**Abstraction and Persistence: Macro-Level Guarantees of Collective Bio-Inspired Teams under Human Supervision**

Michael A. Goodrich and Eric G. Mercer

Brigham Young University

**1. Introduction**

There are advantages to having large numbers (100-200) of robots perform coordinated tasks, but such large teams of robots may be difficult for a human to control.  Limitations on communication bandwidth and human attention make it difficult for a single human to gain and maintain situation awareness of each individual robot, and the same limitations also make it difficult for a human to influence individual robots in such a way that the collective behavior of the robots accomplishes some reasonable goals. In response to these limitations, organizational constraints may be imposed that allow a human to issue commands or plays that dictate collective behavior, but naïve hierarchical approaches can suffer from robustness issues in cases where key robots or communication channels are compromised. In response to these issues, researchers have proposed using bio-inspired models to create robust and purposeful collective behavior that is implemented in a completely decentralized organization. Unfortunately, although robust, it may be difficult for a human to influence such collectives in such a way that the collective is responsive to human input.  We present results from a modeling exercise that systematically relates how individual agents can be influenced by humans to ensure macro-level behaviors of the collective.

**1.1 Motivation**

As noted by Lewis [REF], current methods for controlling teams with large numbers or robots (*N* on the order of 20-200 robots) may grow faster than *O(N)* because of the complexity of coordinating robot activities. Adding multiple operators can help with some problems, but if complexity grows as *O(>N)* then the number of required operators will also grow faster than *O(N)*. This is counter to the goal of maximizing force while simultaneously minimizing the number of humans involved

An important way to deal with complexity that arises from inter-agent interactions is to use decentralized control in which robots coordinate behavior in a decentralized fashion. There is an emerging body of strong work on the design of decentralized controllers and on distributed consensus algorithms that provide performance guarantees on the behavior of such systems [REF].

There has also been some interesting work on having a human control such decentralized teams. This decentralized approach has an important benefit: it can help turn an *O(>N)* problem into an *O(N)* or *O(1)* problem; decentralized control takes care of the required inter-agent coordination, leaving the human to manage the team rather than individual agents. A limitation in much of the prior work is that the human takes on a role of centralized supervisor who controls distributed behavior through some sort of broadcast mechanism or who controls a key agent such as a leader with whom all other agents interact. A centralized supervisor can introduce brittleness into the system. If team behavior requires all agents to receive a consistent and centralized signal (from the human or from a leader), then delays or inacuracies in this signal can cause instabilities or errors

**1.2 Proposed Solution**

We propose to explore other roles that a human can adopt to (a) keep management complexity within bounds (*O(N)* or better) and (b) avoid brittleness. The main constraint on these other roles is that the team must produce guaranteed performance. Two key ideas are required to manage complexity and maintain robustness in the system: *abstraction and persistence*.

*Abstraction* means that the behavior of the collective is more than the behavior of the individual members. This principle is well-known in biological systems where a colony or flock exhibits collective behavior that appears purposeful to a human observer in the large scale, even though the behavior of individual members may not appear predictable or sensible. The human must understand the collective at a macro level and focus influence on managing so-called collective states that appear purposeful. Because this collective phenomenon of purposeful behavior is so well-known in bio-inspired systems, we will begin our work by focusing exclusively on such systems.

*Persistence* means that, since we have replaced centralized control with decentralized human influence, the human must persist in influencing the agents in order to have high confidence that the collective state of the system behaves as needed. Simply put, since the human cannot reliably and quickly control every agent at once, the human must then sustain influence on some set of agents in a way that behavior emerges as desired. Such influence must be understood at a level sufficient to ensure robustness and authority in the system.

In the next sections, we identify the technical challenges associated with abstraction and persistence, discuss how we will use a formal-methods approach for identifying performance guarantees, and discuss how we will demonstrate the ecological validity of the guarantees by implementing and evaluating human interaction with robot teams designed to balance persistence and responsiveness.

**2. Model Class**

We adopt a simplified version of Couzin’s model [Couzin’s model]. The discussion we use here was taken from a paper under review and simply summarizes the model. Let *i = 1, 2, …, N* be a set of homogeneous agents with nonholonomic dynamics given by

where is the *i*th agent’s position, is the angular heading of the agent, *s* is the constant agentspeed, and is the angular velocity control input.

For simplicity we define:

and let denote the sensory adjacency matrix where means that agent *j* is visible to agent *i* at time *t* . Each is determined at time *t* according to a Bernoulli random variable with parameter

where is the Euclidean distance between agents *i* and *j* at time *t.* This method of choosing neighbors is similar to the random neighbor model used in [2] which replicates field observations of starlings [1]. The work is relevant for robot systems because occlusions make visibility less likely with growing distance for visual sensors and interference makes sensing less likely with growing distance for radio or wifi-based sensors.

As with Couzin’s model, agents react to neighbors within three different zones: repulsion, orientation, and attraction. The neighbors in these zones are determined by

where , , and are the sets of agent *i*’s neighbors in the regions of repulsion, orientation, and attraction, respectively. The parameters and are the associated radii of repulsion and orientation. Note that this model allows for overlapping regions of repulsion, orientation, and attraction. This overlap eliminates the hard switch between the repulsion, orientation, and attraction forces seen in [Couzin’s Model] that may be sensitive to sensor transients in real robots.

