2008 IEEE International Conference on
Robotics and Automation
Pasadena, CA, USA, May 19-23, 2008

Decentralized Feedback Controllers for Multi-Agent Teams in Environments with Obstacles

Nora Ayanian and Vijay Kumar

Abstract— **We propose a method for synthesizing decentralized feedback controllers for a team of multiple heterogeneous agents navigating a known environment with obstacles. The controllers are designed to drive agents with limited team state information to goal sets while avoiding collisions and maintaining specified proximity constraints. The method, its successful application to nonholonomic agents in dynamic simulation, and its limitations are presented in this paper.**

I. Introduction

There are many applications where it is necessary to guide multiple vehicles to destinations or goal sets in an environment with obstacles while avoiding collisions and maintaining proximity constraints, either for communication or for sensing. In these situations it may be infeasible for each agent to have state information of the entire team. While there is extensive work on controlling formations [1]–[9], this work does not guarantee that formation constraints are maintained in the presence of obstacles. The synthesis of feedback controllers for vehicles in environments with obstacles are considered by [10]–[17]. However, most of these papers do not consider multi-vehicle constraints. In other words, the algorithms are not explicitly designed to prevent collisions and maintain communication between agents; those which do [16], [17] require tedious hand-tuning of parameters. Some methods of maintaining relative constraints seen in the literature include specifying a leader or setting a priority or agent hierarchy [18], [19], maintaining a rigid formation [6], [7], [20], or adding inter-agent repulsive forces [21], [22].

In this paper, we consider the problem of synthesizing feedback controllers for a team of multiple heterogeneous agents navigating a known environment with obstacles. We want controllers that are guaranteed to drive agents with limited communication to goal sets while avoiding collisions and maintaining specified proximity constraints. We are particularly interested in guiding a heterogeneous team of aerial and ground vehicles in an urban environment to desired goal sets with constraints on relative configurations.

One approach to solving the problem is to use *navigation functions* [10] to synthesize nonlinear feedback controllers that guarantee safety (obstacle avoidance) and global convergence. Extensions of navigation functions to the multi-vehicle case have been addressed by [16], [23], [24]. While this approach has the advantage of resulting in controllers with almost global convergence and smooth feedback, it is tedious for complex spaces, involves nonlinear equations, and requires hand-tuning of parameters.

A second approach is to decompose the configuration space into cells and synthesize controllers in individual cells in a way that guarantees a path between cells from any starting configuration to the goal configuration. Variations on this theme have been reported for piecewise affine systems [13], [14], holonomic systems [11], [25], and non holonomic systems [26]. While not all papers consider obstacles, it is possible to construct cells that are in the obstacle free space and feedback controllers that ensure smooth transitions between cells [11], [15], [25]–[27]. However, these papers do not address the problem of coordinating multiple agents.

Our method is similar in spirit to this second approach. The basic idea is to combine the information about individual Euclidean configuration spaces to construct the configuration space for all n agents. We eliminate from this set configurations that result in collisions or violate our specified proximity constraints to generate a team configuration space. We construct decentralized feedback controllers for point robots in cells to guarantee convergence to goal sets while satisfying all specified constraints. The controller is then extended to nonholonomic robots by feedback linearization. As we will show, the special case where communication is not limited results in a *complete* method. In other words, if there is a solution to the centralized navigation problem, our method will find feedback controllers that will achieve the desired task. Finally, in contrast to previous papers on multi-vehicle coordination [16], [21], [23], our method for synthesizing feedback controllers is completely automated.

We allow for a wide range of task specifications. The most important is collision avoidance. We can also specify *connectivity constraints* between designated pairs of agents that specify a maximum distance between these agents. We may wish to enforce an *exclusion constraint*: there can only be one agent in specified regions. We can also allow *logical combinations of goal configurations*. By this we mean that if there exist n agents and $l \leq n$ goals in the configuration space, then we can require any l of the n agents to achieve the goal positions, or specify tasks for combinations of agents.

The basic approach is as follows. First, we combine the information about the n agent configuration spaces and connectivity and collision constraints to generate the team configuration space, $\mathcal{C}_T$. In this work, we chose to represent the space as a union of polytopes, each of which is described

This research was supported by NSF grants CNS-0410514 and IIS-0427313 and ARO grants DAAD19-02-01-0383 and W911NF-04-1-0148.

N. Ayanian and V. Kumar are with the GRASP Laboratory, Department of Mechanical Engineering and Applied Mechanics, University of Pennsylvania, Philadelphia, PA 19104, USA {nayanian,kumar}@grasp.upenn.edu

N. Ayanian gratefully acknowledges support from the NSF. Special thanks to Nathan Michael for use of his Matlab API and help with experiments.

