Actuation Constraints and Artificial Physics Control[⋆]

Chris Ellis and R. Paul Wiegand

University of Central Florida
{chris@cs,wiegand@ist}.ucf.edu

Abstract. Swarm systems for multiagent control rely on natural models of behavior. Such models both predict simulated natural behavior and provide control instructions to the underlying agents. These two roles can differ when, for example, controlling nonholonomic robots incapable of executing some control suggestions from the system. We consider a simple physicomimetics system and examine the effects of actuation constraint on that system in terms of its ability to stabilize in regular formations, as well as the impact of such constraints on learning control parameters. We find that in the cases we considered, physicomimetics is surprisingly robust to certain types of actuation constraint.

1 Introduction

Swarm intelligence [1] is a popular and successful group of methods for controlling coordinated multiagent teams. Of such approaches, those based on variations of artificial physics models, physicomimetics [2], have particular appeal. The resulting behaviors are quite intuitive; it is easily generalized to allow for modular, heterogeneous and scalable team behaviors [3]; and traditional analytical tools from physics can be used to help diagnose and predict team behaviors[2]. Physicomimetics is particularly well-suited for tasks that require stable geometric formations such as lattices or rings, and under the proper circumstances one can show that teams will settle into "low-energy" positions provided by such structures.

It is clear that control methods based on artificial physics models are performing two essentially different tasks: 1) *predicting motion* of particles within a particle-based physics model (particle model), and 2) *producing control input* to move agents in some real or simulated world (environment). When agents are treated as simple point-mass particles with no additional constraint on their motion, these roles do not conflict. However, for realistic control systems operating in the environment (e.g., nonholonomic robotic platforms), a conflict between these roles occurs when the agents being manipulated cannot move as requested. Analyses regarding the stabilization of regular formations, for instance, rely on a temporal element — the dynamics of the particle model itself. When there is a disconnect between model prediction and control, such analyses are questionable since the dynamic will almost certainly differ, potentially quite radically. Additionally, though parameters for physicomimetics control systems are often hand-coded, complex problems require some kind of learning, and it isn't clear how the disconnect between prediction and control affects the learning gradient.

[⋆] This work was funded as part of the ARL/HRED project on *Team Performance in Human-Agent Collaboration* (W911NF-07-R-0001).

Still, physicomimetics has been successfully applied to a wide range of control problems including mobile robot formation [2], multi-robot chemical plume tracing [4], and heterogeneous, multiagent in-port ship protection [3]. Moreover, in many cases control systems have been demonstrated both in simulation and on physical devices, where actuation constraint varies widely.

We show artificial physics based control systems *are* affected by actuation constraints on the agents; however, physicomimetics is surprisingly robust to such prediction / control disparities. We construct a common nonholonomic control system that constrains agent motion in a number of ways and parameterizes the maximum allowable turning speed and examine the effect of this parameter on lattice formation. Considering a more complex covert tracking problem that requires learning, we discover that added constraints affect properties of the system and influence the learning gradient. Only in extreme cases are the behaviors qualitatively different.

The next section will discuss the control system we are using in detail. Section three will discuss the effects of constraints on simple hexagonal lattice formation, while section four will detail our efforts to learn physicomimetics based control solutions for a covert tracking problem. We finish up with a short discussion of related work, as well as our conclusions and future plans with this research.

2 Representing Agent Behaviors with Artificial Physics

2.1 Physicomimetics

Physicomimetics provides a framework for the control of multiple agents [2]. Each agent has its own physicomimetics model, which is updated at each time step from that agent's knowledge of its environment, including the observed positions and velocities of all observed agents. Agents are treated as point-mass (m) particles. Each particle has a position, $\mathbf{x}$, and velocity, $\mathbf{v}$. We use a discrete time simulation, with time step Δt. At each time step, the particle is repositioned based on the velocity and the size of the step, $\Delta\mathbf{x} = \mathbf{v}\Delta t$. The change in velocity of the particles is determined by the artificial forces operating on the particles, $\Delta\mathbf{v} = \mathbf{F}\Delta t/m$, where $\mathbf{F}$ is the aggregate force on the particle as a result of interactions with other particles and the environment. Each particle also has a coefficient of friction, $c_f \in [0, 1]$. Velocity in the next step becomes $(\mathbf{v} + \Delta\mathbf{v})c_f$, stabilizing the system [2]. When this new velocity is computed in the physicomimetics model, the agent tries to the best of its ability to match it in its own environment. The values of these masses, frictions, and force laws are what govern the motion in this system, and are either hand selected or learned in some fashion.

