecting $\mathbf{q}_0$ to $\mathbf{q}_f$. The search can also be simplified by pruning the connectivity graph based on distance, using the label-correcting algorithm presented in [20]. The connectivity graph is pruned and transformed into a decision tree, reducing the number of feasible channels. A connectivity tree is a graphical representation of all possible channels in $\mathcal{G}$, between κ_0 and κ_f. After visiting a cell, a robot can only move to an adjacent cell, creating a causal process that can be represented as follows:

Definition 4.3: The *connectivity tree* T associated with $\mathcal{G}$ and two cells $\kappa_0 \ni q_0$ and $\kappa_f \ni q_f$ is a tree graph with κ_0 as the root and κ_f as the leaves. The nodes represent void cells. A branch from the root to a leaf represents a channel joining κ_0 and κ_f.

As defined above, the connectivity tree requires that the robot, in any trajectory, must travel along the cells in a channel. For the example workspace in Fig. 4 the connectivity tree is shown in Fig. 5. A connectivity tree is determined for $\ell = [1, \ldots, \mathcal{L}]$ layers. Layers are defined by their adjacency position relative to the initial cell. In Fig. 5 six layers are formulated for the workspace in Fig. 4. For rotating robots, the arcs and node sequences in the tree are determined by the adjacency relationships between the rotation intervals $\mathcal{I}_u$ in the rotation plane.

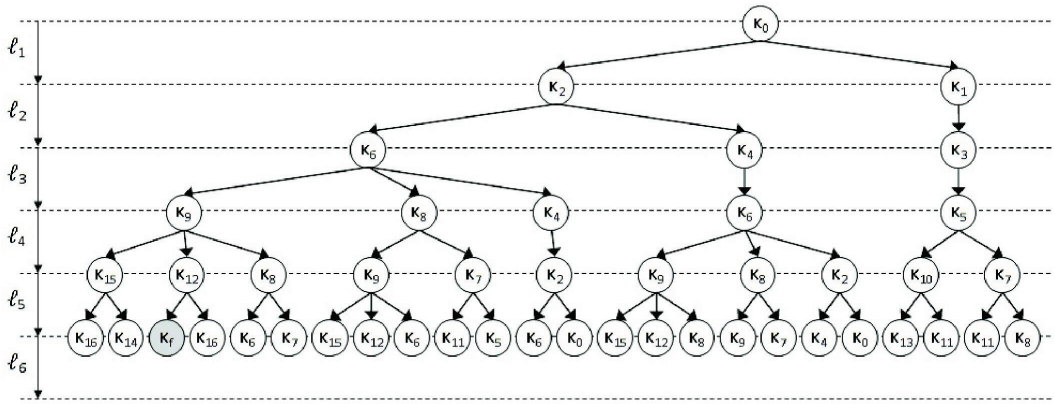

Fig. 5. Example of connectivity tree, $\mathcal{T}$, obtained for $\mathcal{W}$, κ_0, and κ_f shown in Fig. 4.

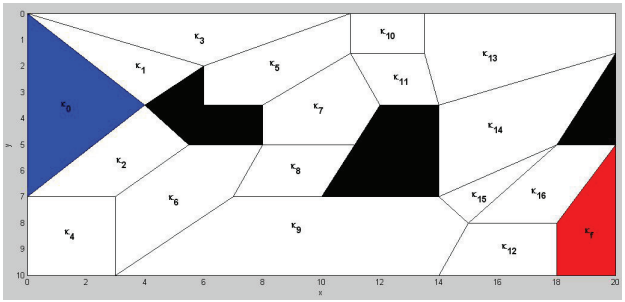

Fig. 4. Example of robot workspace, $\mathcal{W}$.

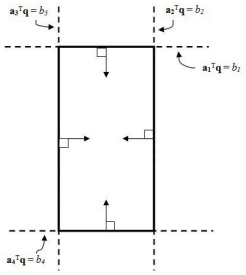

Fig. 6. Equality constraints representing the boundaries of a void rectangloid cell.

