Chapter 1

Abstractions for Planning and Control of Robotic Swarms

Calin Belta*

Department of Mechanical Engineering
Division of Systems Engineering
Boston University, Boston, MA 02446
E-mail : cbelta@bu.edu

In this chapter, we outline a hierarchical framework for planning and control of arbitrarily large groups (swarms) of robots. 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, such as position of the center, shape, orientation, and 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. We present simulation results for the particular case of a continuous abstraction based on centroid and variance of a planar swarm made of fully actuated robots with polyhedral control constraints.

As a result of tremendous advances in computation, communication, sensor, and actuator technology, it is now possible to build teams of hundreds and even thousands of small and inexpensive ground, air, and underwater robots. They are light, easy to transport and deploy, and can fit into small places. Such *swarms of autonomous agents* provide increased robustness to individual failures, the possibility to cover wide regions, and improved computational power through parallelism. However, planning and controlling such large teams of agents with limited communication and computation capabilities is a hard problem that received a lot of attention in the past decade.

It is currently believed that inspiration should be taken from the behavior of biological systems such as flocks of birds, schools of fish, swarms

*This work is partially supported by NSF CAREER 0447721 at Boston University.

of bees, or crowds of people. Over the past two decades, researchers from several areas, including ecology and evolutionary biology, social sciences, statistical physics, and computer graphics, tried to understand how local interaction rules in distributed multi-agent systems can produce emergent global behavior. Over the past few years, researchers in control theory and robotics have drawn inspiration from the above areas to develop distributed control architectures for multi-agent mobile systems. Even though, in some cases, it was observed or proved that local interaction rules in distributed natural or engineered multi-agent systems produce global behavior, the fundamental questions still remain to be answered. What are the essential features of a large group? How do we specify its behavior? How can we generate control laws for each agent so that a desired group behavior is achieved? In this chapter, we outline some ideas and results that could partially answer the above questions. The approach exploits the interplay between the "continuous" world of geometric nonlinear control and the traditionally "discrete" world of automata, languages and temporal logics.

1.1. Specification-induced hierarchical abstractions

The starting point for the approach presented here is the observation that tasks for large groups evolving in complex environments 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 often 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 chapter we briefly describe an approach 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. The result of the continuous abstraction will be a small dimensional continuous control system, whose dimension does not scale with the number of robots, and with constraints induced by the individual constraints of the robots. Intuitively, its state can be seen as the coordinates of a geometrical object spanning the area or volume occupied by the robots. An illustration is given in Figure 1.1 for a planar case, where the "spanning" object is an ellipse. In this example, if we can control the pose and semiaxes of the ellipse with the guarantee that the robots stay inside and their control and communication constraints are satisfied, then in principle we reduce a very high dimensional control problem to a five dimensional one (two coordinates for position of center, one for rotation, and one for each of the two semiaxes).

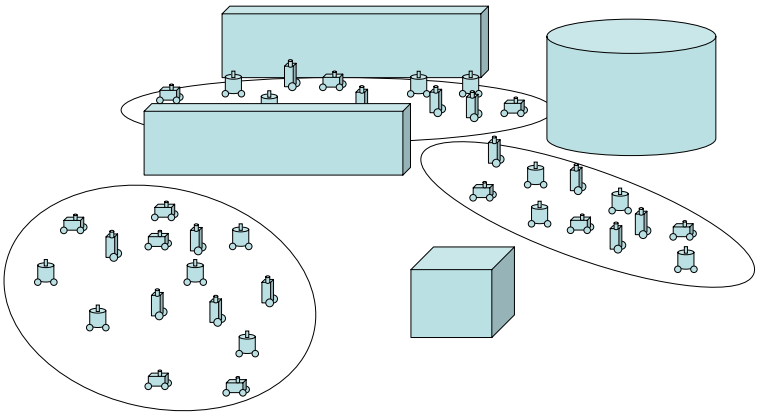


Fig. 1.1. Cooperative tasks for robotic swarms can be specified in terms of low dimensional abstractions capturing the pose and shape of geometrical objects describing the area (volume) occupied by the robots, such as spanning ellipses.

