Proceedings of the 2006 IEEE International Conference on Robotics and Automation
Orlando, Florida - May 2006

Hierarchical Abstractions for Robotic Swarms

Marius Kloetzer and Calin Belta
Center for Information and Systems Engineering
Boston University
Brookline, MA 02446
Email: kmarius, cbelta@bu.edu

***Abstract*— We develop a hierarchical framework for planning and control of arbitrarily large groups of fully actuated robots with polyhedral velocity bounds (swarm) moving in polygonal environments with polygonal obstacles. At the first level of hierarchy, we aggregate the high dimensional control system of the swarm into a small dimensional control system capturing its essential features. These features describe the position of the swarm in the world and its size. At the second level, we reduce the problem of controlling the essential features of the swarm to a model checking problem. In the obtained hierarchical framework, high level specifications given in natural language such as Linear Temporal Logic formulas over linear predicates in the essential features are automatically mapped to provably correct robot control laws.**

I. Introduction

The starting point for this paper is the observation that tasks for large groups of robots are "qualitatively" specified. This notion has a dual meaning. First, a swarm is naturally described in terms of a small set of "features", such as shape, size, and position of the region in plane or space occupied by the robots, while the exact position or trajectory of each robot is not of interest. Second, the accomplishment of a swarming mission usually does not require exact values for swarm features, but rather their inclusion in certain sets. For example, in the planar case, if the robots are constrained to stay inside an ellipse, there is a whole set of values for the pose and semi-axes of the ellipse which guarantees that the swarm will not collide with an obstacle of given geometry. Moreover, specifications for mobile robots are usually temporal, even though time is not necessarily captured explicitly. For example, a swarm might be required to reach a certain position and shape "eventually", or maintain a size smaller than a specified value "until" a final desired value is achieved. Collision avoidance among robots, obstacle avoidance, and cohesion are required "always". In a surveillance mission, a certain area should be visited "infinitely often".

Motivated by the above ideas, in this paper we present a computational method for planning and control of robotic swarms based on *abstractions*. Our framework is hierarchical. At the first level, we construct a *continuous abstraction* by extracting a small set of features of interest of the swarm. Even though the treatment in this paper is quite general, the focus is on a three-dimensional abstraction consisting of the mean and variance of the team, which lead to a description of the swarm position and size. At the second level of hierarchy, we map arbitrary LTL_{-X} formulas over linear predicates in the abstract variables to a control strategy in the abstract space, which is eventually projected back to the individual robots. We show that, for this particular abstraction, and under the assumption that the environment and the obstacles are polygonal, containment in the environment, swarm cohesion, and inter-robot and obstacle collision avoidance translate naturally to LTL_{-X} formulas over linear predicates in the continuous abstraction space. We also note that the semantics of LTL_{-X} over linear predicates in the abstract space is rich enough to capture temporal specifications such as the examples at the end of the previous paragraph. Our framework therefore allows for a rich spectrum of swarm specifications.

The continuous abstraction defined in this paper is inspired from [1], and inherits its invariance properties. One of the contributions of the present work is defining and proving its consistency. This paper also relates to results on reducing the dimension of control systems, such as the ones reported in [11], [15]. However, as opposed to [11], where the approach is time-abstract, our notion of consistency is stronger, and captures time explicitly. The problem of explicitly computing a timed trajectory for a control system from a trajectory of a lower dimensional (abstract) control system is considered in [15]. While focusing on simpler control systems, we generate trajectories for whole equivalence classes produced by the abstraction map, rather than one particular trajectory. From this point of view, our aggregation produces a bisimulation quotient, and relates to using foliations for constructing quotients as in [2]. The discrete abstraction of this paper is an application of results from [8], and it also relates to [14]. Recent works advocating the use of temporal logic in mobile robotics include [9], [12], [4]. One of the main contributions of this work is to show that a large class of robotic swarm specifications translate naturally to linear temporal logic, and a fully automated framework for generation of robot control laws can be constructed.

II. Preliminaries

Let $\Pi = \{\pi_1, \pi_2, \ldots, \pi_K\}$ be a finite set of atomic propositions. A linear temporal logic LTL_{-X} formula over Π is recursively defined as follows ([3]): (1) every atomic proposition π_i, $i = 1, \ldots, K$ is a formula, and (2) if ϕ_1 and ϕ_2 are formulas, then $\phi_1 \vee \phi_2$, $\neg\phi_1$, $\phi_1 \mathcal{U} \phi_2$ are also formulas.