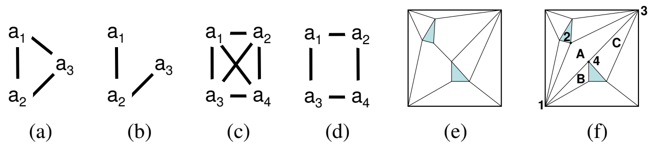


Fig. 1: Panels (a-d) show sample connectivity graphs. Panel (e) shows a legal decomposition, and panel (f) shows an illegal decomposition (inconsistent number of facets on (1,3)).

by an intersection of half-spaces in d-dimensions. We derive a discrete graph on the set of polytopes in which each edge represents an allowable path consistent with the task specifications. We then find a discrete path on this graph to the goal set and derive feedback controllers to drive any state in each polytope (except the goal polytope) to the next polytope on the path. Finally, we derive an appropriate feedback controller for the goal polytope to drive the system to its desired configuration. The controller synthesis is based on the work of Habets and van Schuppen [13].

The outline of the paper is as follows. We first introduce some basic definitions and formulate the problem in Section II. In Section III we determine a discrete path through the space and describe in detail the controller synthesis. We present some simulations and experiments in Section IV, and discuss the complexity of our algorithm in Section V. We conclude with Section VI.

II. Problem Formulation

Consider a team of n kinematic agents $V_A = \{a_i | i = 1, \ldots, n\}$ which, starting from some initial configuration, must reach some goal configuration while maintaining communication between specified agents and without colliding with each other or obstacles in the space. The agent a_i has the configuration or state $x_i \in \mathbb{R}^{d_i}$ with the dynamics:

$$\dot{x}_i = u_i,\ x_i \in X_i \subset \mathbb{R}^{d_i},\ i = 1, \ldots, n. \tag{1}$$

The agents have a predetermined *connectivity graph* whose edges denote constraints on proximity that must be maintained for communication of state information between specified agents and a *collision graph* whose edges describe minimum-distance safety constraints.

Recall that a *graph* is a pair of sets $G = (V, E)$, where $V = \{v_1, ..., v_n\}$ is the set of vertices or nodes and $E \subseteq [V]^2$ is the set of edges on the graph. Pairs of vertices for which $(v_i, v_j) \in E$ are called adjacent. A graph in which all pairs of vertices are adjacent is called a complete graph.

Definition 2.1: The *communication graph* (examples in Fig. 1a-d) on the set of agents is the static graph $G_N^\rho = (V_A, E_N)$ where E_N is the set of edges on the communication graph, describing pairs of agents which communicate state information, and ρ is a metric for determining inter-agent distances. We call pairs of agents which are adjacent on this graph *neighbors* or *neighboring agents*. To maintain communication, pairs $(a_i, a_j) \in E_N$ must maintain a maximum distance $|x_i - x_j|_\rho \leq \delta_{max}^{i,j}$. Agents receive state information only about neighboring agents. The constraint can be written

$$\nu_\rho(x_i, x_j) \leq 0 \quad \forall (x_i, x_j) \in E_N. \tag{2}$$

Definition 2.2: The *collision graph* on the set of agents is a static graph $G_L^\rho = (V_A, E_L)$ where E_L is the set of all pairs of agents which cannot occupy the same coordinates simultaneously. Pairs $(a_i, a_j) \in E_L$ must maintain a nonzero minimum distance $|x_i - x_j|_\rho \geq \delta_{min}^{i,j}$. We write the constraint

$$\lambda_\rho(x_i, x_j) \geq 0 \quad \forall (x_i, x_j) \in E_L. \tag{3}$$

For homogeneous agents occupying the same space, this graph will be complete. However, for heterogeneous agents, this graph may not be complete. For example, if we consider a two-dimensional configuration space for all vehicles, an aerial vehicle cannot collide with a ground vehicle or another aerial vehicle flying at a different altitude.

Definition 2.3: The *configuration space* $\mathcal{C}_i$ of an agent a_i is the set of all transformations of a_i. The *free space* $\mathcal{C}_i^{free}$ of a_i is the set of all transformations of a_i which do not intersect with obstacles in the configuration space.

We choose to decompose $\mathcal{C}_i^{free}$ into p_i polytopes with matching facets. As we will see later, this facilitates the controller synthesis technique we use in this paper. By matching facets we mean that any hyperplane supporting two adjacent polytopes shares the same vertices in both polytopes, and any hyperplane can only support one facet of that polytope. This is illustrated in Fig. 1(e-f),: the decomposition in panel 1e is legal since all facets match exactly on both sides, while the decomposition in panel 1f is illegal, since facet (1,3) has an extra vertex from polytopes B and C on the right.

Definition 2.4: The *team configuration space* is the Cartesian product of the configuration spaces of each agent,

$$\begin{aligned} \mathcal{C}_{all} &= \mathcal{C}_1^{free} \times \mathcal{C}_2^{free} \times \cdots \times \mathcal{C}_n^{free} \\ x &= [x_1, \ldots, x_n] \in \mathcal{C}_{all}. \end{aligned} \tag{4}$$

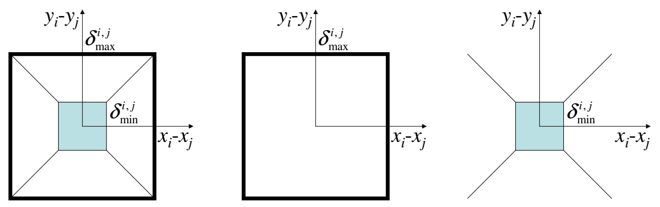
Thus the configuration of all of the n agents is described by a single point in $\mathcal{C}_{all}$. $\mathcal{C}_{all}$ has dimension $d = \sum_{i=1}^n d_i$ and contains $\prod_{i=1}^n p_i$ polytopes.