At the second level of the hierarchy, we use *discrete abstractions* to generate control strategies for the continuous abstraction from high-level, human-like task specifications given as temporal and logic statements about the reachability of regions of interest by the swarm. Simply put, through "smart" partitioning, discrete abstractions reduce the problem of controlling the continuous abstraction to a control problem for an automaton. In the discrete and finite world of automata, such human-like specifications

translate to temporal logic formulas, and the control problem resembles model-checking, which is just a more sophisticated search on a graph.

When combined, the two approaches lead to the **hierarchical abstraction architecture** shown in Figure 1.2, where the *continuous abstraction reduces the dimension of the problem by focusing on swarm cohesion and robot dynamics and constraints*, and the *discrete abstraction captures the complexity of the environment*. The geometry of the obstacles and the specifications of the swarming task determine a partition of the state space of the continuous abstraction produced at the first level. Discrete abstractions will then be used to check the existence and construct admissible control laws for the continuous abstraction, which captures the individual robot constraints. Therefore, in this hierarchical abstraction framework, "qualitatively" specified swarming tasks, such as temporal and logical statement about the reachability of regions of interest, are in principle reduced to analysis of finite state automata.

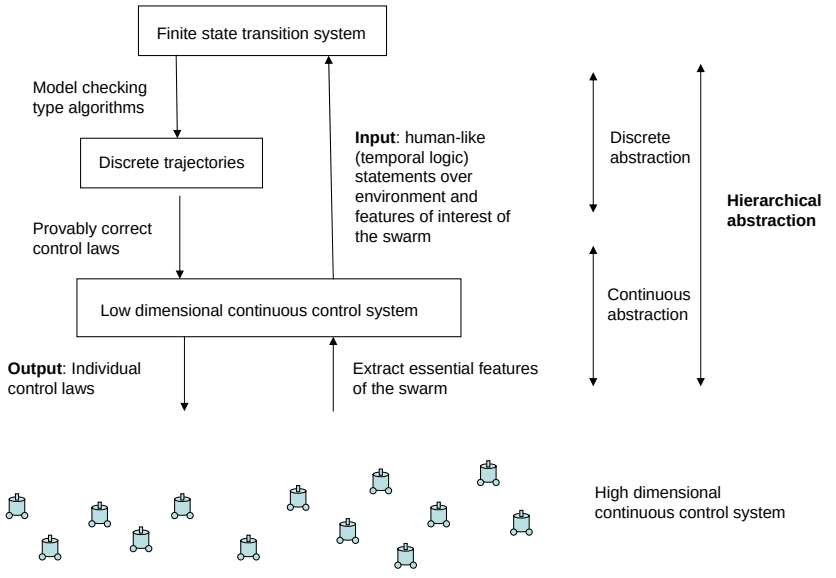


Fig. 1.2. Hierarchical abstraction architecture for planing and control of robotic swarms: high level specifications given in human-like language such as temporal logic formulas are used to construct a discrete and finite description of the problem. Tools resembling model checking are involved to find a solution for this problem, which is then implemented as a hybrid control strategy for the continuous abstraction. Individual robot control laws are then generated through projection.

1.2. Continuous abstractions: extracting the essential features of a swarm

Coordinates of geometrical objects describing the volume (or area) occupied by the robots can serve as abstract descriptions of the swarm. For example, for a planar task, controlling the pose (position and orientation) and semiaxes of a spanning ellipse, while making sure that the robots stay inside it, can be used for a large class of cooperative tasks. A pictorial illustration of this idea is given in Figure 1.1, where the ellipse can be shrunk, reshaped, and/or reoriented for avoiding obstacles.

Inspired by this idea, one can think of abstractions as having a product structure of a Lie *group* capturing the pose of the team in the world frame and a *shape*, which is an intrinsic description of the team, invariant to the world coordinate frame. In our example from Figure 1.1, the group would be the position of the center and the orientation of the ellipse in the world frame, while the shape would correspond to its two semiaxes. For such a group-shape representation of the swarm, it makes sense to require that the group part be controlled separately from shape. Indeed, the idea of having a coordinate free definition of the shape together with shape control law decoupled from group is fundamental in formation control. Without knowing the coordinates of the robot in the world frame, only shape variables can be measured and controlled using on-board sensors of the robots.[1,2] In addition, a description of the swarm should be invariant to robot permutations. This would lead to robustness to individual failures for the planning and control architecture to be developed.