There are two constraints: the magnitude of the force cannot exceed F_{max} and the magnitude of the velocity cannot exceed V_{max}. These restrict acceleration and velocity of particles in the model. Also, since there is an emphasis on *local* interactions, there are further restrictions on the range of effect particles have on each other.

Advantageously, a variety of force laws can be employed to different effect. The parameters of the above model, coupled with the force law parameters, provide engineers with mechanisms to adjust the behaviors of agents. Finally, since physicomimetics is based on physics, practical analyses are possible using traditional physics techniques such as force balance equations, conservation of energy and potential energy [2].

A slight variation of the well-known Newtonian force law will be used in this paper:

$$F_{ij} = \begin{cases} -G\frac{(m_i m_j)^a}{r_{ij}^d} & \text{if } r_{ij} \in [0, R) \\ G\frac{(m_i m_j)^a}{r_{ij}^d} & \text{if } r_{ij} \in [R, E] \\ 0 & \text{otherwise} \end{cases} \tag{1}$$

The magnitude of the force is determined by choice of gravitational constant, G. The force law repels particles closer than R and attracts particles past that distance but within the range of effect, E. The gradient of the force can be controlled using d, and a can raise or lower the importance of mass on the force. In total, there are two parameters associated with each particle (m and c_f) and five parameters associated with their interactions (G, E, R, a, and d). Distance variable r_{ij} is an observed phenomenon.

2.2 Constraining Agent Motion

Our goal was to simulate a parameterized differentially steered device. We used a number of parameterized constraints on agent movement. Physical constraints such as maximum velocity V_{max}, maximum acceleration a_{max}, and maximum turning speed θ_{max} are placed upon the agents, and the physicomimetics control system is allowed to suggest velocities without regard for these limits. At each time step, the agent will update its orientation, velocity, and position according to the following algorithm:

1. The orientation and velocity of the agent is rotated by $\Delta\theta$ towards the suggested velocity, where $\Delta\theta = min\left(\theta_{max}, \theta_\delta\right)$. θ_δ is the difference in orientation between the current agent orientation and the orientation of the suggested velocity.
2. The magnitude of the agent's current velocity is set to $|\mathbf{v}| = |\mathbf{v}_{prev}| \cdot cos(\Delta\theta)$, where $|\mathbf{v}_{prev}|$ is the magnitude of the velocity at the previous time step.
3. The suggested velocity is projected along the updated orientation vector, and agent velocity is updated to as close to the projected suggested velocity as a_{max} permits.
4. The magnitude of the agent's velocity is constrained by the maximum velocity, $|\mathbf{v}| = min\left(|\mathbf{v}|, |V_{max}|\right)$.
5. Agent position is updated according to the new computed velocity, $\Delta\mathbf{x} = \mathbf{v}\Delta t$.

Dampening the speed of the agent proportionately to $\Delta\theta$ in step 2 has a stabilizing effect on the agents, which we use here in place of friction. It is for this reason that an agent with a turning speed constraint of π is different from a traditional agent. Of the thresholds discussed, the turning speed constraint θ_{max} was chosen as the independant variable in order to observe to what degree increasing constraints impact the performance of both the agent and learning algorithms operating on that agent.

3 Constraining Motion in Simple Lattice Formations

One of the simplest and most natural formations obtainable by agents controlled via physicomimetics is an hexagonal lattice. Straightforward swarm design methods can produce solutions capable of settling into a regular isometric grid quickly and efficiently. With appropriate parameters, one does not even need friction for the system to find equilibrium in such stable configurations because the low potential energy wells of the system correspond with these structures.

We refer to our control group as the “traditional model”, described in section 2.1, where there is no inconsistency between the environment and the particle model. In this case, a swarm designer can affect lattice width via the R (attraction-repulsion boundary) parameter. The effect range, E, is typically set at $1.5R$ to be less than the $\sqrt{3}R$ factor that allows second-tier points in the lattice to be visible. Our parameters follow those of [2]: $R = 50, E = 75, G = 1200, a = 1, d = 2$, save that we use 100 particles (Spears used 200), and we do not use friction. This system will settle into a hexagonal lattice.

We investigate two properties of the system: *settling time* and *lattice quality*. Settling time is the time it takes the system to find a quiescent state. Lattice quality is a measure of how faithfully the distributed agents replicate an isometric formation.