Definition 2.5: The *mutual exclusion graph* on the set of agents is the static graph $G_M = (V_A, E_M)$ where E_M is the set of all pairs of agents which cannot simultaneously occupy the same polytope in $\mathcal{C}_i = \mathcal{C}_j$ if they share the same decomposition, or in $\mathcal{C}_{free}$. The constraint can be written

$$\mu(x_i, x_j) = \begin{cases} 0, & (x_i, x_j) \in E_M \text{ in same polytope} \\ 1, & \text{otherwise.} \end{cases} \tag{5}$$

In general, the team of agents can be heterogeneous; thus they might not share the same configuration space, so $d_i \neq d_j$. However, in this paper, we will only consider examples in which all configuration spaces are projected onto a plane. Thus, while the configuration spaces are different for different agents (e.g., aerial versus ground), the dimensions of the configuration spaces are identical. From this point on we will let $d_1 = d_2 = \ldots = d_n$.

The *proximity constraints* specified by the connectivity graph $G_N^\rho = (V_A, E_N)$ and the collision graph $G_L^\rho = (V_A, E_L)$ are realized using ρ as the infinity norm which lends itself easily to convex decompositions. For pairs of agents $(a_i, a_j) \in E_N \cap E_L$ the intersection of these constraints corresponds to a square annulus in the relative space of two agents as in Fig. 2a (the shaded region denotes illegal configurations). Pairs $(a_i, a_j) \in E_N - E_L$ have only the maximum distance constraint (Fig. 2b), and pairs $(a_i, a_j) \in$



(a) Neighbors with collision constraint (b) Neighbors with no collision constraint (c) Collision constraint on non-neighbors

Fig. 2: Proximity Constraints. The shaded region indicates configurations that are not allowed.

$E_L - E_N$ have infinite annuli (Fig. 2c). The decompositions in Fig. 2a and 2c satisfy our legal decomposition criterion.

Definition 2.6: The *task configuration space* $\mathcal{C}_T$ is the set

$$\mathcal{C}_T = \mathcal{C}_{all} \cap \mathcal{L}_\rho \cap \mathcal{M} \cap \mathcal{N}_\rho, \tag{6}$$

where

$$\begin{aligned} \mathcal{L}_\rho &\equiv \{x | x \in \mathcal{C}_{all}, \lambda_\rho(x_i, x_j) \geq 0 \ \forall (a_i, a_j) \in E_L\}, \\ \mathcal{M} &\equiv \{x | x \in \mathcal{C}_{all}, \mu(x_i, x_j) = 1 \ \forall (a_i, a_j) \in E_M\}, \\ \mathcal{N}_\rho &\equiv \{x | x \in \mathcal{C}_{all}, \nu_\rho(x_i, x_j) \leq 0 \ \forall (a_i, a_j) \in E_N\}. \end{aligned} \tag{7}$$

$\mathcal{C}_T$ is a space composed of polytopes in which the agents cannot collide, lose communication or violate the mutual exclusion constraints.

Problem 2.7: For any initial state x_0, consider the system (1) on $\mathbb{R}^d$, with goal configuration $x^g \in X^g \subset \mathcal{C}_T \subset \mathbb{R}^d$ and metric ρ. Find a piecewise affine input function $u : [0, T_0] \rightarrow U$ for any $x_0 \in \mathcal{C}_T$ such that

1) $\forall t \in [0, T_0]$, $x \in \mathcal{C}_{all}$ and $x(T_0)$ arbitrarily close to x^g,
2) $\dot{x}_i = u_i$,
3) $x(t) \in \mathcal{L}_\rho \cap \mathcal{M} \cap \mathcal{N}_\rho$, $\forall t \in [0, T_0]$.

We can also replace x^g by the set X^g in the above problem statement to allow a set of goal configurations.

III. Feedback controllers on $\mathcal{C}_T$

In this section, we develop feedback controllers to solve Problem 2.7. In other words, we ensure that the agents are always inside the team configuration space $\mathcal{C}_T$ and they reach the goal configuration. There are two stages in this process. First, we pursue a discrete representation of the team configuration space and find paths in this discrete representation. Second, we translate these paths into feedback controllers.

The key step in the first stage is to define an adjacency graph on the set of polytopes.

Definition 3.1: The *polytope graph* $G_P = (V_P, E_P)$ on the polytopes in $\mathcal{C}_T$ is the pair of sets $V_P = \{c_1, \ldots, c_n\}$, where c_i is the centroid of the i-th polytope P^i, and E_P, the set of all pairs of polytopes which share a (matching) facet.