To introduce the main ideas, let us assume for simplicity that the swarm is composed of N identical planar fully-actuated point-like robots with polyhedral control constraints. In other words, each robot is described by a control system of the form:

$$\dot{r}_i = u_i,\ u_i \in U,\ i = 1, \ldots, N, \tag{1.1}$$

where $r_i \in \mathbb{R}^2$ is the position vector of robot i in a world frame, u_i is its velocity, which can be directly controlled, and $U \subseteq \mathbb{R}^2$ is a set capturing the control constraints. We collect all the robot states in $r = [r_1^T, \ldots, r_N^T]^T \in \mathbb{R}^{2N}$ and the robot controls in $u = [u_1^T, \ldots, u_N^T]^T \in \mathbb{R}^{2N}$. We denote by Q the set of all swarm configurations, which equals $\mathbb{R}^{2N}$ if no obstacles and environment limits are imposed.

To construct a continuous abstraction with a product structure of group

and shape, we build a smooth surjective map

$$h : Q \to A,\ h(r) = a \tag{1.2}$$

where h is called the (continuous) *abstraction*, or *aggregation*, or *quotient* map and a is denoted as the *abstract state* of the swarm. We require A to have a product structure of a group G and a shape S, *i.e.,*

$$A = G \times S,\ a = (g, s),\ a \in A,\ g \in G,\ s \in S. \tag{1.3}$$

$g \in G$ and $s \in S$ are the group and the shape part of $a \in A$, respectively. The invariance to robot identifications translates to invariance of h to permutations of r_i. The invariance to world frame translate to left invariance for h, which simply means that, if a change occurs in the world frame, which can be modelled as an action by an element $\bar{g} \in G$, then in the image of the swarm configuration through the map h, the group part will be left translated by $\bar{g}$, while the shape part will not change.

The notions of invariance with respect to world frames and permutations of robots, which are fundamental for swarms, are properly approached in shape theory in the well known "n-body problem", traditionally studied in theoretical physics,[3] and more recently applied to robotics.[1] However, these works focus on maximal shape spaces (*i.e.,* largest subspaces of configuration spaces invariant to permutations and world frames), and are therefore restricted to very small teams of robots. To use such ideas for robotic swarms, one can start, as in the works enumerated above, from Jacobi vectors.[4] However, instead of computing maximal Jacobi spaces, one can try "democratic" sums over dot products and triple products of Jacobi vectors, as suggested in.[1] Jacobi vectors are already translation reduced, the dot products are invariant to rotations, and permutations of particles are easy to characterize as actions of the "democracy" group $O(N-1)$. Moreover, such sums can give useful estimates of the area occupied by the robots in the planar case, and of the volume of a spanning region in the spatial case.

Alternatively, one can investigate the eigenstructure of the *inertia tensor* of the system of particles r_i, $i = 1, \ldots, N$. Properly normed and ordered eigenvectors can give the rotational part of the group, while the eigenvalues (and physically significant combinations of them) are good candidates for shape variables. Related to this, one can think of a probabilistic approach, starting from the assumption that the vectors r_i are normally distributed. Then their sample mean and covariance converge to the parameters of a normal distribution when N is large enough. The corresponding *concentra-*

tion ellipsoid is guaranteed to enclose an arbitrary percentage of the robots inside it, and can therefore be used as a spanning region. If the distribution is not normal, higher central moments can be useful in defining abstractions as well. For example, the third moment (*skewness*) is a measure of how far the distribution is from being symmetric, while the fourth central moment (*kurtosis*) can be used as a measure of the coverage by the robots of the area of the ellipsoid.