B. Disjunctive Programming Representation of the Connectivity Tree

Once the connectivity tree has been obtained by the procedure described in Section IV-A, it can be used to write the disjunctive program for the robot's path. Unlike previous DP approaches, in this paper the robot is required to be inside at least one cell at all times. Each cell is a rectangloid with boundaries that can be easily represented by linear equality constraints, as illustrated in Fig. 6. The DP inequality constraints, in this paper, require the robot to be inside a cell in $\kappa \in \mathcal{K}_{\text{void}}$ to avoid collisions, i.e.,

$$\bigwedge_{s=1,\dots,S(\kappa)} \mathbf{a}_s^T \mathbf{q}_i(t) \geq b_s \tag{7}$$

where $S(\kappa)$ is the number of boundaries of $\kappa \in \mathcal{K}_{\text{void}}$, and $\mathbf{a}_s^T$ and b_s are known coefficients, as shown in Fig. 6. The above conjunction of inequalities represents the statement that at time t, the t^{th} robot configuration, $\mathbf{q}_i(t)$, must be inside the void cell κ. Then, a layer in the tree can be represented by a disjunction of clauses, where each clause represents a node and, thus, a conjunction of inequalities in the form (7).

For a connectivity tree, $\mathcal{T}$, each layer, ℓ consists of a set of cells that the robot configuration can exist in for a corresponding time interval. The set of cells, $\mathcal{J}^\ell[\kappa]$ (with length $\Lambda(\mathcal{J}^\ell[\kappa])$), is valid for the time interval $[t_\ell, t_{\ell+1}]$ in T, where $t_1 = 0$ and $t_{\ell+1} = T$. Therefore, for each layer, $\Delta t_\ell = t_{\ell+1} - t_\ell$ sample positions are determined. In order to avoid obstacles during a given time interval, $\mathbf{q}_i$ must lie in one of the cells in the layer, i.e., $\kappa[\mathbf{q}_i(t)] \; \exists \; \mathcal{J}^\ell[\kappa]$ for $t_\ell \leq t \leq t_{\ell+1}$. Thus, for layer ℓ in $\mathcal{T}$, the robot must satisfy the following set of inequalities in order to avoid collisions:

$$\forall t \in [t_\ell, t_{\ell+1}] : \bigvee_{\lambda=1,\dots,\Lambda(\mathcal{J}^\ell[\kappa])} \; \bigwedge_{s=1,\dots,S(\kappa)} \mathbf{a}_{s\lambda}^T \mathbf{q}_i(t) \geq b_{s\lambda} \tag{8}$$

This states that the robot must be inside at least one of the cells in the specified layer of the connectivity tree. For the approximate cell decomposition case, in which the cells are rectangloids, the inequalities can be represented as:

$$\forall t \in [t_\ell, t_{\ell+1}] : \bigvee_{\lambda=1,\dots,\Lambda(\mathcal{J}^\ell[\kappa])} \begin{cases} & \mathbf{q}_i(t)(1) \leq x'_\kappa \\ \text{and} & \mathbf{q}_i(t)(1) \geq x_\kappa \\ \text{and} & \mathbf{q}_i(t)(2) \leq y'_\kappa \\ \text{and} & \mathbf{q}_i(t)(2) \geq y_\kappa \\ \text{and} & \mathbf{q}_i(t)(3) \leq \theta'_\kappa \\ \text{and} & \mathbf{q}_i(t)(3) \geq \theta_\kappa \end{cases}$$

where $[\mathbf{q}_i(t)(1) \; \mathbf{q}_i(t)(2) \; \mathbf{q}_i(t)(3)] = [x_i(t) \; y_i(t) \; \theta_i(t)]$.

Then, for given initial and final positions, $\mathbf{q}_{0_i}$ and $\mathbf{q}_{f_i}$, an obstacle-free path for robot i, described in Section II, can be

computed from the following disjunctive program:

$$\begin{aligned}
\min \quad & \sum_{t=1}^{T} \left(\mathbf{q}_i(t) - \mathbf{q}_i(t-1)\right)^2 \\
& \Delta\mathbf{q}_{min} \leq \Delta\mathbf{q}_i \leq \Delta\mathbf{q}_{max} \\
& \mathbf{q}_1 = \mathbf{q}_{0_i} \text{ and } \mathbf{q}_T = \mathbf{q}_{f_i} \\
& \bigcup_{\ell=1}^{\mathcal{L}} \mathcal{J}^{\ell}[\kappa] \equiv \mathcal{T}(\mathbf{q}_{0_i}, \mathbf{q}_{f_i}) \\
\text{subject to} \quad & \forall \ell \in [1, \ldots, \mathcal{L}] : \forall t \in [t_\ell, t_{\ell+1}] : \\
& \bigvee_{\lambda=1,\ldots,\Lambda(\mathcal{J}^{\ell}[\kappa])} \bigwedge_{s=1,\ldots,S(\kappa)} \mathbf{a}_{s\lambda}^T \mathbf{q}_i(t) \geq b_{s\lambda}
\end{aligned}$$

The flexibility of DP allows other constraints to be incorporated into the program such as multi-agent avoidance [13], as well as kinematic and dynamic constraints of the system. This DP can be converted into a mixed-integer quadratic program (MIQP) using the method presented in [13]. Therefore the problem, for one rotating robot, can be represented as:

$$\begin{aligned}
\min \quad & \sum_{t=1}^{T} \left(\mathbf{q}_i(t) - \mathbf{q}_i(t-1)\right)^2 \\
& \Delta\mathbf{q}_{min} \leq \Delta\mathbf{q}_i \leq \Delta\mathbf{q}_{max} \\
& \mathbf{q}_1 = \mathbf{q}_{0_i} \text{ and } \mathbf{q}_T = \mathbf{q}_{f_i} \\
& \bigcup_{\ell=1}^{\mathcal{L}} \mathcal{J}^{\ell}[\kappa] \equiv \mathcal{T}(\mathbf{q}_{0_i}, \mathbf{q}_{f_i}) \\
\text{subject to} \quad & \forall \ell \in [1, \ldots, \mathcal{L}] : \forall t \in [t_\ell, t_{\ell+1}] : \\
& \forall \lambda \in [1, \ldots, \Lambda(\mathcal{J}^{\ell}[\kappa])] : \begin{cases}
\mathbf{q}_i(t)(1) \leq x'_\kappa - M * (B_{i\lambda}(t) - 1) \\
\mathbf{q}_i(t)(1) \geq x_\kappa + M * (B_{i\lambda}(t) - 1) \\
\mathbf{q}_i(t)(2) \leq y'_\kappa - M * (B_{i\lambda}(t) - 1) \\
\mathbf{q}_i(t)(2) \geq y_\kappa + M * (B_{i\lambda}(t) - 1) \\
\mathbf{q}_i(t)(3) \leq \theta'_\kappa - M * (B_{i\lambda}(t) - 1) \\
\mathbf{q}_i(t)(3) \geq \theta_\kappa + M * (B_{i\lambda}(t) - 1)
\end{cases} \\
& \sum_{\lambda=1}^{\Lambda(\mathcal{J}^{\ell}[\kappa])} B_{i\lambda}(t) = 1 \\
& B = 0 \text{ or } 1
\end{aligned}$$

where M is a large positive number and B is a set of binary variables. The methodology presented above can be easily solved using commercially available software such as TOMLAB/CPLEX [21], [22].

V. Simulation Results

As a first step, the method is illustrated for a workspace that can be decomposed using exact cell decomposition, as shown in Fig. 4, to obtain the connectivity tree shown in Fig. 5. For illustrative purposes, in this example, it is assumed that the robot is a point mass and the obstacles are polyhedral, such that the exact decomposition and corresponding connectivity tree can be easily illustrated. By neglecting the robot geometry one also neglects the rotations and, thus, the initial configuration $\mathbf{q}_0 = [2,\ 2]^T$ and the goal configuration $\mathbf{q}_f = [19,\ 9]^T$ represent the initial and goal coordinates of the point mass in $\mathcal{W}$ (Fig. 7). In this case, the connectivity tree, $\mathcal{T}$, has six layers, and takes the form shown in Fig. 5. Using the approach presented in this paper, the cells and causal relationships in $\mathcal{T}$ can be represented by disjunctive inequalities in a DP and, subsequently, the path planning problem can be solved via MIQP, as shown by the results in Fig. 7. It can be seen that, unlike existing approaches [13], the methodology presented in this paper uses disjunctions of inequalities that guarantee the robot is inside a convex cell, as opposed to outside a convex obstacles, thereby allowing to account for obstacles that are not necessarily convex. Unlike cell decomposition, this approach produces a smooth robot trajectory instead of a sequence of cells that must then be processed by another algorithm in a hierarchical fashion, and has the potential to account for nonholonomic dynamics.