Using the polytope graph we determine a discrete path from every polytope to the goal polytope. When goal positions are not specifically assigned to agents (e.g. when only m of n agents are required to reach the goal set X^g) we have a finite number of goal nodes rather than one goal node.

Due to the communication requirements, we must triangulate the polytopes P^i into simplices s^i_j. The reason for this will become clear in the discussion of the controller we will use. We now define an adjacency graph on the set of simplices in each polytope.

Definition 3.2: The i-th *simplex graph* $G^i_S = (V^i_S, E^i_S)$ on the simplices s^i_j in polytope P^i is the pair of sets $V^i_S = \{c^i_1, \ldots, c^i_n\}$, where c^i_j is the centroid of the j-th simplex $s^i_j \in P^i$, and E^i_S, the set of all pairs of simplices which share a facet.

We use the simplex graph to determine a discrete path from each simplex in a polytope to simplices on its exit facet.

Theorem 3.3 (Necessary condition): Problem 2.7 has a solution only if the polytope graph, G_P is connected.
Proof: $\mathcal{C}_T$ contains every allowable configuration x in our polytopic world model. G_P contains all the information about the connectivity of $\mathcal{C}_T$. Thus, if there is a solution to Problem 2.7, there must exist a path from any node in G_P to the goal node(s). Conversely, if there is no path on the graph G_P between any two nodes, there is no solution to Problem 2.7.

Since there are multiple paths to the goal, we can use the usual notion of a distance on a graph to find the shortest paths in the polytope graphs using an algorithm like Dijkstra's algorithm. From our view point, the shortest path minimizes the number of polytopes that are visited; this in turn minimizes the number of transitions between polytopes. Similarly we find the shortest path in the same sense on the simplex graphs, to any of the simplices on the exit facet. In the goal polytope, P^g, we find the shortest path from any simplex in its triangulation to the goal simplex, s^g.

Once the paths on the polytope and simplex graphs are identified, we want to be able to synthesize decentralized feedback controllers to solve Problem 2.7. The synthesis procedure is similar in spirit to those discussed in [12]–[15], but is closest to the one developed by Habets and van Schuppen [13] for determining a centralized affine state feedback that satisfies a set of inequalities on a polytope. In their paper, they derive controllers that drive a linear system from any initial condition in a polytope through a desired exit facet in the polytope while guaranteeing the system does not leave the polytope. We slightly modify this algorithm to design decentralized piecewise affine controllers.

We now consider the subproblem of steering states in a simplex to a specified exit facet with limited state information.

Problem 3.4: Consider the affine system (1) on simplex $s^i_j \in P^i \in \mathcal{C}_T$, where s^i_j is the j-th simplex in P^i, the ith polytope on a path to the goal, and $x^g \notin s^i_j$. Let $\mathcal{F}_k$ be the facet shared by s^i_j and s^{j+1}_i with normal vector n_k pointing out of s^i_j. For any initial state $x_0 \in s^i_j$, we have to find a time-instant $T_i \geq 0$ and an input function $u : [0, T_i] \rightarrow U$ (where u is realized by the application of a continuous feedback law $u(t) = Fx + g$, $g \in \mathbb{R}^{2n}$) such that

1) $\forall t \in [0, T_i] : x(t) \in s^i_j$,
2) $x(T_i) \in \mathcal{F}_k$, and T_i is the smallest time-instant in the interval $[0, \infty)$ for which the state reaches facet $\mathcal{F}_k$,
3) $n^T_k \dot{x}(T_i) > 0$, i.e. the velocity vector $\dot{x}(T_i)$ at $x(T_i) \in \mathcal{F}_k$ has a positive component in the direction of n_k,
4) matrix F, composed of matrices $F^{ij} \in \mathbb{R}^{2 \times 2}, i, j = 1, \ldots, n$, is such that $F^{ij} = 0$ if $(a_i, a_j) \notin E_N$.

The special case of Problem 3.4 without restricting the choice of F matrices results in a centralized controller.

Theorem 3.5: The centralized case of Problem 3.4 has a solution if there exist inputs satisfying conditions (1) and (2) of Proposition 3.1 and (2c) of Theorem 4.7 in [13]. A solution is guaranteed if the system is fully actuated and the inputs are not constrained.

Proof: See [13].

Once in s^g, we solve the equation

$$\dot{x}(x = x^g) = Fx^g + g = 0. \tag{8}$$

Theorem 3.6 (Sufficient condition): Problem (2.7) has a solution if G_P is connected, there are no bounds on inputs, and communication between agents is not limited.

Proof: Follows from Theorem 3.5.

This constraint (4) on F limits the state information available to each agent: only if two agents are neighbors will they exchange state information. Because of the addition of these constraints on the solution of the linear program, we cannot guarantee that a solution will be found.