In addition to providing a description of the swarm position, size, and shape that is invariant to permutations and world frames, we will require h to perform a *correct aggregation* of the large dimensional state space Q of the swarm. Let $u(r)$ and $w(a)$ be the vector fields giving the full dynamics of the swarm and of the continuous abstraction, respectively, with $a = h(r)$. In our view, a correct aggregation has three requirements. First, the flow $\dot{r} = u(r)$ and the quotient map (1.2) have to be *consistent*, which intuitively means that swarm configurations that are equivalent (indistinguishable) with respect to h are treated in an equivalent way by the flow of the swarm. Second, a correct aggregation should allow for any motion on the abstract space A, which we call the *actuation* property. Third, we do not allow the swarm to spend energy in motions which are not "seen" in the abstract space A (*detectability*). In,[5] using a geometrical approach, we provide necessary and sufficient conditions for correct aggregation. In short, the consistency of the flow $\dot{r} = u(r)$ and of the quotient map (1.2) is equivalent to the "matching condition" $dh(r)u(r) = dh(r')u(r')$, for all r, r' with $h(r) = h(r')$, where $dh(r)$ denote the differential (tangent) map of h at point r. The actuation property is equivalent to requiring h to be a submersion, while the detectability requirement is equivalent to restricting $u(r)$ to the range of $dh^T(r)$. Note that all these characterizations are all very simple conditions that can be easily checked computationally. In conclusion, if all the above conditions are satisfied, then arbitrary "abstract" vector fields $w(a)$ ($a = h(r)$) in A can be produced by "swarm" vector fields $u(r)$.

Examples of continuous abstractions In,[6] we defined a 5 - dimensional abstraction (1.2) that can be used for swarms of fully actuated planar robots. The abstraction satisfies all the requirements stated above. An extension to unicycles, based on a simple input-output regulation intermediate step, is described in.[7] In the abstraction $a = (g, s)$, the 3-dimensional group part is $g = (\mu, \theta) \in SE(2)$, where μ is the centroid of the swarm and $\theta \in S^1$ parameterizes the rotation that diagonalizes the inertia tensor

of the system of particles. The 2-dimensional shape $s \in \mathbb{R}^2$ is given by the eigenvalues of the inertia tensor. In this interpretation, the abstraction captures the pose and side lengths of a rectangle with the guarantee that the robots are always inside. Equivalently, if the robots are assumed normally distributed, μ can be interpreted as sample mean, θ as rotation diagonalizing the sample covariance matrix, while the shape variables s are the eigenvalues of the covariance matrix. In this second interpretation, the abstraction parameterizes an ellipse, with the guarantee that an arbitrary percent of the robots stay inside. A simulation result is shown in Figure 1.3, where a swarm of $N = 100$ robots (evolving on a 200-dimensional space!) is driven through a tunnel by designing controls on a 5 - dimensional space parameterizing a spanning ellipse.

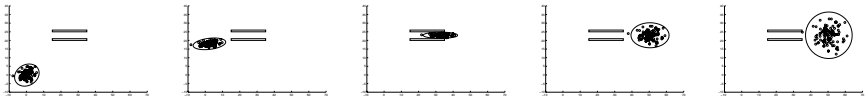


Fig. 1.3. 99 of $N = 100$ normally distributed planar robots are driven through a tunnel by designing 5 - dimensional controls for the corresponding equiprobability ellipse.

When orientation is not relevant for a certain application, we can define a simpler 3 - dimensional abstraction, by restricting the group g to the position of the centroid μ, and by collapsing the shape s to the sum of the two shape variables from above. In this case, the N robots described by a 3 - dimensional abstract variable $a = (\mu, s)$ are enclosed in a circle centered at μ and with radius proportional to s. In Figure 1.4, we show a simulation for controlling $N = 30$ robots using this simpler abstraction. Initially, the robots are distributed on three concentric circles.