The control input is determined by first computing the repulsion, orientation, and attraction vectors

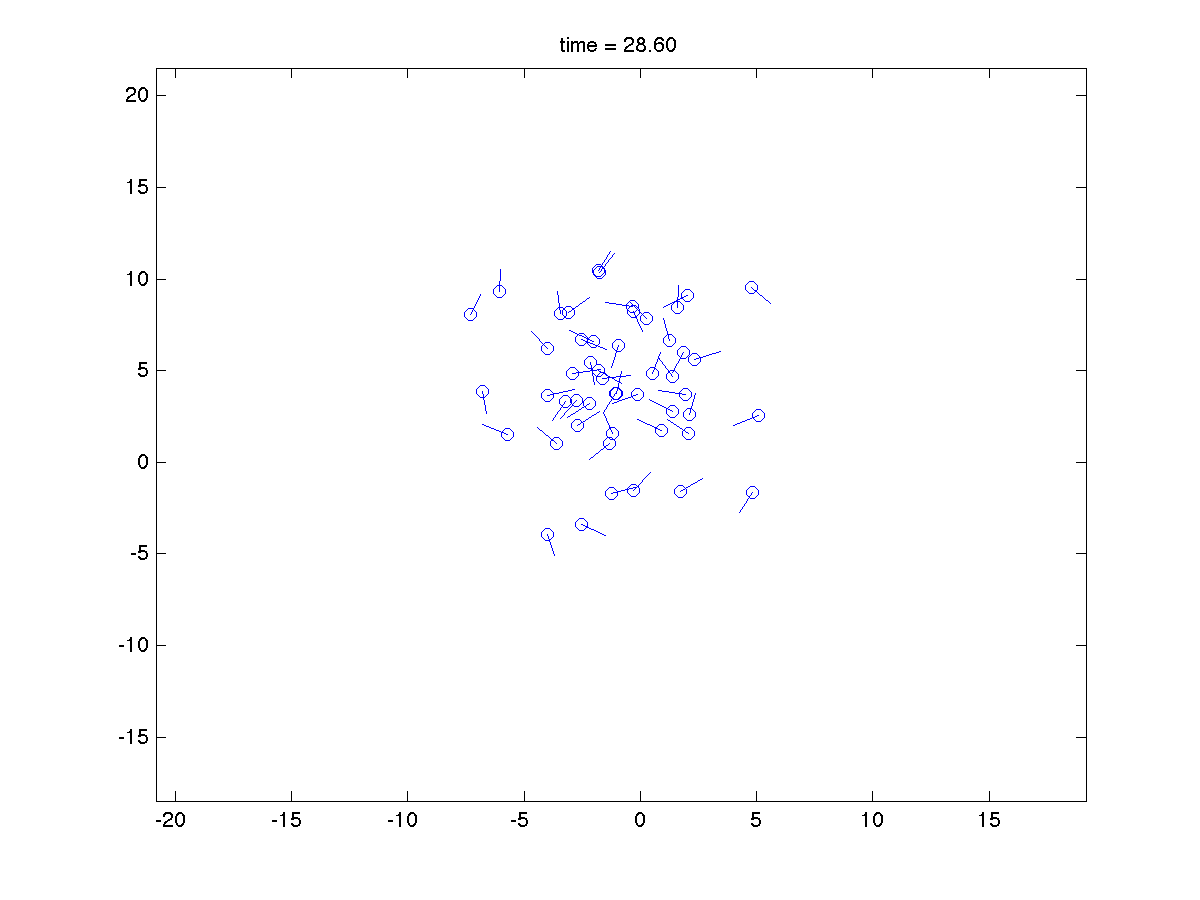