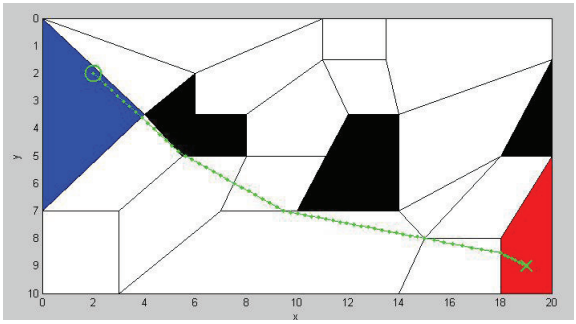

Fig. 7. Minimum-distance path for a robot in a workspace (Fig. 4) with concave polyhedral obstacles that allow for an exact cell decomposition of $\mathcal{C}_{free}$.

The methodology was then applied to a robot with a discrete geometry that is allowed to rotate, in a workspace populated by concave obstacles. As shown in Figs. 2-3 even if the robot geometry is a convex polytope the C-obstacles may amount to concave objects of any shape in 3D configuration space. In this case exact cell decomposition cannot be used to obtain a set of convex cells from $\mathcal{C}_{free}$, the approximate cell decomposition method in Section IV is applied to obtain rectangloid decompositions for a set of rotation intervals. For $[\theta^{min}, \theta^{max}] = [-90, 90]$, five intervals were used to calculate the approximate cell decomposition of a workspace, $\mathcal{W}$, shown in Fig. 8. As an example, for a robot with the rectangular geometry $\mathcal{A}$ shown in Fig. 9, the approximate decomposition plotted in Fig. 8 was obtained for the interval $\mathcal{I}^1 = [-90, -60]$. The full decomposition was then used to generate the connectivity graph, and a connectivity tree with six layers was obtained by pruning the connectivity graph for the given initial and final configurations. The path computed via MIQP for a robot, with $\mathbf{q}_0 = [3\ 8\ -75]^T$ and $\mathbf{q}_f = [7\ 2\ -75]^T$, is shown in Fig. 9 by plotting a subset of robot configurations. These results shows the effectiveness of this novel approach and how it combines cell decomposition and disjunctive programming to account for the robot's rotation and geometry in an environment with concave and convex obstacles.

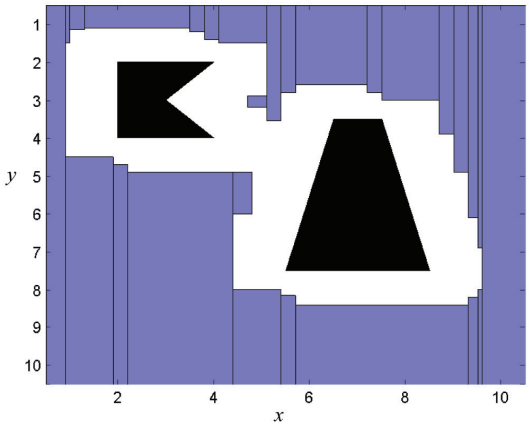

Fig. 8. Approximate cell decomposition for $\mathcal{I}^1 = [-90, -60]$.

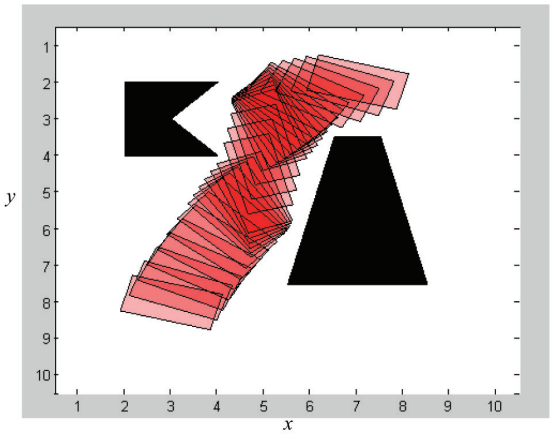

Fig. 9. Minimum distance path for a robot with a discrete geometry and rotations, in a workspace populated by concave obstacles.