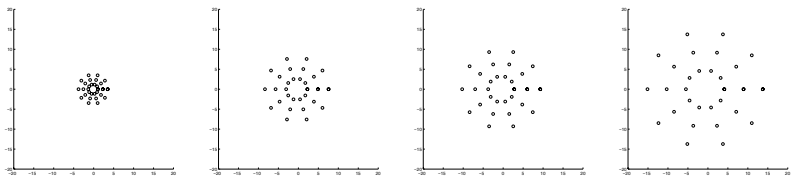
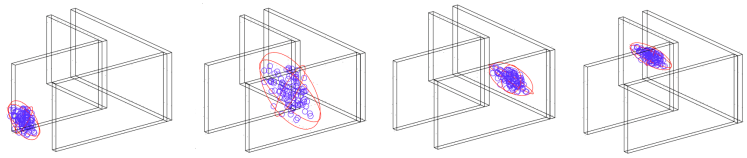


Fig. 1.4. $N = 30$ robots experiencing an expansion using control laws based on 3-dimensional abstraction. The centroid is kept fixed. Orientation, parallelism, angles, and ratios of lengths are preserved.

A generalization of these ideas to 3D environments is possible. In,[8] we constructed a 9-dimensional abstraction ($a = (g, s)$ for a swarm in 3D,



(a) (b) (c) (d)

Fig. 1.5. $N = 100$ robots pass a corridor by controlling the 9-dimensional abstraction parameterizing a concentration ellipsoid.

where the group part $g = (\mu, R) \in SE(3)$ consists of centroid $\mu \in \mathbb{R}^3$ and rotation $R \in SO(3)$ that diagonalizes the inertia tensor. As before, the shape $s \in \mathbb{R}^3$ captures the eigenvalues of the inertia tensor. In Figure 1.5, we show a simulation of controlling a swarm of $N = 100$ robots in space based on this 9-dimensional abstraction, which can be seen as parameterizing an ellipsoid spanning the volume occupied by the swarm.

1.3. Discrete abstractions: accommodating rich specifications

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 $w(a)$ in the abstract space A. These vector fields will be constructed from high-level temporal and logical specifications over the continuous abstraction a. For example, assume that $a = (\mu, s) \in \mathbb{R}^3$, where $\mu \in \mathbb{R}^2$ gives the centroid of a 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 and with a size smaller than s^d, this can be written more formally as "eventually always ($\mu \in P^d$ and $s < s^d$)", with the obvious interpretation that it will eventually happen that $\mu \in P^d$ and $s < s^d$ and this will remain true for all future times. 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 in the development of the discrete abstraction is the observation that such specifications translate to formulas of temporal logics

over linear predicates in the abstract variables. Informally, LTL formulas are made of temporal operators, Boolean operators, and atomic propositions connected in any "sensible way".[9] Examples of temporal operators include **X** ("next time") , **F** ("eventually", or "in the future"), **G** ("always" or "globally"), and **U** ("until"). The Boolean operators are the usual $\neg$ (negation), $\vee$ (disjunction), $\wedge$ (conjunction), $\Rightarrow$ (implication), and $\Leftrightarrow$ (equivalence). The atomic propositions are properties of interest about a system, such as the linear inequalities in μ and s enumerated above. For example, the last specification in the above paragraph corresponds to the LTL formula $(s < \bar{s})\mathbf{U}(\mu = \bar{\mu} \wedge s = \bar{s}) \wedge (\mathbf{FG}(\mu \in P^d \wedge s < s^d))$.