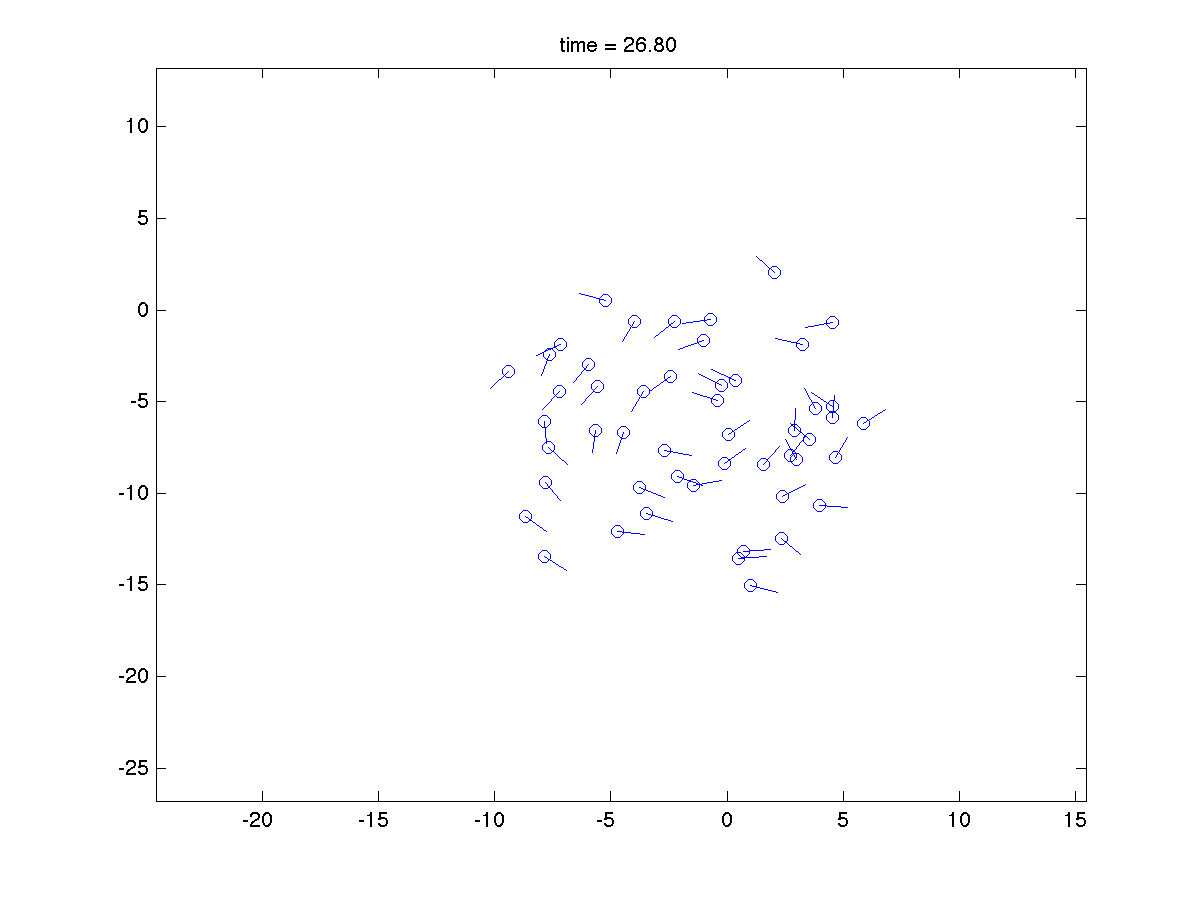
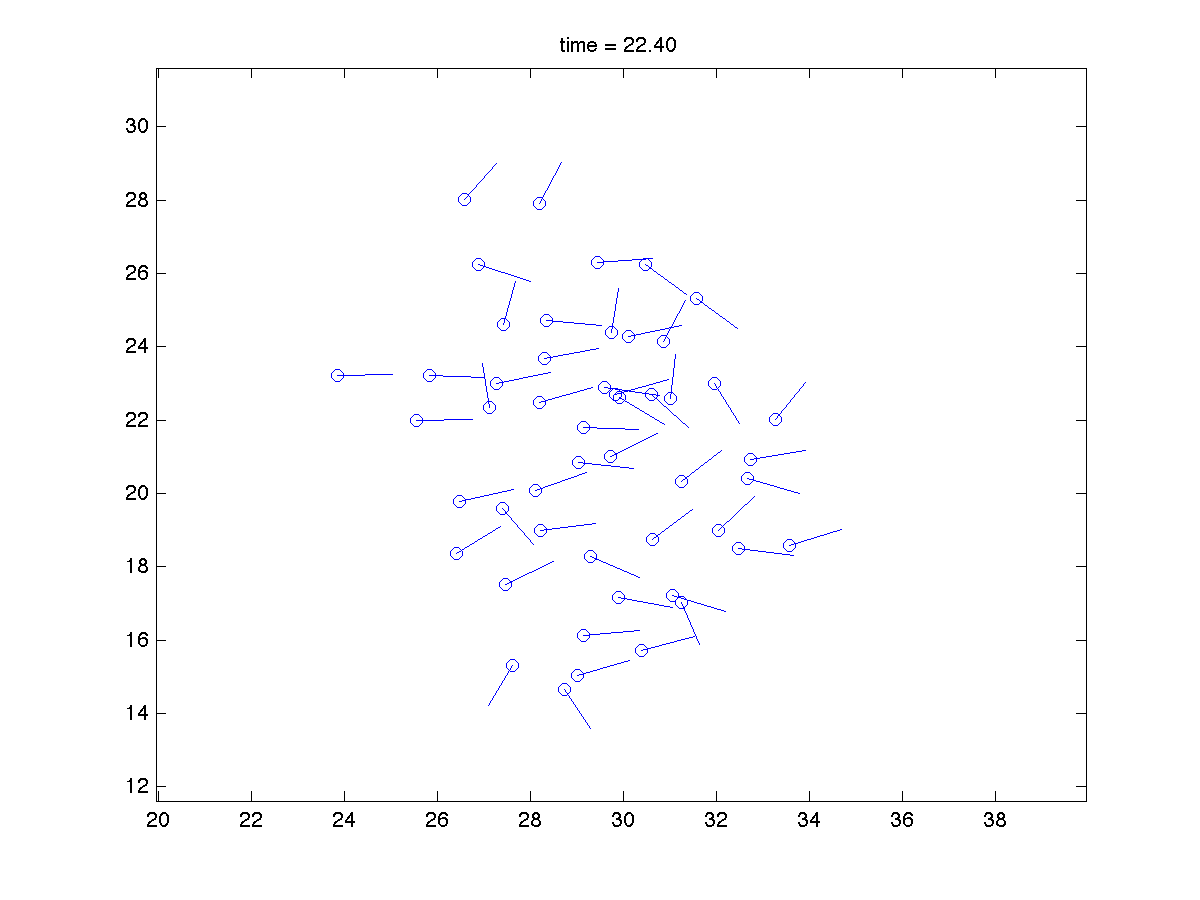
Next, the desired heading vector is computed as