3.1 Settling Time

Under the right conditions, a particle model will lose energy as it converges on a stable formation. When agents cannot move as dictated by their surrogate particles, unstable dynamics might be introduced in the physicomimetics system since the criteria of the proofs for stability in [2] are not all met. To test this, we considered control systems with varying constraints on turning speed ($\theta_{max} \in \{0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9, 3.14\}$ radians), as well as the traditional (non-constrained) model. We ran each model 30 times and analyzed the dynamics in terms of the average scalar acceleration magnitude of all agents each step.

Using our nonholonomic control system above, we find that the system not only settles into a formation in all cases, but the damping factor for sharper turning seems to help the system settle faster. The left panel in Figure 1 below illustrates averages over the thirty trials of the settling behavior in four of the above groups. We examined all of the above groups, and the basic curve characteristic is similar in all cases, and standard deviations (not shown) indicate very little variability in this settling behavior.

To investigate this behavior more carefully, we consider the *settling time* of the system: the number of time steps taken for the average magnitude of acceleration to drop below an empirically selected threshold, 0.01 distance units per step squared. The right side of Figure 1 shows these results for all experimental groups. Pair-wise t-tests using Bonferoni adjustment indicates no statistical differences between any of the constrained groups, but all constrained groups have a significantly lower settling time than the traditional model (95% confidence).

While constraining the motion of the agents undoubtedly affects the rate at which they settle into a stable formation, our nonholonomic constraints do not prevent this ability in general. Indeed, our differentially steered agents settle *faster* than the traditional approach because of the damping influence on sharp angle motions.

3.2 Lattice Quality

Arriving at a stable configuration quickly does not necessarily imply that the same configuration is reached. Again, swarm design on the traditional system to produce hexagonal lattices is predicated on a basic understanding of traditional physics. This understanding is of questionable value when particles cannot move freely.

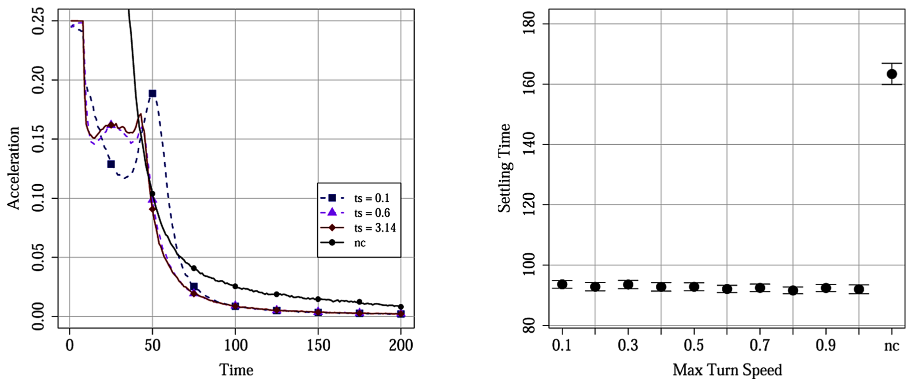


Fig. 1. *Left graph*: four systems settling into a formation. Each curve is the average scalar acceleration magnitude of the system across thirty different simulations. *Right graph*: the settling time for each of the groups. Points and wings represent means and 95% confidence intervals.

To investigate this, we again run the above experimental groups ($\theta_{max} \in \{0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9, 3.14\}$ and traditional case) for thirty trials apiece and measure the average lattice quality for fifty steps after the system has settled (acceleration magnitude has dropped below 0.01). Peaks in the acceleration graph represent the moment when agents are slipping into their minimal energy configurations. Peak position is likely a function of the number of agents.

We found the lattice quality measure used in [2] too sensitive to the value of R for our purposes. Instead, we compute the Delaunay triangularization [5] of the particles, then measure the coefficient of variation in edge lengths in this graph. To reduce boundary effects, we use only edges with points in the inner 85% of total area covered by the agents. Figure 2 below illustrates the mean and confidence intervals for the quality results for all groups.

Using the multi-way comparison previously described, we find that the lattice quality of the traditional group is significantly better than the constrained cases. Also, the $\theta_{max} = 0.1$ case differs from all other cases. The 0.2 case differs from the 0.6, 0.7, 0.8, 0.9, and 3.14 cases. All other comparisons are statistically indistinguishable.