The semantics of LTL_{-X} formulas are given over ω -words $w = w(1)w(2)w(3)\ldots$, where $w(i) \in 2^{\Pi}$, $i \geq 1$. Specifically, the satisfaction of formula ϕ at position $i \in \mathbb{N}$

of word w, denoted by $w(i) \vDash \phi$, is defined recursively as follows: (1) $w(i) \vDash \pi$ if $\pi \in w(i)$, (2) $w(i) \vDash \neg\phi$ if $w(i) \nvDash \phi$, (3) $w(i) \vDash \phi_1 \vee \phi_2$ if $w(i) \vDash \phi_1$ or $w(i) \vDash \phi_2$, and (4) $w(i) \vDash \phi_1 \mathcal{U} \phi_2$ if there exist a $j \geq i$ such that $w(j) \vDash \phi_2$ and for all $i \leq k < j$ we have $w(k) \vDash \phi_1$. A word w satisfies an LTL_{-X} formula ϕ, written as $w \vDash \phi$, if $w(1) \vDash \phi$.

The symbols $\neg$ and $\vee$ stand for negation and disjunction. The boolean constants $\top$ and $\bot$ are defined as $\top = \pi \vee \neg\pi$ and $\bot = \neg\top$. The other Boolean connectors $\wedge$ (conjunction), $\Rightarrow$ (implication), and $\Leftrightarrow$ (equivalence) are defined from $\neg$ and $\vee$ in the usual way. The *temporal operator* $\mathcal{U}$ is called the *until* operator. Formula $\phi_1 \mathcal{U} \phi_2$ intuitively means that (over a word) ϕ_2 will eventually become true and ϕ_1 is true until this happens. Two useful additional temporal operators, "eventually" and "always" can be defined as $\Diamond\phi = \top\mathcal{U}\phi$ and $\Box\phi = \phi\mathcal{U}\bot$, respectively. Formula $\Diamond\phi$ means that ϕ becomes eventually true, whereas $\Box\phi$ indicates that ϕ is true at all positions of w. More expressiveness can be achieved by combining the temporal operators. Examples include $\Box\Diamond\phi$ (ϕ is true infinitely often) and $\Diamond\Box\phi$ (ϕ becomes eventually true and stays true forever).

Let us now assume that the propositions π_i in Π are strict linear inequalities in $\mathbb{R}^n$, $n \geq 2$, *i.e.,* Π is given by

$$\Pi = \{\pi_i \,|\, \pi_i : c_i^T x + d_i < 0,\ i = 1, \ldots, K\}, \qquad (1)$$

where $c_i \in \mathbb{R}^n$ and $d_i \in \mathbb{R}$. Let $a : R_+ \to \mathbb{R}^n$ be a (possibly non-smooth) continuous curve in $\mathbb{R}^n$ (a is allowed to have self-intersections). We also assume that $c_i^T \bar{a} + d_i \neq 0$ and $c_i^T a(0) + d_i \neq 0$ for all $i = 1, \ldots, K$, where $\bar{a} = \lim_{t\to\infty} a(t)$ (if it exists). The semantics of an LTL_{-X} formula in Π over a continuous curve a follows naturally from the above definitions [8]. A *word* generated by a is a sequence $w_a = w(1)w(2)w(3)\ldots$, $w(i) \in 2^{\Pi}$, $i \geq 1$ obeying the following rules: (1) $w(1)$ is the set of all atomic propositions satisfied by $a(0)$, (2) A symbol $w(i) \neq w(i-1)$, $i \geq 2$ is added to w_a if there exist t_1, t_2, $0 \leq t_1 < t_2$ so that $a(t_1)$ satisfies all the propositions in $w(i-1)$, $a(t_2)$ satisfies all the propositions in $w(i)$, and $a(t)$ satisfies only propositions from $w(i-1) \cup w(i)$, for all $t_1 \leq t \leq t_2$, (3) An infinite number of symbols $w(i)$, $i \geq 1$ is added to w_a if the region represented by $w(i)$ is a "sink" for trajectory a, in the following sense: $\exists \tau > 0$ such that all and only propositions in $w(i)$ are satisfied by $a(t)$, $\forall t \geq \tau$. A trajectory $a : R_+ \to \mathbb{R}^n$ satisfies ϕ, written as $a \vDash \phi$ if and only if $w_a \vDash \phi$ (as defined above). Intuitively, a word produced by trajectory a is an enumeration of the sets of propositions from Π satisfied by $a(t)$ while time evolves.

III. Problem formulation and approach

Consider a set of N identical planar fully-actuated point-like robots described by

$$\dot{r}_i = u_i,\ r_i \in P,\ u_i \in U,\ i = 1, \ldots, N, \qquad (2)$$

where $r_i \in \mathbb{R}^2$ is the position vector of robot i in a world frame $\{F\}$ and u_i is its velocity, which can be directly controlled. $U \subseteq \mathbb{R}^2$ is a polyhedral set capturing the control constraints and P is a polygonal environment. Assume that P contains a set of polygonal obstacles O_j, $j = 1, \ldots, o$. When necessary, and as it will become clear from the context, we also use P and O_j to denote the propositional logic formulas describing the polygonal environment and the obstacles, respectively (they consist of conjunctions and disjunctions over linear inequalities in $\mathbb{R}^2$).