Note that, because of the normalization in (5) and (6), orientation and attraction forces are always equally weighted in the model. This keeps one of the two fundamental forces from overpowering the other. It also allows the exponentially growing repulsion vector to overpower the orientation and attraction forces as agents move closer together, which aids in collision avoidance. Finally, angular velocity, , is computed as

where we limit . Since , is bounded by .

**3. Abstraction**

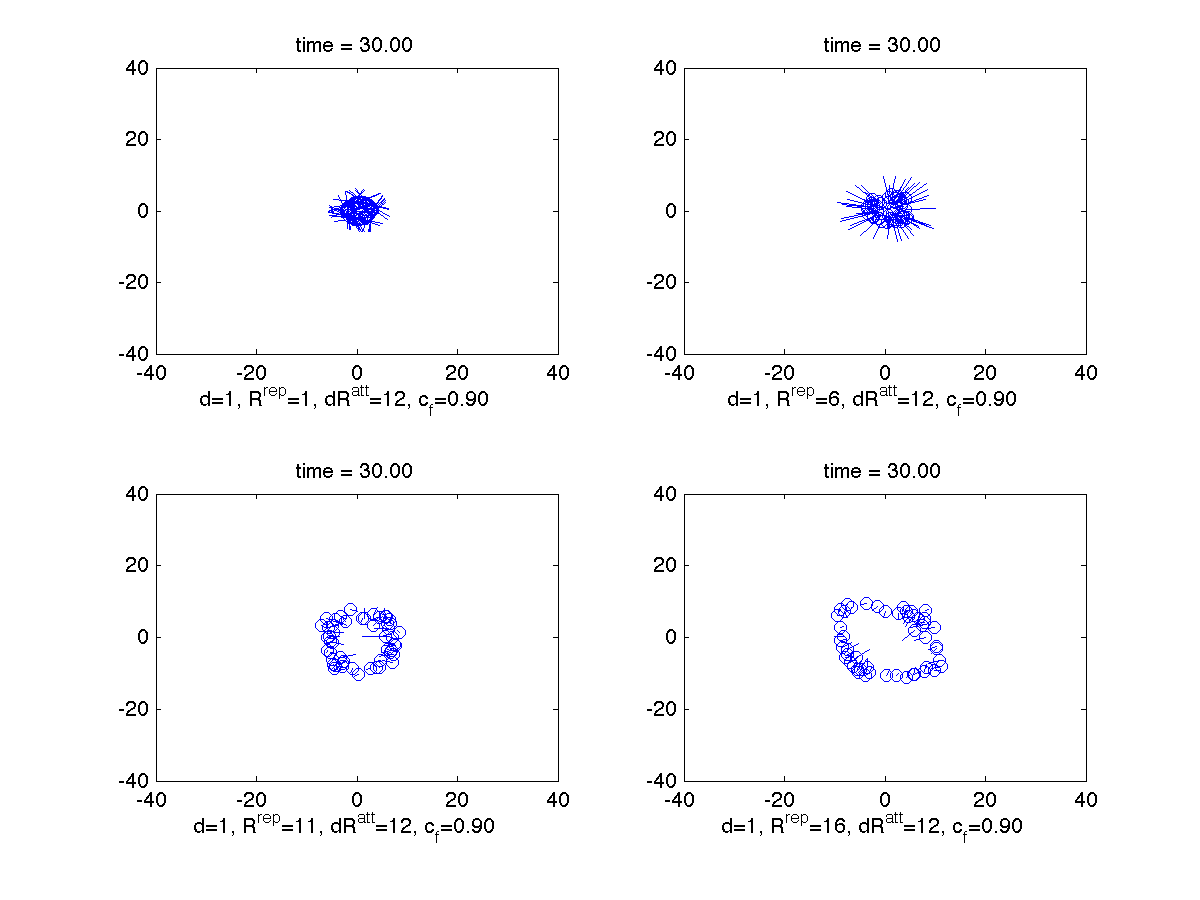
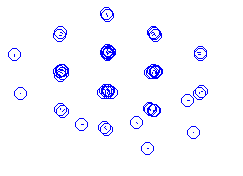
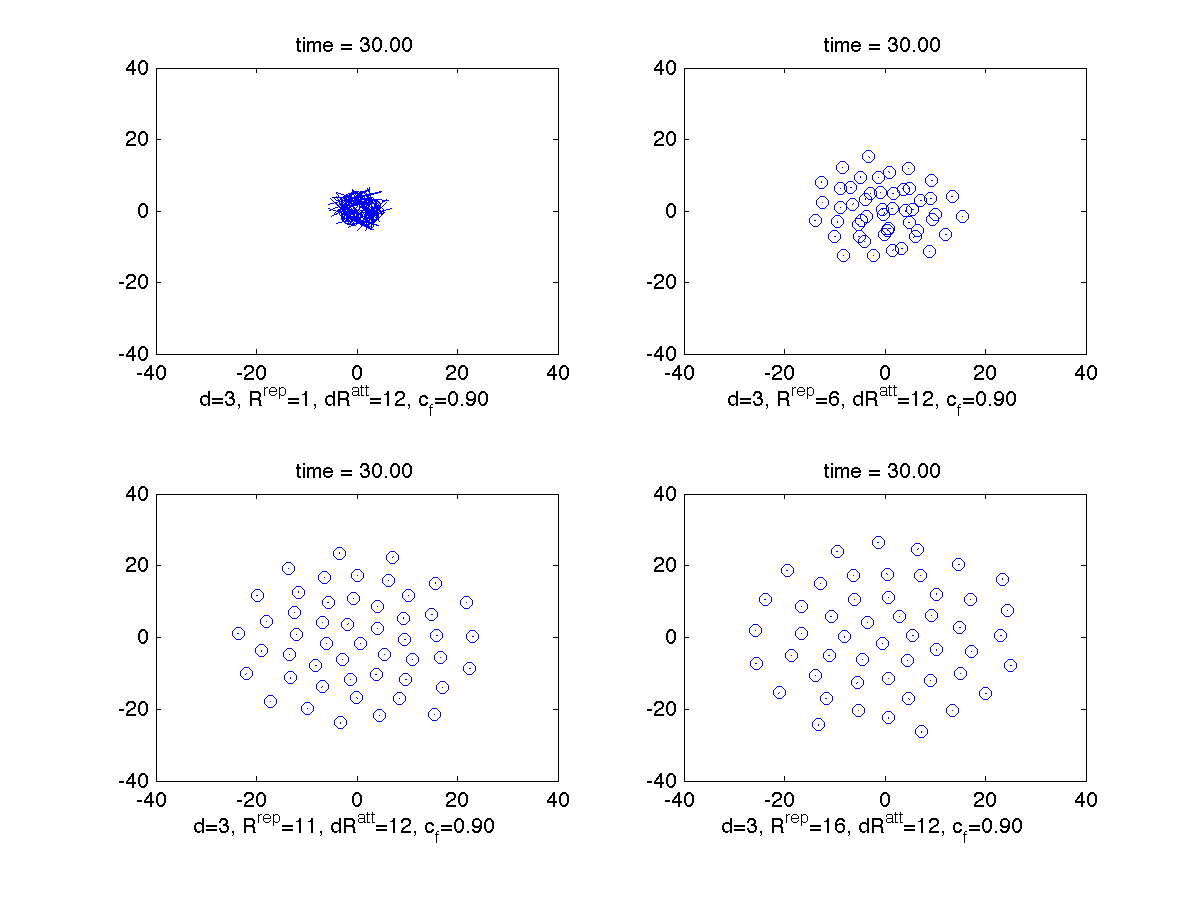
Returning back to Couzin’s model that underlies our simplified model, individual agents are repelled by near agents, are attracted to far agents, and align with intermediate agents. Although the individual states of agents is useful, such complexity does not scale to large numbers of robots; however, when the agents are viewed at a macro level, they display a suite of collective behaviors, illustrated below, which we call *abstract states* (from left to right: parallel group (e.g., flock), torus (e.g., mill) and swarm).