The constraint (4) can be easily formulated as a supplementary equality constraint on the linear program used to solve for the inputs at the vertices of each simplex. Without requirement (4), F and g are calculated after solving the linear program, by using the equation

$$\begin{bmatrix} F^{\mathrm{T}} \\ \hline g^{\mathrm{T}} \end{bmatrix} = \begin{bmatrix} {v_1}^{\mathrm{T}} & 1 \\ \vdots & \vdots \\ {v_{d+1}}^{\mathrm{T}} & 1 \end{bmatrix}^{-1} \begin{bmatrix} {u_1}^{\mathrm{T}} \\ \vdots \\ {u_{d+1}}^{\mathrm{T}} \end{bmatrix} = \begin{bmatrix} V^{\mathrm{T}} & 1 \end{bmatrix}^{-1} U. \tag{9}$$

Since we know certain entries of F must be zero, we restrict solutions of u accordingly. Define $W \equiv \begin{bmatrix} V^{\mathrm{T}} & 1 \end{bmatrix}^{-1}$. Then we impose $F_{i,j} = W_i \begin{bmatrix} u_{1,j} \cdots u_{d+1,j} \end{bmatrix}^{\mathrm{T}} = 0$ in the linear program.

In summary, the algorithm for controller synthesis or the solution to Problem 2.7 involves the following four steps:.

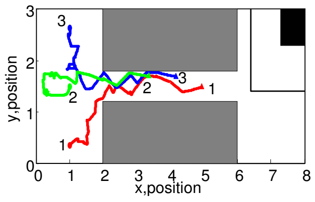
Algorithm 3.7:

1) Construct task configuration space $\mathcal{C}_T$ (Definition 2.6).
2) Find paths on polytope and simplex graphs G_P, G_S.
3) Solve Problem 3.4 on all simplices except s^g.
4) Solve Equation 8 in s^g.

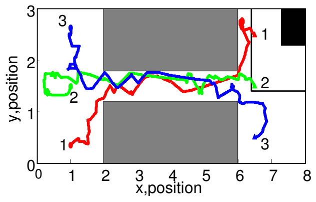
Using vector fields instead of potential or navigation functions for a controller has several advantages. The vector field approach, resulting in piecewise linear or piecewise affine feedback, is a computational approach instead of an analytical approach, so it can be used in any environment, while navigation functions are only formulated for star-shaped sets [10]. The vector field method can be automated, since all values can be determined with a linear program. In contrast, at least one parameter must be manually chosen for the navigation function method: the more complex the space, the more parameters must be chosen [10]. While the navigation function method does not scale, the vector field method scales exponentially (as described in Section V).

IV. Simulations and Experiments

In this section, we solve many multi-agent coordinated control problems to illustrate the application of the technique. The simulations run on either MATLAB alone, using the Multi-Parametric Toolbox for polytope computations, or on a MATLAB interface for Gazebo.

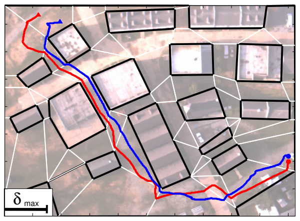


(a) Intermediate State

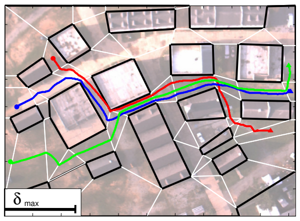


(b) Final State

Fig. 3: Three agents through a narrow corridor. The black boxes on the upper right show proximity constraints.



(a) Two ground agents



(b) Two ground agents and one UAV

Fig. 4: A team of unmanned vehicles navigate an urban environment. Buildings are enclosed by black polygons. The bar on the bottom left shows $\delta_{\max}$ for each simulation. In 4b, one agent is a UAV, which has a long range of communication.

A. *Three Agents Negotiate Passage through a Corridor*

Fig. 3a and 3b show a simulation in which the agents successfully traverse a corridor too narrow to allow the agents to traverse it in anything other than a single-file formation, making the proximity constraints difficult to preserve. Note that no formation is specified. Only the collision graph (complete) and connectivity graph (as in Fig. 1b) are part of the problem specification ($\delta_{min} = 0.7$, $\delta_{max} = 1.6$). Fig. 3a shows an intermediate stage, where all three agents are in the corridor.

B. *Two Agents Through a Complex and Real Space*

Fig. 4 shows a simulation of a real world problem where vehicles with realistic ranges of communication (150 m. in 4a and 250m. in 4b) navigate through an urban environment to their respective destinations while keeping within the specified range. Fig. 4a has two ground vehicles with complete connectivity and collision graphs. Fig. 4b shows two ground vehicles with one aerial vehicle with a very large communication range (greater than 1000 m.). All these plans were generated automatically using Algorithm 3.7.

C. *Three Agents in* MATLAB *and* GAZEBO

In these simulations, preprocessing of the controller is done in MATLAB. Three-dimensional dynamic simulation of the robots is done using GAZEBO, part of the PLAYER/STAGE/GAZEBO project [28]. GAZEBO is an open source multi-robot simulator, designed to accurately simulate a small population of robots with high fidelity. We use a MATLAB API which interacts directly with the GAZEBO to provide real-time control. For details on the model used in GAZEBO, and the conversion from a holonomic to a nonholonomic system, see [29].