We collect all the robot states in $r = [r_1^T, \ldots, r_N^T]^T \in \mathbb{R}^{2N}$ (referred to as the configuration of the "swarm") and the robot controls in $u = [u_1^T, \ldots, u_N^T]^T \in \mathbb{R}^{2N}$. To recover the individual states and controls, we define the canonical projection $\pi_i(r) = r_i$, $\pi_i(u) = u_i$, $i = 1, \ldots, N$. Eqns. (2) can therefore be written as:

$$\dot{r} = u,\ \pi_i(r) \in P,\ \pi_i(u) \in U,\ i = 1, \ldots, N, \qquad (3)$$

Swarming tasks are specified in high level language in terms of a small set of properties to be satisfied by the swarm. Examples of such properties include *containment* of motion inside the environment P, *avoidance of obstacles* O_j, $j = 1, \ldots, o$, *cohesion* (*i.e.,* all pairwise distances smaller than a maximum predefined value), and *inter-robot collision avoidance* (*i.e.,* all pairwise distances larger than a minimum predefined value). In addition to these, the motion tasks are usually given in terms of temporal and logical specifications over a small set $a \in \mathbb{R}^n$, $n << N$ of essential features, while the exact values of r are not of interest. The essential features usually include information about the position, orientation, size, and shape of the region in plane spanned by the swarm. For example, assume that $a = (\mu, s) \in \mathbb{R}^3$, where $\mu \in \mathbb{R}^2$ gives the centroid of a square swarm and $s \in \mathbb{R}$ is its size (*e.g.,* area). If it is desired that the swarm converges to a configuration in which its centroid belongs to a polygon $P^d \subset P$ and with a size smaller than s^d, this can be written more formally as "eventually always ($\mu \in P^d$ and $s < s^d$)". If during the convergence to the final desired configuration it is necessary that the swarm visits a position $\bar{\mu}$ with a size $\bar{s}$, then the specification becomes "eventually (($\mu = \bar{\mu}$ and $s = \bar{s}$) and (eventually always ($\mu \in P^d$ and $s < s^d$)))". If in addition it is required that the size s is smaller than $\bar{s}$ for all times before $\bar{s}$ is reached, the specification changes to "$s < \bar{s}$ until (($\mu = \bar{\mu}$ and $s = \bar{s}$) and (eventually always ($\mu \in P^d$ and $s < s^d$)))".

The starting point for this work is the observation that such specifications translate naturally to LTL_{-X} formulas over linear predicates interpreted over trajectories of essential features a (as defined in Section II). For example, the last specification in the above paragraph corresponds to the formula $(s < \bar{s})\mathcal{U}(\mu = \bar{\mu} \wedge s = \bar{s}) \wedge (\Diamond\Box(\mu \in P^d \wedge s < s^d))$. In this paper we consider the following problem:

Problem 1: Identify a set of features a describing the region spanned by the swarm, and construct robot control strategies $u_i \in U$, $i = 1, \ldots, N$ so that:

(i) containment, obstacle avoidance, inter-robot collision avoidance, and cohesion are achieved, and
(ii) arbitrary LTL_{-X} formulas over arbitrary linear predicates in a are satisfied by all produced trajectories $a(t)$, $t \geq 0$.

To provide a solution to Problem 1, we propose a *hierarchical abstraction* approach. In the first level of abstraction, called *continuous abstraction*, we extract the essential features of the swarm by building a smooth surjective map

$$h : \mathbb{R}^{2N} \to \mathbb{R}^n,\ h(r) = a, \tag{4}$$

where h is called the (continuous) *abstraction*, or *aggregation*, or *quotient* map, and a is denoted as the *abstract state* of the swarm. In addition to providing a description of the swarm position, size, and shape, we will require h to perform a *correct aggregation* of the large dimensional state space $\mathbb{R}^{2N}$ of the swarm (see Section IV). If the state is correctly aggregated, then any trajectory $a(t) \in \mathbb{R}^n$ can be produced by the swarm. In the second level of abstraction, called *discrete abstraction*, we employ the method from [8] to generate control strategies in $\mathbb{R}^n$ so that arbitrary LTL_{-X} formulas over arbitrary linear predicates in $\mathbb{R}^n$ are satisfied by the abstract trajectories $a(t)$. A description of this method is given in Section V.

IV. Continuous abstraction

Let $u \in T\mathbb{R}^{2N}$ and $w \in T\mathbb{R}^n$ be two vector fields giving the full dynamics of the swarm

$$\dot{r} = u(r) \tag{5}$$

and its abstracted dynamics

$$\dot{a} = w(a), \tag{6}$$

respectively, where $a = h(r)$. Let $dh(r) : T_r\mathbb{R}^{2N} \to T_{h(r)}\mathbb{R}^n$ denote the differential (tangent) map of h at point r. If $h = (h_1, \ldots, h_n)$, then $dh(r)$ is a $n \times 2N$ real matrix whose rows are dh_i, $i = 1, \ldots, n$.