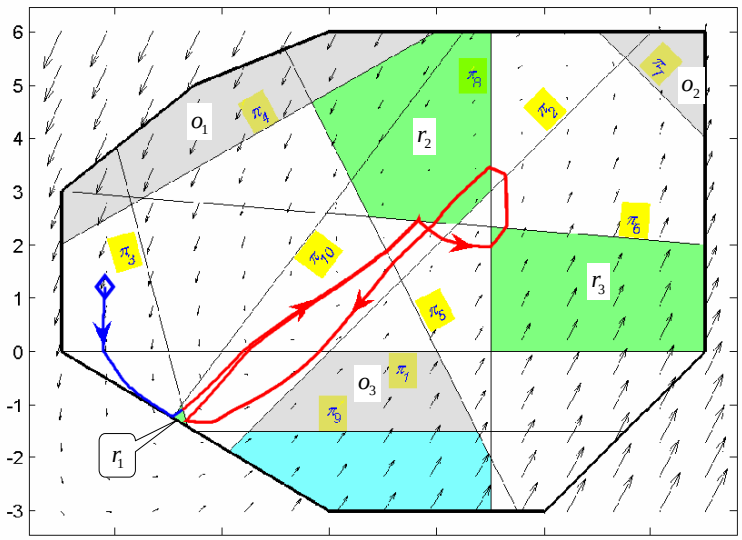


Fig. 1.6. (a) A polygonal 2D state space, the target polygonal regions r_1, r_2, r_3 (shown in green) and the regions to be avoided o_1, o_2, o_3 (shown in grey), which can be expressed as conjunctions of the linear predicates $\pi_1, \pi_2, \ldots, \pi_{10}$. The arrows indicate the drift vector field of system (1.4). There exist control strategies such that all trajectories of the closed loop system (1.4) satisfy formula (1.5) for any initial state in P, except for the grey and the blue regions. A sample trajectory originating in the allowed set of initial conditions is shown.

In,[10] we developed a computational framework for control of arbitrary linear systems (*i.e.*, $\dot{x} = Ax + b + Bu$) from specifications given as Linear Temporal Logic (LTL) formulas in arbitrary linear predicates over the state

x. To illustrate the method, consider the 2D linear system

$$\dot{x} = \begin{bmatrix} 0.2 & -0.3 \\ 0.5 & -0.5 \end{bmatrix} x + \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} u + \begin{bmatrix} 0.5 \\ 0.5 \end{bmatrix}, x \in P,\ u \in U, \tag{1.4}$$

where P is the bounding polytope shown in Figure 1.6, and the control constraint set is $U = [-2, 2] \times [-2, 2]$. Assume the task is to visit a sequence of three regions r_1, r_2, and r_3, in this order, infinitely often, while avoiding regions o_1, o_2, and o_3 for all times (see Figure 1.6). Using the temporal and logic operators defined above, this specification translates to the LTL formula:

$$\mathbf{G}(\mathbf{F}(r_1 \wedge \mathbf{F}(r_2 \wedge \mathbf{F}r_3)) \wedge \neg(o_1 \vee o_2 \vee o_3)) \tag{1.5}$$

The solution is given as the set of initial states from which the formula can be satisfied (see Figure 1.6) and a corresponding set of feedback control strategies.

1.4. Hierarchical abstractions: automatic deployment of swarms from human-like specifications

We are now ready to present a computational framework allowing for automatic deployment of arbitrarily large swarms of fully actuated robots from specifications given as arbitrary temporal and logic statements about the satisfaction of linear predicates by the swarm's mean and variance. These types of specifications seem to be enough for a fairly large class of tasks. In addition, the method allows for automatic containment inside the environment, obstacle avoidance, cohesion, and inter-robot collision avoidance for all times.

To extract the mean and variance of the swarm, we define the following 3-dimensional abstraction map:

$$h(r) = a,\ a = (\mu, \sigma),\ \mu = \frac{1}{N}\sum_{i=1}^{N} r_i,\ \sigma = \sqrt{\frac{1}{N}\sum_{i=1}^{N}(r_i - \mu)^T(r_i - \mu)}. \tag{1.6}$$

In,[5] we show that the robot controllers

$$u_i(r_i, a) = \left[I_2 \ \tfrac{r_i - \mu}{\sigma} \right] w(a),\ i = 1, \ldots, N, \tag{1.7}$$

where $w(a)$ is an arbitrary vector field in the abstract space $\mathbb{R}^3$, provide a correct aggregation with respect to the abstraction map (1.6). Under the assumption that the environment and the obstacles are polyhedral, then we also show that containment inside the environment, obstacle avoidance,

cohesion, and inter-robot collision avoidance can be expressed as LTL formulas over linear predicates in the abstract variables (μ, σ). In addition, arbitrary polyhedral control constraints can be accommodated. The proofs are based on the following two main facts: (i) under the control laws (1.7), the swarm undergoes an affine transformation, which preserves the vertices of its convex hull, and the pairs of robots giving the maximum and minimum pairwise distances, and (ii) quantifier elimination in the logic of the reals with addition and comparison is decidable.