A simple multiply connected space is shown in Fig. 5a and 5c. In these simulations, the agents share the same configuration space, collision graph (complete), start and goal configurations, and proximity constraints ($\delta_{min} = 0.7m$, and $\delta_{max} = 2.5m$). In the centralized case, the communication graph is complete; however, in the decentralized case (Fig. 5c and 5d), it is not complete (as in Fig. 1b, with agent 1 red, agent 2 green, and agent 3 blue). The distance between the solid lines and the black dotted lines at the starting point is due to feedback linearization distances, (centralized: 0.3, decentralized: 0.4). The agents take different routes to the goal since the feedback linearization points for the two cases at $t = 0$ are in different polytopes.

Fig. 5b and 5d show the distance between agents in each case. The first row corresponds to agents 1 and 2, the second to agents 2 and 3, and the third to agents 1 and 3. The left column shows Euclidean distance between pairs of agents; center and right columns show distance in the x-direction and y-direction, respectively. The plots show that proximity constraints are maintained. In the center plot in Fig. 5d, the red line depicting inter-robot distance dips below the dashed blue line, indicating that the robot distance has exceeded the limit; the feedback linearization point, however, does not.

D. Experiments with Two Agents

In experiments, we have used two SCARAB robots in a complex environment, pictured in Fig. 5e, interfacing with them with a similar MATLAB interface. Overhead cameras are used for robot tracking, which is accurate within 3 cm. We have used a feedback linearization distance of 18 cm, and padded the obstacles accordingly. Figures 5e and 5f shows the results of one experiment which is in the supplemental video. Figure 5f shows distance between the two robots and the distance between the feedback linearization points ($\delta_{min} = 0.5$m, $\delta_{max} = 3$m). This experiment shows that the controller can successfully be applied to real robot navigation problems.

V. COMPUTATIONAL COMPLEXITY

In this section we discuss the computational complexity of our method. Worst case complexity is mostly determined by the number of polytopes in $\mathcal{C}_T$, which scales exponentially with n. However, actual numbers observed are much lower.

$\mathcal{C}_{all} \in \mathbb{R}^d$ contains $\prod_{i=1}^n p_i$ polytopes, where p_i is the number of polytopes in $\mathcal{C}_i^{free}$. For each pair of agents with collision constraints we have one annulus with 4 regions, resulting in a maximum of $4^{n(n-1)/2}$ proximity regions intersected with $\mathcal{C}_{all}$. This results in a maximum of

$$P_{max} = 4^{n*(n-1)/2} \prod_{i=1}^{n} p_i \tag{10}$$

polytopes in $\mathcal{C}_T$. We must solve for a controller in each polytope in $\mathcal{C}_T$ to have a global solution to Problem 2.7.

This is a worst-case scenario, as some of these polytopes violate connectivity constraints, and the proximity constraints

TABLE I: Complexity of each step

Task	**Complexity**	**Ref**
Construct $\mathcal{C}_i^{free}$	$\mathcal{O}(v_i + \min\{v_i, r_i^2, r_i^4\})$	[30]
Construct $\mathcal{C}_T$	$\mathcal{O}(P_{\max} \cdot LP(h_i + 1, d))$	[31]
Plan on polytopes	$\mathcal{O}(\lvert E_A \rvert + P_{\max} \log P_{\max})$	[32]
Solve inequalities[a]	$\mathcal{O}(LP(\lvert V_i \rvert \lvert F_i \rvert - d, d^2))$	[33]
Triangulation[a]	$\mathcal{O}(\lvert V_i \rvert^{d/2})$	[31]
Plan in goal polytope P^g	$\mathcal{O}(\lvert E_S \rvert + \lvert S_i \rvert \log \lvert S_i \rvert)$	[32]
Solve inequalities[b] in P^g	$\mathcal{O}(LP(d^2 + d + 1, d^2))$	[33]
Solve F, g on polytopes[b]	$\mathcal{O}\left(\frac{2}{3}(d+1)^3 + 2d(d+1)^2\right)$	[33]

[a]per polytope, [b]per simplex

TABLE II: Critical values in our simulations

	$P_{\max}$	$P = \lvert \mathcal{C}_T \rvert$	$\sum_{i=1}^P V_i/P$	$\sum_{i=1}^P F_i/P$
Fig. 4a	$12,996$	2572	18	9
Fig. 5a	4096	464	92	15

are dependent ($x_1 < x_2, x_2 < x_3 \implies x_1 < x_3$). The polytopes in $\mathcal{C}_{all}$ which violate these constraints are eliminated in $\mathcal{C}_T$.

Table I specifies the complexity of every step in the process. Here, v_i (resp. r_i) is the total number of vertices (resp. reflex vertices) in $\mathcal{C}_i^{free}$, represented by a quasi-in-simple polygon. LP(m, p) represents the complexity of a linear program in p dimensions and m constraints. h_i is the number of inequalities used to describe the polytope after proximity constraints are added. $|V_i|$ (resp. $|F_i|$) is the number of vertices (resp. facets) in polytope P^i. $|S_i|$ is the number of simplices in the Delaunay triangulation of P^i.

The computational expense of the process depends largely on the methods used for decompositions, Cartesian products, and intersections, as well as the number of agents, connectivity, and complexity of the space.