As opposed to when approximate cell decomposition is used alone, the approach presented in Section IV can be used for planning the paths of multiple robots cooperatively, such that mutual collisions and collisions with obstacles are avoided. Initial methodology decomposed the obstacles rather than the free space. This is implemented for three rectangular robots in an obstacle-populated workspace that, due to the nature of the obstacles' and robots' geometries, can be decomposed exactly into convex rectangloid cells. By accounting for the robots' dynamics and positions in time, the MIQP approach computes minimum-distance trajectories that avoid mutual collisions, and the obstacles. It can be inferred from close inspection of Figs. 10-12 that at no point do the robots' geometries intersect. Fig. 12 plots the distances between each robot, which makes it apparent that no two robots approach each other close enough to cause a collision. By this method, nonholonomic dynamic constraints and multiple vehicles can be treated within a common trajectory planning framework that allows to simultaneously account for concave robot and obstacle geometries.

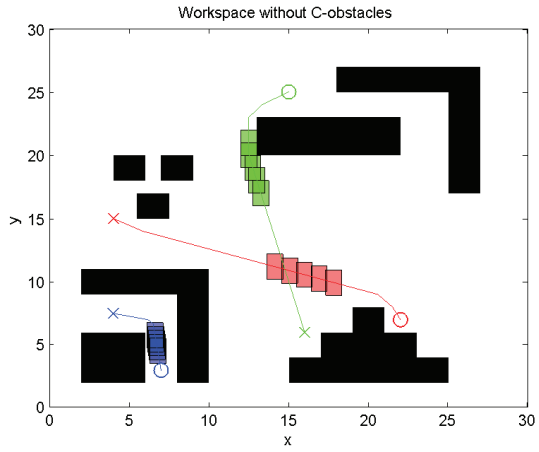

Fig. 10. Positions of three robots during a given time step

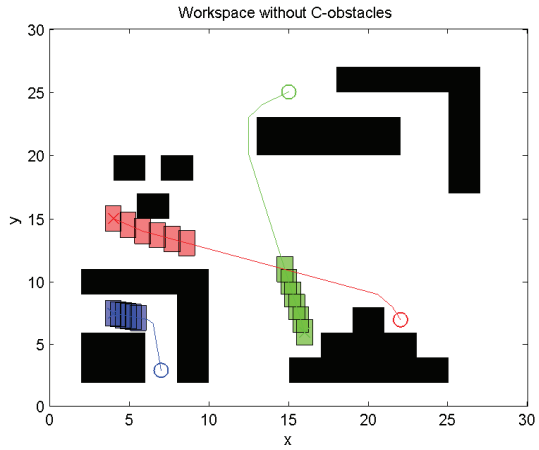

Fig. 11. Positions of three robots during an alternative time step to Fig. 10

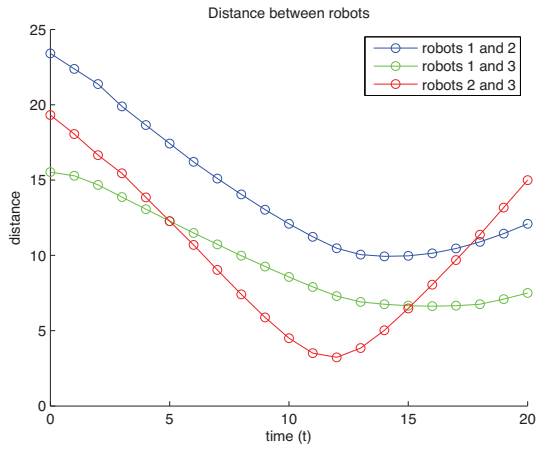

Fig. 12. Distances between the system of robots shown in Figs. 10-11 during time t

VI. Conclusions

This paper presents a novel approach for planning the minimum distance path of rotating robotic vehicles with discrete geometries, in an obstacle-populated workspace. The approach overcomes the limitations of existing cell decomposition and disjunctive programming approaches by allowing the consideration of multiple robots modeled by any differential equation, and environments populated by a large number of non-convex non-polyhedral obstacles. In order to produce programs that are computationally tractable, this approach derives a subset of all possible inequality constraints from a connectivity tree obtained by pruning the connectivity graph based on adjacency relationships between cells, and the principle of optimality. The approach is demonstrated through several numerical simulations involving multiple robots with discrete geometries, that are capable of rotating and thereby are characterized by three-dimensional, non-polyhedral C-obstacles.