Definition 1 (h-related vector fields [13]): The vector fields $u \in T\mathbb{R}^{2N}$ and $w \in T\mathbb{R}^n$ are called h-related (h is the smooth surjection from Eqn. (4)) if

$$w(h(r)) = dh(r)u(r),\ \forall r \in \mathbb{R}^{2N} \tag{7}$$

and the following matching condition is satisfied

$$dh(r)u(r) = dh(r')u(r'),\ \forall r, r' \text{ with } h(r) = h(r'). \tag{8}$$

In the above definition, $dh(r)$ denotes the differential (tangent) of the map h at point r. The h - relation is an extension of the more used notion of push - forward, which is only defined when h is a diffeomorphism [13].

Definition 2 (Correct aggregation): The map (4) and the vector field (5) define a correct aggregation of the swarm if the following three properties are satisfied:

(i) Consistency: $h(r(t)) = h(r'(t))$, $\forall t \geq 0$, for all trajectories $r(t)$ and $r'(t)$ of (5) with $h(r(0)) = h(r'(0))$;
(ii) Actuation: The linear map $dh(r) : T_r\mathbb{R}^{2N} \to T_{h(r)}\mathbb{R}^n$ is surjective for all $r \in \mathbb{R}^{2N}$;
(iii) Detectability: $u(r) = 0$ if and only if $w(h(r)) = 0$, for all $r \in \mathbb{R}^{2N}$.

In other words, consistency means that swarm configurations which are equivalent with respect to the quotient produced by h remain equivalent for all times under the flow (5). This condition is necessary and sufficient to reduce system (5) to system (6), if the specifications for the trajectories of (5) are given in terms of $a = h(r)$, rather than explicitly in terms of r. Actuation guarantees that any velocity $w(a)$ (and therefore any motion) in the abstract space $\mathbb{R}^n$ can be achieved by the swarm. The detectability condition (iii) guarantees that the swarm does not spend energy in "uninteresting" motions. Indeed, $u(r) \neq 0$ and $w(h(r)) = 0$ would correspond to a motion of the swarm resulting in no change in the abstract state $a \in \mathbb{R}^n$, which captures the features of interest of the swarm.

Proposition 1: Given a vector field (5) and a map (4), the consistency condition (i) from Definition 2 is equivalent with the matching condition (8).

The proof of the above proposition is omitted for brevity and can be found in [7]. The actuation condition (ii) is equivalent to requiring that h be a submersion. In other words, h_i, $i = 1, \ldots, n$ are functionally independent, or equivalently, dh_i are linearly independent for all r, which again is equivalent to dh is full row rank for all r. Indeed, it is well known that a linear map is surjective if and only if it is full row rank.

The submersion h determines an orthogonal decomposition of $T_r\mathbb{R}^{2N}$ in $\mathcal{N}(dh(r))$ (of dimension $2N-n$) and $\mathcal{R}(dh^T(r))$ (of dimension n), where $\mathcal{N}$ and $\mathcal{R}$ denote the null space and range of a matrix, respectively. With this observation, the detectability condition (iii) is equivalent to restricting $u(r) \in \mathcal{R}(dh^T(r))$ [1]. We can now collect all these results in the following Theorem:

Theorem 1: [Correct aggregation] The smooth surjection (4) and the vector field (5) define a correct aggregation of the swarm if and only if

(i) The matching condition (8) is satisfied,
(ii) h is a submersion, and
(iii) $u(r) \in \mathcal{R}(dh^T(r))$.

If the conditions of Theorem 1 are satisfied, then arbitrary "abstract" vector fields $w(a)$ ($a = h(r)$) in $\mathbb{R}^n$ can be produced by "swarm" vector fields $u(r)$ using Eqn. (7), which is well defined since the aggregation is consistent. A particular solution of this equation is the minimum (Euclidean) norm solution

$$u(r) = dh^T(r)(dh(r)dh^T(r))^{-1}w(a),\ \forall r \in h^{-1}(a), \tag{9}$$

where $h^{-1}(a)$ is the equivalence class of a, or explicitly $h^{-1}(a) = \{r \in \mathbb{R}^{2N} | h(r) = a\}$. Note that $dh(r)dh^T(r)^{-1}$ is invertible for all r since h is a submersion. It is obvious that $u(r)$ given by (9) satisfies condition (iii) of Theorem 1. It also satisfies condition (i) since $dh(r)u(r) = w(a)$, for all $r \in h^{-1}(a)$. This result is summarized in the following Corollary of Theorem 1:

Corollary 1: If $w(a)$ is an arbitrary vector field in $\mathbb{R}^n$ and h is a submersion, then $u(r)$ from (9) and h define a correct aggregation of the swarm.

[1] There is a slight abuse of notation in this equation. dh_i are differential forms and their span is a co-distribution. However, when written in coordinates, they can be treated as vector fields, when their span is a distribution.