The model in the previous section is capable of producing the flock and the torus behaviors and, importantly, can produce either collective behavior given the same set of model parameters ( and , the radii of repulsion and orientation). The key to using this model is that either behavior can be produced, and which one is produced depends on the initial states of the agents. By manipulating initial states, a human can influence the emergent behavior of the collective. Even more importantly, a human can cause the collective to switch from one abstract behavior state to another by influencing a subset of the agents [cite paper in review and RSS paper].

From the perspective of the abstract states on the collective, the behavior of individual agents evolves as a stochastic difference equation, making it difficult to characterize or predict the behavior of any individual agent’s behavior. Fortunately, both the abstract states of tori and flocks are attractors of the dynamic system that is created by the decentralized control laws from the previous section. As such, the abstract states represent attractors in the dynamic system. Characterizing the stochastic system with abstract states leads to the definition of a critical challenge in human interaction with bio-inspired robot teams: how does one allow the agents to maintain their decentralized implementation while allowing the human to influence individual agents and thereby influence which attractor or abstract state is exhibited by the collective.

Before proceeding, it is useful to provide some evidence that the abstract state is not peculiar to our model described above. Couzin’s results are well-known, but distinct states are also exhibited by other systems. For example, a physically inspired system created by Spears [Spears] exhibits distinct phases also illustrated below (bottom, from left to right: isotropic, anisotropic, structured [TechnicalReport]). From these two models, we hypothesize that other stochastic systems have similarly defined attractors that can be used as abstract states to model and control collective behavior.



**3.1 Influencing the Attractor**

We propose that *efficient human interaction with a decentralized robot team must operate at the “macro-level”* encoded by abstract states that are attractors in the underlying dynamic system. Consistent with both bio-inspired and communication-limited systems, we assume that the decentralized implementation means that (a) the number of agents that influence agent *i*’s behavior is small, and (b) the number of agents influenced by operator or environment inputs is small. Furthermore, we assume that the inter-agent topology varies over time (such as when an agent travels in and out of another agent’s neighborhood).