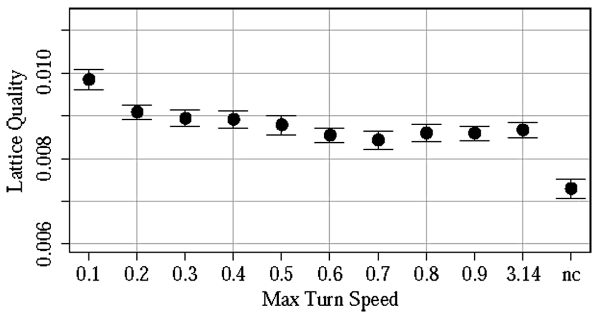


Fig. 2. Lattice quality for each group. Points and wings: means and 95% confidence intervals.

While our nonholonomic agents form hexagonal lattices that are of marginally lower quality than the traditional models, there is no doubt that the lattices *are* formed. Visually, they look very similar. A random dispersion of 100 points results in a lattice quality of more than four times that of the worst of our groups; all groups formed lattices that were significantly better from a statistical point of view. Moreover, the lattice quality is fairly robust to the *degree* of constraint (in terms of θ_{max}): using constraints marginally decreases lattice quality, but the degree of constraint is not particularly important.

4 Constraining Motion in the Covert Tracking Problem

We are interested in a generalized form of a multi-target tracking problem, where our coordinated team of agents learns to distribute the task of tracking various targets with differing capabilities and with different objectives in mind. Such problem domains require fairly sophisticated swarm designs and (typically) some kind of learning.

However, it isn't clear how portable swarm design methods are when there are profound disconnects between how agents *can* move and how a swarm system *directs* them to move. Moreover, it isn't clear how such constraints impact learning performance. To begin to answer this question, we focus on a simple form of our more general problem and investigate how our constrained nonholonomic controller impacts learning, as well as the final solution quality.

4.1 Covert Tracking

For this experiment we use a single tracker and a single target. The goal is for the tracker to follow the target as closely as possible and not be seen by the target. The target has a field of vision that consists of two concentric circles. The target will detect the tracker if it is anywhere in the inner circle, 10 distance units. The outer radius (30 distance units) is a $\frac{3}{2}\pi$ radians arc, such that the target has a "blind spot" directly behind its facing direction. The target moves at a maximum velocity of 1 unit per step and is constrained in a way similar to the tracker with a $\theta_{max} = 0.05$. It randomly wanders as follows: with a probability of 0.05 each time step, it selects a new desired facing direction and changes its velocity to match that preference (allowing for actuation constraints). The new direction is chosen uniformly at random within a relative ± 3 radians. The behavior is fairly smooth, with occasional surprising turns.

Initial positions of the target and tracker are chosen uniformly at random in a rectangular area of 80×60 units in size. The tracker has a 2π radians field of view up to 80 distance units and has a maximum velocity of 2 units per step. It's behavior is controlled via physicomimetics as described above using three types of particles. The first represents the target, the second the tracker, and the third a *virtual particle* — a particle representing an unembodied concept to be used by the control system, described below. Each step, the tracker constructs a particle model, placing a tracker particle in its own position, a target particle in the position of the target (if it is seen), and a virtual particle in a position computed using a relative range and bearing from the position of the target. The offset bearing is relative to the bearing of the target, and the tracker estimates the target bearing based on its change in position.

The virtual particle is necessary if we hope to have the trackers exploit the blind spot of the target. While our representation does not prescribe how this particle is used, the most obvious solution is to place the particle in the blind spot. The learning system is responsible for determining this.

The control system requires 23 parameters. The mass and coefficient of each of the three particles (6), the force law parameters for each interaction ($5 \cdot 3$), and a range / bearing offset for computing the position of the virtual particle. These are all represented as real values. The table below describes the permitted ranges for these values.

Table 1. Ranges for control system parameters, $\epsilon = 0.00001$

Type of parameter	Range	Type of parameter	Range
mass, m	$[0, 200]$	distance power, d	$[-10, 10]$
friction, c_f	$[0, 1000]$	mass power, a	$[-10, 10]$
effect range, E	$[0 + \epsilon, 480]$	virtual particle range, ρ_v	$[0, 60]$
AR boundary, R	$[0 + \epsilon, E]$	virtual particle bearing, θ_v	$[-\pi, \pi]$
gravity, G	$[-1000, 1000]$		

4.2 Learning a Physics Model for Control

The 23 parameters just discussed were encoded into a real-valued genome, and a (5+35)-ES was used for optimization. Gene values were in the range $[0.0, 1.0]$, and were scaled to the ranges of each individual parameter of the force laws the physicomimetics control system. Fitness for an individual was aggregated over 20 trials, for 600 timesteps per trial. Adaptive mutation was employed as in [6] with $\sigma_{init} = 0.25$ and $\sigma \in [0.005, 0.25]$. Evolution took place over 50 generations, and the most fit individual found was then evaluated with a number of metrics for our empirical analysis (described below).