Remark 1: There is an interesting connection between consistency (Definition 2 (i)) and bisimilar quotients of continuous systems: the quotient system produced by the equivalence classes determined by h in $\mathbb{R}^{2N}$ is a bisimilarity quotient if and only if the aggregation is consistent. The interested reader is referred to [10] for general definitions of bisimilarity relations.

Finally, note that, if in addition to being linearly independent, dh_i, $i = 1, \ldots, n$ are orthogonal (in Euclidean metric), then Eqn. (9) assumes a particularly simple form

$$u(r) = \sum_{i=1}^{n} \frac{w_i(a)}{dh_i(r)dh_i^T(r)} dh_i^T(r) \tag{10}$$

V. Discrete abstraction

Once the large dimensional state of the swarm is correctly aggregated by properly choosing the aggregation map and the robot control laws, we have the freedom to assign arbitrary vector fields (6) in the abstract space $\mathbb{R}^n$. To provide a solution to Problem 1, the produced trajectories should satisfy arbitrary LTL_{-X} formulas over linear predicates in $\mathbb{R}^n$. To this goal, we use the computational framework for control of linear systems from LTL_{-X} specifications over linear predicates from [8]. In this section, we very briefly outline this procedure.

Let ϕ denote an arbitrary LTL_{-X} formula over linear predicates in $\mathbb{R}^n$ and let Π (Eqn. 1) be the set of all linear predicates appearing in ϕ. The framework described in [8] consists of two main steps. In the first, a finite state transition system is constructed. This construction starts with a proposition preserving partition of $\mathbb{R}^n$ into polytopes determined by feasible combinations of linear predicates from Π. The states of the transition system are the equivalence classes produced by the partition. Its transitions are determined by adjacency of polytopes and existence of affine feedback controllers making such polytopes invariant or driving all states in a polytope to an adjacent polytope through a common facet. The second step consists of producing runs of the transition system that satisfy formula ϕ. This is in essence a model checking problem. The algorithms in [8] return a set of initial states in the form of a union of polytopes in $\mathbb{R}^n$ and a feedback control strategy for (6) induced by the runs found using model checking. All trajectories $a(t)$ of the closed loop system satisfy ϕ as defined in Section II. The feedback controllers w can have different values at different times at the same state a. The produced trajectories are in general non-smooth, can have self-intersections, and are continuous in time. Arbitrary polyhedral control constraints $W \subset \mathbb{R}^n$ can be accommodated.

VI. Hierarchical abstraction based on mean and variance

In this section, we focus on a particular abstraction map $h : \mathbb{R}^{2N} \to \mathbb{R}^3$ ($n = 3$ in Eqn. (4)) given by

$$h(r) = a,\ a = (\mu, \sigma),\ \mu = \frac{1}{N}\sum_{i=1}^{N} r_i,$$
$$\sigma = \sqrt{\frac{1}{N}\sum_{i=1}^{N}(r_i - \mu)^T(r_i - \mu)}. \tag{11}$$

It is easy to see that h is smooth everywhere in $\mathbb{R}^{2N}$ except for the set $r_1 = r_2 = \ldots = r_N$, which corresponds to $\sigma = 0$ (all the robots coincide). In what follows, we exclude this degenerate case.We will show that this choice of an abstraction provides a useful and *fully automatic* solution to Problem 1. We first need to design robot control laws to make sure that the state of the swarm is correctly aggregated as described in Definition 2. By calculating the differential $dh(r)$ of h, it can be seen that h is a submersion (under the initial assumption that $\sigma \neq 0$). Moreover, dh_i, $i = 1, 2, 3$ are mutually orthogonal and $dhdh^T = \frac{1}{N} I_3$. Using Eqn. (9) and by canonical projection, we obtain the control laws for each robot in the form:

$$u_i(r_i, a) = \left[\ I_2 \quad \tfrac{r_i - \mu}{\sigma}\ \right] w(a),\ i = 1, \ldots, N, \tag{12}$$

where $w(a)$ is an arbitrary vector field in the abstract space $\mathbb{R}^3$. According to Corollary 1, the control laws (12) and the abstraction map (11) define a correct aggregation of the swarm.

Note that the control u_i of robot i depends on its own state r_i and on the abstract state a. A careful examination of (12) shows that all position vectors r_i, $i = 1, \ldots, N$ undergo the same affine transformation parameterized by the abstract variables μ and σ. By integration, Eqn. (12) leads to

$$u_i(r_i(0), a(0), a) = \left[\ I_2 \quad \tfrac{r_i(0) - \mu(0)}{\sigma(0)}\ \right] w(a), \tag{13}$$

$i = 1, \ldots, N$, where we emphasize that the control law u_i depends on the initial state of the robot, the initial value of the abstract state, and the current value of the abstract state (as opposed to the equivalent form in Eqn. (12) where the dependence was on the current state of the robot and of the abstract state).