Given this definition of abstract state, the human’s task is to influence the collective to travel from one abstract state to another using the tools available to it. Prior work has explored leaders, predators, and stakeholders to influence how a collective moves, but much more work needs to be done to model and understand this influence in relation to collective movement. We will talk more about methods of influence in the design section below, but at this level of abstraction it is appropriate to note that there are inputs available to a human through which he or she can guide the agent from one abstract state to another.

A principal challenge in the macro-level abstraction is that the transitions between states behave non-deterministically over an undetermined period of time. Human influence is eventually implemented as commands to individual agents and these agents then influence other agents through the network topology. Thus, the human is “injecting” influence into the system, and that influence percolates over time to other agents. As such transitions between abstract states take a non-deterministic amount of time depending on the percolation rate through the collective. To better understand how to influence a collective through individual agents, we build and analyze a probabilistic model that further abstracts details of individual agent states.

**3.2 A Model of Influence**

In the simplified Couzin model presented earlier, we have observed two distinct and controllable abstract states: flock and torus. Each state emerges depending on initial conditions of agents, and parameters can be selected so that each state is equally probable given a broad set of those initial conditions. We build a Markov Decision Process (MDP) over these two abstract states, and we build the model in such a way that individual agent states are implicitly, rather than explicitly, modeled. In other words, we no longer directly track, as in the Couzin based model, the underlying stochastic system governing individual agent behavior, and we instead view agents as either participating in the collective state or note.

The state of the MDP is given by the tuple where and . The variable *S* indicates the abstract state of the collective and is either flock (F) or torus (*T*). The variable *M* is the number of agents adopting the behavior of a new abstract state. In this model, *M* is limited to 100 agents. Intuitively, the human influences some number *M* of agents, and over time, that number grows as that influences percolates until the collective emerges into the new desired abstract state. We will build the MDP incrementally starting with the abstract states *S* which are equally likely in the model:

(F,0)

(T,0)

It is possible to cause the agents to change from one state to another by controlling the number of agents in the collective who are informed about the location of some desired object. This is done as follows.

A *stakeholder* is an agent that is influenced by the group and also influenced by its interaction with a human operator. In [3], Couzin showed how a limited number of stakeholders influence the behavior of a flock. In other prior work, we have shown that not only can stakeholders allow a human to control the direction of a flock, but that stakeholders can also be used to control a torus and dynamically switch between attractors.

Recall that *N* represents the number of agents in the collective. We assume that the human operator can broadcast a desired location to *M < N* stakeholder agents. By broadcasting a reference input to a limited number of agents a human may influence the movement of the swarm and even cause the group to switch attractors.

A stakeholder has dynamics and control inputs just like every other agent. The difference is that the control input is modified to allow a human operator to influence the stakeholder’s behavior. This influence is added into the stakeholder attraction dynamics through a weighted sum as follows:

where is the reference, is the priority of the reference input (relative to the importance of the influence of other agents), and is determined as described in Section 2. Orientation and repulsion are given as described in Section 2 and combine with the stakeholder attractor as follows:

This control input enables human influence on the system without eliminating inter-agent dynamics. The input is limited to the attraction dynamics in order to avoid tampering with orientation dynamics and repulsion dynamics, which are necessary for collision avoidance.

This model of influence allows the operator to manage either the number of agents influenced or the priority of reference input versus inter-agent influence. As a function of and *M*, prior work has identified the probability, using Matlab simulation, of a swarm switching from torus to flock (left) and from flock to torus (right).

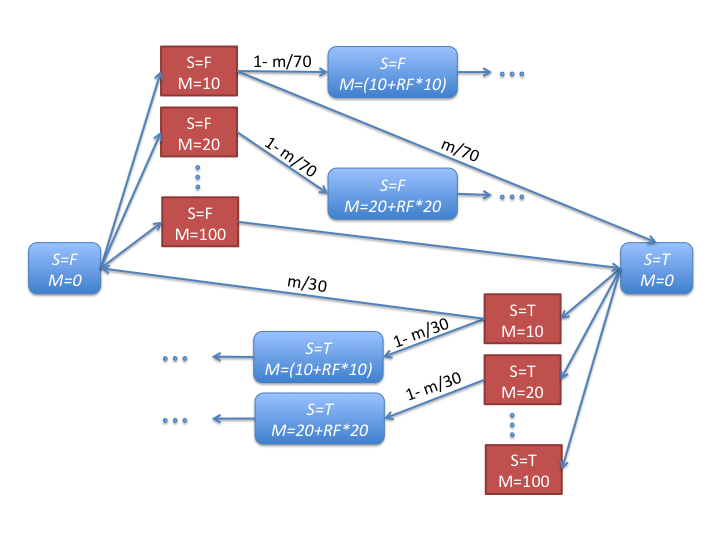
 

In the interest of simplicity, the MDP model only considers the control variable *M*, meaning that the human can choose how many agents are influenced, and that influence percolates over time through the collective until the desire abstract state emerges. We use a linear model of the rate of influence with slope given by *RF*. To be specific, if *m* agents are influenced at time *t*, then .