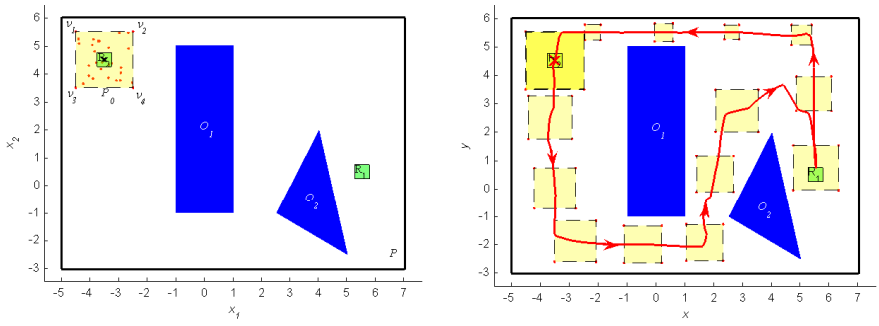
For illustration, consider a swarm consisting of $N = 30$ robots moving in a rectangular environment P with two obstacles O_1 and O_2 as shown in Figure 1.7 (a). The initial configuration of the swarm is described by mean $\mu(0) = [-3.5,\ 4.5]^T$ and variance $\sigma(0) = 0.903$. The convex hull of the swarm is initially the square of center $\mu(0)$ and side 2 shown in the top left corner of Figure 1.7 (a).

Consider the following swarming task given in natural language: *Always stay inside the environment P, avoid the obstacles O_1 and O_2, and maintain maximum and minimum inter-robot distances of 3.5 and 0.01, respectively. 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.* Regions R_1 and R_2 are the two squares shown in Figure 1.7 (a).

This task translates to the following LTL formula over linear predicates in μ and σ:

$$\mathbf{G}(P_{co} \wedge P_d) \wedge \{(\sigma > 0.54)\mathbf{U}[(\mu \in R_1) \wedge \mathbf{FG}((\mu \in R_2) \wedge (\sigma > \sigma(0)))]\}, \tag{1.8}$$

where P_{co} guarantees containment and obstacle avoidance and is a propositional logic formula consisting of 27 occurrences of 19 different linear predicates in μ and σ, P_d gives upper and lower bounds for σ that guarantee cohesion and inter-robot collision avoidance, and $\sigma > 0.54$ corresponds to pairwise distance greater than 0.03. A solution is shown in Figure 1.7 (b) as the trace of the convex hull of the swarm. By close examination, it can be seen that the specified task is accomplished.



(a) (b)

Fig. 1.7. (a) Initial deployment of a swarm consisting of 30 robots in a rectangular environment P with two obstacles O_1 and O_2. The regions R_1 and R_2 are used in the task specification. (b) Trace of the convex hull of the swarm (yellow) and trajectory of centroid μ (red).

1.5. Limitations of the approach and directions for future work

While providing a fully fully automated framework for deployment of arbitrarily large swarms from rich specifications given in human-like language, the approach presented in the previous section has several limitations. First, it is computationally expensive, since, in addition to polyhedral operations, it involves quantifier elimination and a graph search resembling LTL model checking. Second, in order to guarantee that the swarm does not collide with an obstacle, we impose that the whole convex hull of the swarm has empty intersection with the obstacle. One can imagine that there might exist motions of the swarm where an obstacle enters the convex hull without hitting any of the robots. Third, we restrict our attention to a very small set of essential features, which only allow for translation and scaling of the convex hull. For example, if rotation was allowed, motions where the spanning polytope would rotate to avoid obstacles rather than just unnecessarily shrinking, would have been possible. Fourth, our approach to the control of the essential features of the swarm based on discrete abstractions is conservative, as detailed in.[10]

Fifth, in its current form, the approach assumes that the environment is static and known. Any change in the boundaries of the environment or in the obstacles should be followed by a full re-computation. This being said, for a static and known environment, our framework is robust with respect to small errors in knowledge about the environment. This results from the

fact that the discrete abstraction procedure for control of the continuous abstraction from temporal logic specifications over linear predicates (developed in our previous work[10]) is robust to small changes in these predicates. Sixth, throughout this work, we assume that the robots are fully actuated point masses. However, to accommodate robots of non-negligible sizes, one can enlarge the obstacles and shrink the environment boundaries.[11] To accommodate more complex dynamics, one can add another level in the hierarchy. For example, using input-output regulation, controlling a unicycle can be reduced to controlling the velocity of an off-axis reference point.[12] Alternatively, one might try to capture robot under-actuation constraints by properly constructing the continuous abstraction, as suggested in.[6]