A. Description of the region spanned by the swarm

Let $V \subseteq \{1, \ldots, N\}$ denote the set of indices of robots which are at the vertices of the convex hull of the swarm. Since the controls (12) determine an affine transformation, then V does not change in time. At time 0, we convert from the vertex to the hyperplane representation of the convex hull of the swarm:

$$\mathcal{P}_0 = conv\{r_i(0), i \in V\} =$$
$$\{x \in \mathbb{R}^2 \,|\, a_i^T(x - \mu(0)) + b_i \leq 0,\ i \in V\},$$

where $a_i \in \mathbb{R}^2$ are the unit outer normals to the facets of the polytope and $-b_i$ are the distances from $\mu(0)$ to the facets, $i \in V$ (note that $b_i \leq 0$). Since (12) correspond to a particular affine transformation consisting of translating and scaling by the same factor, the convex hull of the swarm at time t will be described by

$$\mathcal{P} = \{x \in \mathbb{R}^2 \,|\, a_i^T(x - \mu(t)) + \frac{\sigma(t)}{\sigma(0)} b_i \leq 0,\ i \in V\}. \tag{14}$$

It is important to note that description (14) of the region spanned by the swarm at time t is a conjunction of linear inequalities in the abstract variables $a(t) = (\mu(t), \sigma(t))$. The coefficients a_i, b_i, and $\sigma(0)$ are all determined at time 0 and constant during the motion.

B. Containment and obstacle avoidance

Recall from Section III that the polygonal environment and the obstacles are described by propositional logic formulas P and O_j, $j = 1, \ldots, o$ over linear predicates in the plane. Using description (14) of the area spanned by the swarm, containment and obstacle avoidance is guaranteed if the following first order formula is true:

$$\forall x : \mathcal{P} \Rightarrow (P \bigwedge_{j=1}^{o} \neg O_j) \tag{15}$$

Since $\mathcal{P}$ is linear in x and a, and P, O_j are linear in x, (15) is a formula in the logic of the reals with addition and comparison. Informally, the formulas of this first-order logic consist of linear inequalities with rational coefficients connected by logical and quantification operators. A basic computational feature of this logic is that any formula is equivalent to a quantifier-free formula, which can be effectively computed [16]. Let P_{co} denote a quantifier-free formula equivalent to (15), which is of course linear in the free variables μ and σ. Since containment in the environment and obstacle avoidance is desired for all times during a task, this leads to the following LTL_{-X} formula over linear predicates in μ and σ:

$$\text{containment and obstacle avoidance} : \Box P_{co} \tag{16}$$

C. Cohesion and inter-robot collision avoidance

Since all pairwise distances scale by the same factor $\sigma(t)/\sigma(0)$ under the affine transformation (12), the initial maximum and minimum pairwise distances remain maximum and minimum at any time. This leads to simple conditions for guaranteeing maximum and minimum distances between robots for all times, which we call cohesion and inter-robot collision avoidance, respectively. Let d_{min} and d_{max} denote specified minimum and maximum pairwise distances. Let $d^0_{min} \geq d_{min}$ and $d^0_{max} \leq d_{max}$ denote the minimum and maximum pairwise distances at the initial time $t = 0$. Then the predicate

$$P_d : \frac{d_{min}\sigma(0)}{d^0_{min}} \leq \sigma(t) \leq \frac{d_{max}\sigma(0)}{d^0_{max}} \tag{17}$$

guarantees that at time t all pairwise distances are between d_{min} and d_{max}. The specification that cohesion and inter-robot collision avoidance are required for all times becomes an LTL_{-X} formula over two linear predicates in the abstract variable σ:

$$\text{cohesion and inter} - \text{robot collision avoidance} : \Box P_d \tag{18}$$

Remark 2: Due to technical reasons that go beyond the scope of this paper, the LTL_{-X} control algorithm from [8] is restricted to formulas over strict inequalities as in Eqn. 1. Therefore, with the price of adding a bit of conservatism, we assume that the inequalities from P_{co} and P_d are strict. We also restrict the additional specifications (Problem 1 (ii)) to be given in terms of LTL_{-X} formulas over strict linear inequalities in μ and σ. From an application point of view, this assumption makes sense, since it is unreasonable to assume that a sensor could detect equality constraints.

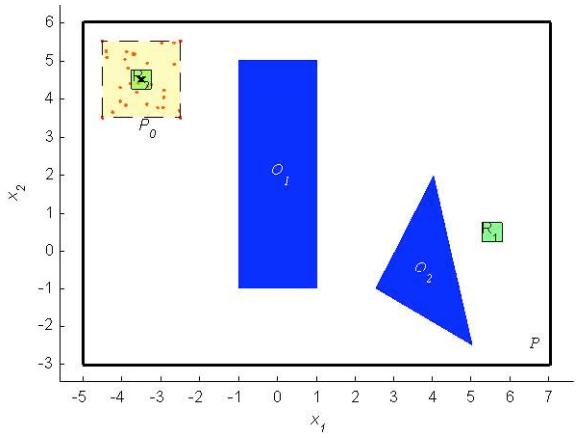

Fig. 1. Initial deployment of a swarm consisting of 30 robots in a rectangular environment P with two obstacles O_1 and O_2.