We are now ready to present our full MDP model illustrated below. Selecting a value for gives the probability of transitioning between torus and flock attractors given the number of agents under influence, . Similarly, selecting also gives the probability of transitioning between flock and torus attractors, . We model these distributions with a linear function defined from the above surface plots at The square nodes in the MDP represent actions. From either of the two stable abstract states, there is a choice of action that determines how many agents the user directly influences. Those actions are equally probable in this model. Once an action is selected, the model nodeterministically transitions either to the stable abstract state or to a state where more agents adopt the new behavior. The probability of transition is determined by the value m. As in the surface plots above, at some level of m, the transition to a stable abstract state is probability 1. These bounds are reflected in the denominator of the probability calculation as 70 and 30 for transitions from flock to torus or torus to flock respectively.

This shifts the probabilistic model above from a static model to a less complex MDP that can be analyzed to understand influence in the agent system.

This model is very simple and more complicated models will be necessary. It will be necessary to consider not only the attractor, but also the spatio-temporal aspects of the attractor. For example, the direction of rotation of the torus and the direction of flight of the flock must also be specified. Future work will consider more sophisticated models, but for the purposes of this work the model above is chosen and a finite set of values of *M* is considered as defined previously.



**3. Persistence**

Creating the MDP is nice because it allows an abstract description for how a human can influence which abstract state is manifest by the collective. Importantly, the MDP can be used to show formally that the interaction between human and collective is *sound* and *complete*. The first step toward performing this formal analysis is defining the property of persistence.

*Persistence* is a quantitative measure of authority in this MDP that represents the level of control exerted over a period of time by a human operator necessary to affect emergent behavior in a collective. Persistence encapsulates both span of control (i.e., the number of agents directly influenced by the human) and a minimum time bound over which the human exerts input to the agents before the desired collective behavior emerges.

Important to persistence is a notion of spatial and temporal locality. Spatial locality is the span of control (i.e., the number of agents directly influenced by the human, which is precisely the initial value *M* in the MDP described above). Temporal locality is the minimum time bound over which the human exerts input to the automation at a given span of control before the desired behavior emerges. Both aspects of the measure, temporal and spatial, are necessary to understand and characterize actions in the MDP model of the human-robot team.

We propose to formalize persistence using *probabilistic computation tree logic* (PCTL) over states and actions of the MDP abstraction. PCTL is an extension of computation tree logic (CTL). CTL describes universal and existential properties of branching in a non-deterministic transition system [CTL\_ref]. The PCTL extension adds to the branching time logic probabilities and soft deadlines [bea98]. For example, PCTL enables properties such as, “there is a 98% probability that the collective will form a torus structure within 2 seconds of the human input *u*op” to be naturally stated over the alphabet of the MDP.

Informally, persistence is a bound, *[lt,ut]*, on the amount of time a particular human input, or action, at a given span-of-control *M*, must be given to the robots to guarantee, that an observed collective state, , reflects a desired emergent collective behavior. The property is computed in our MDP model using the PRISM stochastic model checker -- a tool for formal modeling and analysis of systems that exhibit random or probabilistic behaviors [prism\_ref]. To measure time, the MDP is augmented with rewards. From the point where the action is resolved, each state is rewarded with the constant 1, representing a single time step in a discrete space, until the system adopts the desired collective behavior in the abstract state. It is now possible to measure persistence, or the expected time to transition to the abstract state, by asking PRISM to find computation paths in the MDP that maximize or minimize the reward. The general pattern for the PRISM property is expressed as

filter(min, Rmin=? [ **F** "flock" ], ("torus"&"m10"))

which is understood intuitively as, “from torus abstract state where the chosen action is m=10, what is the minimum reward when the system eventually reaches the flock abstract state.” PRISM supports a similar operator, Rmax, to compute the maximum reward.

In our simple MDP model the expected time to transition to a Flock from a Torus when directly influencing 10 agents is 2.43 units. The value, of course, changes as we adjust the probability distributions in the model. In general, the maximum time to transition from a Flock to a Torus in the worst case (m=10) is 4.4 time units. The minimum time is 1 which occurs for M >= 70. The maximum time to transition from Flock to Torus is 2.43 and the minimum time is 1 for M>=30. The running time to compute these bounds by PRISM is negligible.

Working at this level of abstraction is useful for identifying guaranteed bounds on the behavior of the abstract system. To be of practical use for real human-robot teams, we must map the behavior of the team at the abstract level to the implementation-level details encoded in model described in Section 2. Let *f=* and let *g=* … This leads to an important technical challenge, namely using formal methods to show that the low-level human inputs correctly implement the abstract action, that is,



using *fi*  and *gi*.

To help understand why this technical challenge is important, note that persistence measures human control over the automation. An input from the human with very low persistence at a high probability indicates centralized authority where the human is directly controlling the collective. In contrast, a very high persistence from an action probably indicates decentralized authority with limited control over the collective behavior. With such a notion of persistence, it is possible to define and characterize control actions, *αn*, that reflect a desired level of authority in the automation.