Last but not least, the resulting communication architecture is centralized. In fact, for all the continuous abstractions reviewed in this article, the control law for each robot depends on the current state of the robot and the state of the continuous abstraction (see Equation (1.7)). However, this does not seem very restrictive due to two main reasons. First, in a practical application, this communication architecture can be easily implemented if a swarm of ground vehicles is deployed together with an UAV, such as a blimp. The UAV can be equipped with a camera, and each time it can localize the robots, compute the abstract variable a, and then disseminate it back to the robots. Note that the bandwidth of the broadcast variable a is low (since a is low dimensional) and does not depend on the size of the swarm. Second, recent results[13] in consensus algorithms show that global variables, such that the centroid μ, can be estimated using local information, under some mild assumptions of the topology of the communication graph. Starting from this idea, as suggested in,[14] one can build an "abstract state estimator" based only on local information, and feed the estimated values into the framework described above.

1.6. Conclusion

We reviewed some basic ideas and preliminary results allowing for fully automated deployment of arbitrarily large swarms from high level and rich specifications. 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. For planar robots with polyhedral con-

trol 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 are captured.

References

1. F. Zhang, M. Goldgeier, and P. S. Krishnaprasad. Control of small formations using shape coordinates. In *Proceedings IEEE ICRA*, Taipei, Taiwan, (2003).
2. E. Justh and P. Krishnaprasad. Steering laws and continuum models for planar formations. In *Proc. IEEE Conf. Decision and Control*, pp. 3609 – 3614, (2003).
3. R. G. Littlejohn and M. Reinsch, Internal or shape coordinates in the n-body problem, *Phys. Rev. A*. **52**(3), 2035–2051, (1995).
4. C. G. J. Jacobi, *Vorlesungen uber Dynamik*. (Reimer, Berlin, 1866).
5. M. Kloetzer and C. Belta, Temporal logic planning and control of robotic swarms by hierarchical abstractions, *IEEE Transactions on Robotics*. **23**(2), 320–331, (2007).
6. C. Belta and V. Kumar, Abstraction and control for groups of robots, *IEEE Trans. on Robotics*. **20**(5), 865–875, (2004).
7. C. Belta, G. Pereira, and V. Kumar. Control of a team of car-like robots using abstractions. In *42nd IEEE Conference on Decision and Control*, Maui, Hawaii, (2003).
8. N. Michael, C. Belta, and V. Kumar. Controlling three dimensional swarms of robots. In *IEEE International Conference on Robotics and Automation*, Orlando, FL, (2006).
9. E. A. Emerson. Temporal and modal logic. In ed. J. van Leeuwen, *Handbook of Theoretical Computer Science: Formal Models and Semantics*, vol. B, pp. 995–1072. North-Holland Pub. Co./MIT Press, (1990).
10. M. Kloetzer and C. Belta. A fully automated framework for control of linear systems from LTL specifications. In eds. J. Hespanha and A. Tiwari, *Hybrid Systems: Computation and Control: 9th International Workshop*, vol. 3927, *Lecture Notes in Computer Science*, pp. 333 – 347. Springer Berlin / Heidelberg, (2006).
11. J. Latombe, *Robot Motion Planning*. (Kluger Academic Pub., 1991).
12. J. Desai, J. Ostrowski, and V. Kumar, Modeling and control of formations of nonholonomic mobile robots, *IEEE Transactions on Robotics and Automation*. **17**(6), (2001).
13. R. Olfati-Saber and R. M. Murray, Consensus problems in networks of agents with switching topology and time-delays, *IEEE Trans. on Automatic Control*. **49**(9), 1520–1533.
14. P. Yang, R. A. Freeman, , and K. M. Lynch, Multi-agent coordination by decentralized estimation and control, *IEEE Trans. Aut. Control*. (2003). in print.