D. Robot control bounds

We now map the robot control constraints to constraints for the control of the abstract state. In other words, for an arbitrary polyhedral set $U \subset \mathbb{R}^2$, we construct a polyhedral set $W \subset \mathbb{R}^3$ with the property that $w \in W$ guarantees $u_i \in U$, $i = 1, \ldots, N$, where w is the velocity in the abstract space and u_i is the control of robot i, which are related by the linear map (13). Assume U is given in the hyperplane representation:

$$U = \{u \in \mathbb{R}^2 \,|\, f_k^T u + g_k < 0,\, k \in C\}, \tag{19}$$

where $f_k \in \mathbb{R}^2$, $g_k \in \mathbb{R}$, and C is some index set. Let us also denote by $A_i \in \mathbb{R}^{2\times 3}$, $i = 1, \ldots, N$ the matrix from Eqn. (13). Then it is easy to see that $u_i \in U$ if and only if $w \in W_i$, where

$$W_i = \{w \in \mathbb{R}^3 \,|\, f_k^T A_i w + g_k < 0,\, k \in C\} + \mathcal{N}(A_i), \tag{20}$$

and $\mathcal{N}$ denotes the null space of a matrix. Since the swarm undergoes an affine transformation, it can be proved that

$$u_i \in U,\ \forall i = 1, \ldots, N \Leftrightarrow w \in W = \bigcap_{j \in V} W_j. \tag{21}$$

E. Case study

Consider a swarm consisting of $N = 30$ robots moving in a rectangular environment P with two obstacles O_1 and O_2 as shown in Fig. 1. The initial configuration of the swarm is described by mean $\mu(0) = [-3.5,\ 4.5]^T$ and variance $\sigma(0) = 0.903$. The convex hull of the swarm is initially the square of center $\mu(0)$ and side 2 shown in the top left corner of Fig. 1. The cohesion requirement is given in terms of a maximum pairwise distance $d_{max} = 3.5$, while the inter-robot collision avoidance imposes $d_{min} = 0.01$. The control bounds for robots are captured by the set $U = [-2, 2] \times [-2, 2]$. The corresponding constraint set W as in Eqn. (21) is an octahedron in $\mathbb{R}^3$ with vertices (0,0,-2), (-2,-2,0), (-2,2,0), (2,2,0), (2,-2,0), (0,0,2). Let R_1 and R_2 be two square regions as shown in Fig. 1.

Consider the following swarming task given in natural language: *Always respect containment, obstacle avoidance,*

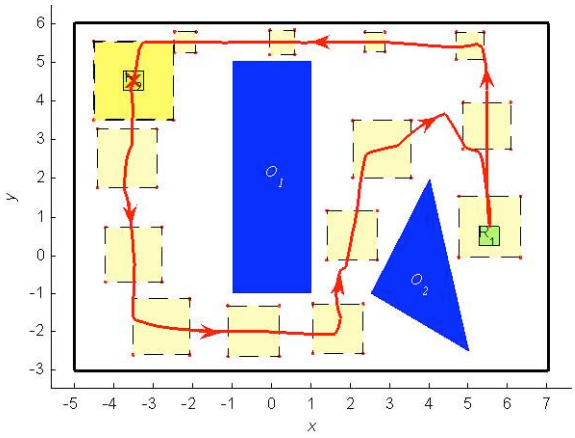

Fig. 2. Trace of the spanning polytope $\mathcal{P}$ (yellow) and trajectory of centroid μ (blue).

cohesion and inter-robot collision avoidance. In addition, the centroid μ must eventually visit region R_1. Until then, the minimum pairwise distance must be greater than 0.03*. After R_1 is visited, the swarm must reach such a configuration that its centroid is in region R_2 and the spanned area is greater than the initial one, and remain in this configuration forever.* This task translates to the following LTL_{-X} formula over linear predicates in μ and σ:

$$\phi = \Box(P_{co} \wedge P_d) \wedge \{(\sigma > 0.54)\mathcal{U}[(\mu \in R_1) \wedge \diamond\Box((\mu \in R_2) \wedge (\sigma > \sigma(0)))]\}, \quad (22)$$

where P_{co} and P_d were defined in Sections VI-B and VI-C, respectively, and $\sigma > 0.54$ corresponds to pairwise distance greater than 0.03. After eliminating the quantifier from formula (15), P_{co} consists of 27 occurences of 19 different linear predicates in μ and σ. By running the algorithms from [8], we conclude that there exists a trajectory in the abstract space $\mathbb{R}^3$ satisfying the formula from the initial values of μ and σ. *i.e.,* the task can be accomplished by the swarm. The trace of the spanning polytope $\mathcal{P}$ is given in Fig. 2, from which it can be seen that the specified task was accomplished.