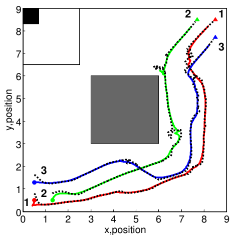
Table II presents critical values from our simulations. Although $P_{\max}$ is exponential, the actual number of polytopes in $\mathcal{C}_T$ is much lower. The table also presents average number of vertices and facets per polytope.

There are several ways to decrease the computation time required. Combinations of polytopes which violate proximity constraints can be ruled out before taking the Cartesian product. Additionally, given a limited number of polytopes which contain all possible initial configurations of the agents, solve only in those polytopes which they will pass through.

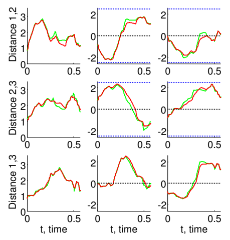
VI. CONCLUDING REMARKS

We presented a method for synthesizing feedback controllers for a team of multiple heterogeneous agents navigating a known environment with obstacles, and its application to navigation in urban environments. In MATLAB simulations, we showed the application of the algorithm to agents navigating a narrow corridor as well as an urban environment. In experiments as well as 3-D dynamic GAZEBO simulations, the controllers, although not specifically designed for nonholonomic robots, successfully drive agents with limited system state information to goal sets while avoiding collisions and maintaining specified proximity constraints. Additionally, we have shown in experiments that the controllers can be successfully applied to real robot navigation problems.

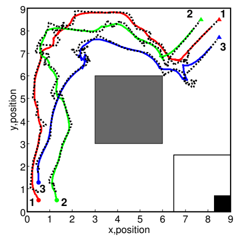
One limitation of this algorithm is that the complexity is exponential in the number of agents. However, we have



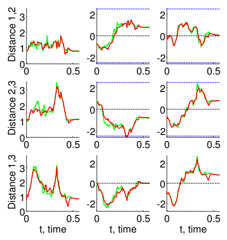
(a) Centralized Results



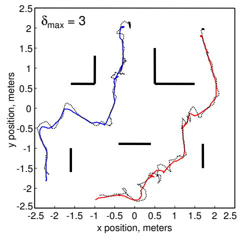
(b)



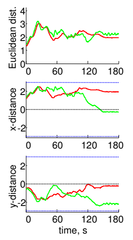
(c) Decentralized Results



(d)



(e) Experimental Results



(f)

Fig. 5: Simulations and an experiment in a multiply connected space on the SCARAB. In 5a, 5c, 5e, solid (dotted) lines show the position of the robot (feedback linearization point). Black boxes show proximity constraints (written in 5e). In 5b, 5d, 5f, red (green) depicts the robot position (feedback linearization point), dotted blue marks the maximum allowable distance.

briefly discussed two methods of decreasing the complexity. Additionally, the proximity constraint dependency combined with connectivity constraints significantly decrease the number of polytopes in the space, as shown in Table II.

There is potential to reduce complexity by considering that we are taking Cartesian products of identical graphs. By taking this into account, we may build the polytope graph with less computation. This is a direction for ongoing work, as is more extensive experimentation with multiple SCARAB robots.

REFERENCES

[1] V. Kumar, N. Leonard, and A. S. Morse, Eds., *Cooperative Control*, ser. Lecture Notes in Control and Information Sciences, vol. 309. Berlin Heidelberg New York: Springer-Verlag, 2004, http://www.cis.upenn.edu/ kumar/wcc/index.html.

[2] N. J. Cowan, O. Shakernia, R. Vidal, and S. Sastry, "Vision-based formation control," in *Intelligent Robots and Systems (IROS)*. Las Vegas, NV: IEEE/RSJ, October 2003.

[3] J. Desai, J. Ostrowski, and V. Kumar, "Modeling and control of formations of nonholonomic mobile robots," *IEEE Trans. on Robotics and Automation*, vol. 17, no. 6, Dec. 2001.

[4] H. Tanner, A. Jadbabaie, and G. J. Pappas, "Coordination of multiple autonomous vehicles," in *Proc. of IEEE Mediterranean Conf. Control and Automation*, Rhodes, Greece, June 2003.

[5] C. Belta and V. Kumar, "Towards abstraction and control for large groups of robots," in *Control Problems in Robotics, Springer Tracts in Advanced Robotics*. Berlin: Springer-Verlag, 2002.

[6] M. Egerstedt and X. Hu, "Formation constrained multi-agent control," *IEEE Trans. on Robotics and Automation*, vol. 17, no. 6, December 2001.

[7] M. C. D. Gennaro and A. Jadbabaie, "Formation control for a cooperative multi-agent system using decentralized navigation functions," in *Proceedings of the American Control Conference*, Minneapolis, Minnesota, June 2006.

[8] S. Zelinski, T. Koo, and S. Sastry, "Optimization-based formation reconfiguration planning for autonomous vehicles," in *Proc. IEEE Conf. Robotics and Automation*, Taiwan, September 2003.