A second technical challenge is the relationship between the abstract state and what is observable from the system. We will formally model *β* as a predicate that, within probabilistic bounds, indicates the abstract state of the collective. We will then use the requirements for this predicate to identify what must be observed by the human, meaning that we will relate *β* to the observations available from the robots (*ya*, using the notation from section 2).

Given such a characterization of control through persistence, it is possibly to classify actions, *n*, in a three dimensional space defined by control-authority (measured as temporal span), span-of-control (measured as the number of agents influenced), and emergent behavior (measured as the probability of a particular attractor or macro-level behavior). With such a classification, we can identify a third technical challenge, namely creating a catalog or library actions that uses formal methods to identify what various values of can do to the system. This catalog is the basis for algorithms that will implement user interfaces that allow a human to manage persistence time and level of control authority. A plan over decentralized control gives more autonomy to the collective while a plan over centralized control minimizes autonomy.

**4. Future Work**

Future work on abstraction includes:

1. Find ways to define abstract states for Couzin’s and other bio-inspired systems; include how the human can influence a transition from one state to another.
2. Use empirical methods to identify state transition probabilities under various forms of human influence.
3. Generalize these models to more sophisticated bio-inspired teams, like colonies or foraging groups that depend not just on inter-agent dynamics and human input, but also have agents that autonomously respond to the environment through *hi*.
4. Generalize to other decentralized structures, that is, beyond bio-inspired teams to other command-and-control organizations.

Future work on persistence includes:

1. Formally modeling persistence and generating probabilistic bounds required for a given abstract input to produce a desired abstract behavior.
2. Using formal methods to show that a low-level implementation in the dynamic system correctly implements the abstract action.
3. Modeling the relationship between abstract state and low-level agent states.
4. Categorizing the effects of various abstract actions on collective behavior.

Future work on validation includes the following:

In the previous sections, we have argued that it is beneficial to shift from an implementation-level, which uses individual agent states and dynamics plus precise human inputs, to a macro-level, which uses abstract states and human-directed state transitions. This benefit allows us to define formal properties of the system and then to relate those formal properties to requirements that must be satisfied in order for the system to exhibit desirable behavior.

We propose to “close the loop” between those formal guarantees and the design of real systems. A measure of success in these real systems is how well they reflect *scalability*:

1. Scalability in number of robots: from 20-200.
2. Scalability in temporal scales: from seconds to minutes to hours. (This is equivalent to scalability in the presence of agent degradation.)
3. Scalability in the number of human teammates: from a single supervisor to a squad to a company.

We propose to design user interfaces that allow us to evaluate scalability using a series of simulations and user studies. We begin with a series of hypotheses about the design of collective behaviors.

*Hypotheses of good design*

1. There is a tension in knowing when a human should be able to override the collective behavior of a team of agents. Sometimes, the human should be able to override as in when the human is trying to direct a change from one state to another. Other times, the human may be a disturbance to the system because the human may be making an error or the agents may have information that the human does not know. Persistence, required for producing the guarantees above, is a design tool for resolving this tension. *The tension between human-directed and collective-directed behavior should be represented in the user interface as a minimum level of spatial and temporal locality.*
2. To make the system change quickly, the human must be able to control how much time is required before the system is guaranteed to respond. This control suggests that the *human should be able to exert more control authority but to be informed that doing so trades off responsiveness for potential error.* Requesting more bandwidth or broadcasting power increases control authority supplanting distributed influence with more centralized control.
3. Design becomes the art of planning behaviors that balance temporal persistence with control authority while respecting performance guarantees. *The “set point” in the balance between temporal persistence and control authority can be associated with the human’s role*:

* Controller
* Information Source
* Stakeholder
* Leader
* Parameter-manager
* Supervisor
* Manager.

*Technical Challenges*

We propose two methods for evaluating these hypotheses.

1. Design user interfaces that (a) *prevent the human* from engaging in an undesired sequence or (b) *provide decision support to the operator* when they are about to engage in an activity that can yield undesirable consequences. Of particular interest are problems that arise with scalability:
   1. Very small time scales that do not allow sufficient time for human influence to propagate.
   2. Small team sizes (approximately 10-20 robots) that are not sufficiently large to guarantee collective behavior.
   3. Very large time scales such as those that might emerge when robots are placed in a naval environment and over an extended period of time, and which may experience performance degradation.
   4. Multiple humans who exert loci of control that conflict each other.
2. Conduct a series of experiments with humans and simulated robots that implement user interface concepts and evaluate their success or failure in terms of performing plausible reconnaissance and monitoring tasks. Measure usability, workload, situation awareness, and task performance.

**5. Conclusions**

* Bio-inspired systems can be modeled at an abstract level as an MDP with state as (cohesion matrix, management matrix, centroid) and actions as abstract commands to transition between states.
* Formal methods can be used to relate the states and actions of the MDP to behaviors of the implementation-level state-space system, guaranteeing that the abstraction behaves correctly.
* Persistence (temporal locality and span of control) can be used to describe formal guarantees of behavior for the MDP.
* We can use these formal guarantees as guidelines for designing user interfaces that (a) enforce sequencing that avoids incorrect behavior or (b) inform the operator when a choice is made that moves the system outside of guaranteed operational parameters.
* We can generalize these results to other (additive) decentralized systems.