F. Implementation

We developed a program for planning and control of robotic swarms in Matlab. Through a graphical interface, the package takes as input the polygonal environment P, the obstacles O_j, $j = 1, \ldots, o$, the control constraint set U, the initial positions of the robots, and an LTL_{-X} formula ϕ over linear predicates in the mean and variance of the swarm. The program tests the feasibility of the task, computes a control strategy, and displays the produced motion. From a computational point of view, four main steps are involved: (a) Quantifier elimination for calculation of formula P_{co} from Section VI-B. (b) Generation of the transition system from Section V. (c) Model checking of the transition system against the LTL_{-X} formula, and (d) Calculation of abstract controllers and generation of individual robot controllers. For (a), we used Redlog [16]. Steps (b) and (d) involve polyhedral set operations and triangulations for which we used CDD [5]. For (c) we used LTL2BA [6] and the well known Dijkstra's algorithm. However, the use of all these is transparent to the user, who interacts with the Matlab interface only. Due to space constraints, we omit a discussion on complexity issues, and refer the reader to [7].

VII. Conclusion

We proposed a fully automated framework for deployment of arbitrarily large swarms of fully actuated robots. Our approach is hierarchical. In the first level of the hierarchy, we aggregate the large dimensional state space of the swarm into a small dimensional continuous abstract space which captures essential features of the swarm. In the second level, we control the continuous abstraction so that specifications given in Linear Temporal Logic over linear predicates in the essential features are satisfied. Individual robot control laws are generated by projection. For planar robots with polyhedral control constraints moving in polygonal environments with polygonal obstacles, and a 3D continuous abstraction consisting of mean and variance, we show that a large class of specifications is captured.

References

[1] C. Belta and V. Kumar. Abstraction and control for groups of robots. *IEEE Trans. on Robotics*, 20(5):865–875, 2004.

[2] M. Broucke. A geometric approach to bisimulation and verification of hybrid systems. volume 1569 of *Lecture Notes in Computer Science*, pages 61–75. Springer-Verlag, 1999.

[3] E. A. Emerson. Temporal and modal logic. In J. van Leeuwen, editor, *Handbook of Theoretical Computer Science: Formal Models and Semantics*, volume B, pages 995–1072. MIT Press, 1990.

[4] G. E. Fainekos, H. Kress-Gazit, and G. J. Pappas. Temporal logic motion planning for mobile robots. In *Proceedings of the 2005 IEEE International Conference on Robotics and Automation*, April 2005.

[5] K. Fukuda. CDD/CDD+ package. URL http://www.cs.mcgill.ca/ fukuda/soft/cdd_home/cdd.html.

[6] P. Gastin and D. Oddoux. Fast LTL to Büchi automata translation. In H. Comon G. Berry and A. Finkel, editors, *Proceedings of the 13th Conference on Computer Aided Verification (CAV'01)*, number 2102, pages 53–65, 2001.

[7] M. Kloetzer and C. Belta. Linear temporal logic planning and control of robotic swarms. Technical Report CISE 2005-IR-0080, Boston University, 2005. http://www.bu.edu/systems/research/publications/2005/2005-IR-0080.pdf.

[8] M. Kloetzer and C. Belta. A fully automated framework for control of linear systems from LTL specifications. In *The 9th International Workshop on Hybrid Systems: Computation and Control*, Santa Barbara, CA, 2006. to appear.

[9] S. G. Loizou and K. J. Kyriakopoulos. Automatic synthesis of multiagent motion tasks based on LTL specifications. In *43rd IEEE Conference on Decision and Control*, December 2004.

[10] R. Milner. *Communication and Concurrency*. Prentice Hall, 1989.

[11] G. J. Pappas and S. Simic. Consistent abstractions of affine control systems. *IEEE Trans. on Automatic Control*, 47(5):745–756, 2002.

[12] M. M. Quottrup, T. Bak, and R. Izadi-Zamanabadi. Multi-robot motion planning: A timed automata approach. In *Proceedings of the 2004 IEEE Int. Conf. on Rob. and Aut.*, page 44174422, New Orleans, LA, 2004.

[13] M. Spivak. *A Comprehensive Introduction to Differential Geometry*. Publish or Perish, 1979.

[14] P. Tabuada and G. Pappas. Model checking LTL over controllable linear systems is decidable. volume 2623 of *Lecture Notes in Computer Science*. Springer-Verlag, 2003.

[15] P. Tabuada and G. J. Pappas. Hierarchical trajectory generation for a class of nonlinear systems. *Automatica*, 41(4):701–708, 2005.

[16] V. Weispfenning. A new approach to quantifier elimination for real algebra. Technical Report MIP-9305, Universität Passau, Germany, 1993.