[9] O. Orqueda and R. Fierro, "Robust vision-based nonlinear formation control," in *Proc. American Control Conf.*, June 2006.

[10] E. Rimon and D. Koditschek, "Exact robot navigation using artificial potential functions," *IEEE Trans. Robotics and Automation*, vol. 8, no. 5, October 1992.

[11] D. Conner, A. Rizzi, and H. Choset, "Composition of local potential functions for global robot control and navigation," in *Proc. IEEE Int. Conf. Intelligent Robots and Systems*, Las Vegas, Nevada, October 2003.

[12] D. Conner, H. Choset, and A. Rizzi, "Provably correct navigation and control for non-holonomic convex-bodied systems operating in cluttered environments," in *Robotics: Science and Systems*. Philadelphia, Pennsylvania: MIT Press, August 2006.

[13] L. Habets and J. van Schuppen, "A control problem for affine dynamical systems on a full-dimensional polytope," *Automatica*, vol. 40, no. 1, January 2004.

[14] B. Roszak and M. E. Broucke, "Necessary and sufficient conditions for reachability on a simplex," *Automatica*, vol. 42, no. 11, November 2006.

[15] S. Lindemann and S. Lavalle, "Computing smooth feedback plans over cylindrical algebraic decompositions," in *Robotics: Science and Systems*. Philadelphia, Pennsylvania: MIT Press, August 2006.

[16] S. G. Loizou and K. J. Kyriakopoulos, "Closed loop navigation for multiple holonomic vehicles," in *Intl. Conf. on Robotics and Automation*, Lausanne, Switzerland, October 2002.

[17] H. Tanner, S. Loizou, and K. Kyriakopoulos, "Nonholonomic navigation and control of cooperating mobile manipulators," *IEEE Trans. Robotics and Automation*, vol. 19, no. 1, February 2003.

[18] H. Tanner, G. J. Pappas, and V. Kumar, "Leader to formation stability," *IEEE Trans. on Robotics and Automation*, vol. 20, no. 3, June 2004.

[19] M. Ji, A. Muhammad, and M. Egerstedt, "Leader-based multi-agent coordination: controllability and optimal control," in *American Control Conference*, Minneapolis, Minnesota, June 2006.

[20] A. Das, R. Fierro, V. Kumar, J. Ostrowski, J. Spletzer, and C. J. Taylor, "Vision based formation control of multiple robots," *IEEE Trans. on Robotics and Automation*, vol. 18, no. 5, October 2002.

[21] M. A. Hsieh, S. G. Loizou, and V. Kumar, "Stabilization of multiple robots on stable orbits via local sensing," in *Accepted to IEEE International Conference on Robotics and Automation*, 2007, to appear.

[22] V. Gazi and K. Passino, "Stability analysis of social foraging swarms," *IEEE Trans. Systems, Man, and Cybernetics*, vol. 34, no. 1, February 2004.

[23] S. G. Loizou and K. J. Kyriakopoulos, "A feedback-based multiagent navigation framework," *International Journal of Systems Science*, vol. 37, no. 6, 2006.

[24] S. Loizou, D. Dimarogonas, and K. Kyriakopoulos, "Decentralized feedback stabilization of multiple nonholonomic agents," in *International Conference on Robotics and Automation*. IEEE, 2004.

[25] S. R. Lindemann and S. M. LaValle, "Smoothly blending vector fields for global robot navigation," in *IEEE Conference on Decision and Control*, Seville, Spain, 2005.

[26] S. R. Lindemann, I. I. Hussein, and S. M. LaValle, "Real time feedback control for nonholonomic mobile robots with obstacles," in *IEEE Conference on Decision and Control*, 2006.

[27] D. Conner, A. Rizzi, and H. Choset, "Construction and automated deployment of local potential functions for global robot control and navigation," Carnegie Mellon University, Robotics Institute, Pittsburgh, Pennsylvania, Tech. Rep. CMU-RI-TR-03-22, 2003.

[28] B. Gerkey, R. T. Vaughan, and A. Howard, "The player/stage project: Tools for multi-robot and distributed sensor systems," in *Proc. of Int. Conf. on Advanced Robotics*, Coimbra, Portugal, June 2003.

[29] N. Michael, J. Fink, and V. Kumar, "Experimental testbed for large multi-robot teams: Verification and validation," *IEEE Robotics and Automation Magazine*, Mar 2008.

[30] M. Keil and J. Snoeyink, "On the time bound for convex decomposition of simple polygons," in *Proceedings of the 10th Canadian Conference on Computational Geometry*, 1998.

[31] K. Fukuda, "Frequently asked questions in polyhedral computation," 2000, http://www.ifor.math.ethz.ch/ fukuda/polyfaq/polyfaq.html.

[32] T. H. Cormen, C. E. Leiserson, R. L. Rivest, and C. Stein, *Introduction to Algorithms*, 2nd ed. MIT Press and McGraw-Hill, 2001.

[33] S. Boyd and L. Vandeberghe, *Convex Optimization*. Cambridge, UK: Cambridge University Press, 2004.