References

[1] D. Zhu and J.-C. Latombe, "New heuristic algorithms for efficient hierarchical path planning," *IEEE Transactions on Robotics and Automation*, vol. 7, no. 1, pp. 9–20, 1991.

[2] J. Schwartz and M. Sharir, *On the 'Piano Movers' problem: II. General Techniques for Computing Topological Properties of Real Algebraic Manifolds*. New York, NY: Academic, 1983.

[3] K. Kedem and M. Sharir, "An efficient motion planning algorithm for convex polygonal object in 2-dimensional polygonal space," Courant Institute of Mathematical Science, New York, NY, Tech. Rep. 253.

[4] F. Avnaim, J. D. Boissonnat, and B. Faverjon, "A practical motion planning algorithm for polygonal objects amidst polygonal obstacles," INRIA, Sophia-Antipolis, France, Tech. Rep. 890.

[5] J.-C. L. D. Hsu and R. Motwani, "Path planning in expansive configuration spaces."

[6] L. E. Kavraki, P. Svetska, J. C. Latombe, and M. H. Overmars, "Probabilistic roadmaps for path planning in high-dimensional configuration space," *IEEE Transactions on Robotics and Automation*, vol. 12, no. 4, pp. 566–580, 1996.

[7] Y. Koren and J. Borenstein, "Potential field methods and their inherent limitations for mobile robot navigation."

[8] B. L. J. Barraquand and J.-C. Latombe, "Numerical potential field techniques for robot path planning."

[9] N. Ayanian and V. Kumar.

[10] M. Earl and R. D'Andrea, "Iterative milp methods for vehicle-control problems," in *IEEE Transactions on Robotics*, vol. 21, no. 6, 2005, pp. 1158–1167.

[11] O. Purwin and R. D'Andrea, "Path planning by negotiation for decentralized agents," in *American Control Conference*, 2007, pp. 5296–5301.

[12] A. Richards and J. P. How, "Aircraft trajectory planning with collision avoidance using mixed integer linear programming," in *Proceedings of the American Controls Conference*, pp. 1936–1941.

[13] J. H. T. Schouwenaars, B. De Moor and E. Feron, "Mixed integer programming for multi-vehicle path planning," in *Proceedings of the European Control Conference*, 2001, pp. 2603–2608.

[14] H. Li and B. Williams, "Generalized conflict learning for hybrid discrete/linear optimization."

[15] L. Blackmore and B. Williams, "Optimal manipulator path planning with obstacles using disjunctive programming," *Proceedings of the American Control Conference*, 2006.

[16] D. P. Bertsekas, *Dynamic Programming and Optimal Control, Vols. I and II*. Belmont, MA: Athena Scientific, 1995.

[17] J. C. Latombe, *Robot Motion Planning*. Kluwer Academic Publishers, 1991.

[18] I. E. Grossman, "Review of nonlinear mixed-integer and disjunctive programming techniques," *Optimization and Engineering*, vol. 3, pp. 227–252, 2001.

[19] T. S. A. Richards, J. How and E. Feron, "Plume avoidance maneuver planning using mixed integer linear programming," in *Proceedings of AIAA Guidance Navigation and Control Conference*, 2001.

[20] S. Ferrari and C. Cai, "Information-driven search strategies in the board game of CLUE®," *IEEE Transactions on Systems, Man, and Cybernetics - Part B*, vol. in review, 2008. [Online]. Available: http://fred.mems.duke.edu/silvia.ferrari/SMCCLUEarticle.pdf

[21] A. G. K. Holmstrom and M. Edvall, *User's Guide for Tomlab 7*, TOMLAB OPTIMIZATION. [Online]. Available: http://tomopt.com/docs/TOMLAB.pdf

[22] *ILOG CPLEX 9.0 User's Manual*, ILOG, 2003.