Fitness at each time step for each individual was evaluated as follows.

$$F(a \in trackers, b \in targets) = R_\alpha \cdot \left(sees(a,b) \frac{D(a) - r_{a,b}}{D(a)} \right)^{R_\beta} - P_\alpha \cdot sees(b,a)$$

$$sees(a, b \in agents) = \begin{cases} 1 \text{ if agent } a \text{ `sees' agent } b \\ 0 \text{ otherwise} \end{cases}$$

Here $D(a)$ represents the vision range of agent a. There are three parameters R_α, R_β, and P_α, used here to tune the ES towards different desired behaviors. R_α is the reward scaling factor, R_β is an exponent that changes the signifigance of the distance of the trackers to the targets they see, and P_α is the penalty scaling factor. For the purposes of our experiment, $R_\alpha = 1$, $R_\beta = 1$, and $P_\alpha = 3$. This creates an environment where the fitness reward increases linearly with the proximity of the tracker to its target. The reward given per time step can be up to 1.0. A flat penalty of 3 is applied at each time step if the tracker is seen by the target. Increasing the reward for smaller distances causes the tracker to get as close as possible, while the penalty for being seen ensures that the tracker will avoid the vision area of the target.

4.3 Predicting vs. Controlling Agent Behavior

The first hypothesis that needs to be confirmed is that limitations on agent mobility create a disconnect between the executed behaviors of the controlled agents and the described behaviors of their analogous particles. If that is so then presumably the learning system will have to work with, or compensate for, those differences. Our suggested position difference measure, Δ_{pos} confirms the first part of this reasoning.

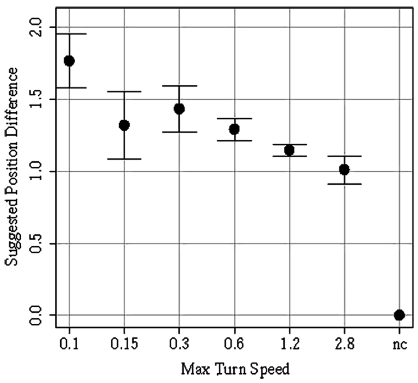
At each time step the physicomimetics system suggests a velocity that the agent, to the best of its ability, tries to execute. This yields a suggested position, $\hat{\boldsymbol{x}}$, and the actual updated position the agent is capable of achieving, $\boldsymbol{x}$. The suggested position difference for a given time step is then the Euclidean distance between the suggested and actual position for that step. The magnitude of this value represents the degree to which the agent is incapable of matching the directions given by the control system. Our measure aggregates this value over all time steps according to the following equation: $\Delta_{pos} = \frac{\sum_{t=1}^{T}(\|\hat{\boldsymbol{x}}(\boldsymbol{t})\| - \|\boldsymbol{x}(\boldsymbol{t})\|)}{T}$.

The Δ_{pos} for the best of run for each of thirty trials of the EA were collected. The results were aggregated over each value of θ_{max}, and a Bonferroni-adjusted t-test was applied to test for statistical differences between the results (see the left graph of Figure 3). As is expected, as the turn speed constraint is relaxed, the overall trend of the output of the physicomimetics model moves to more closely match the actual change in location of the agent. However, even as the turn speed approaches the point where an agent may turn any direction in a single time step, there is still a statistically significant difference compared to the traditional agent. As expected, the constraints create a disconnect between particle model prediction and the resulting executed behavior.

4.4 Effects on Learning Performance

Despite the fact that there is the disconnect discussed above, the learned solutions are surprisingly good. Though the learned solution in the traditional case is significantly better than all the others (in fact, all groups are significantly different from one another), all but the most extreme cases learn the same basic behavior: Set the virtual particle inside the blind spot of the target, then track the target from that position.

Consider the right plot in Figure 3, below. First, all values are reported in terms of their average fitness per step. Second, the means and confidence intervals of the best of run values for each experiment group are plotted. Finally, an *incursion zone* is plotted as a shaded rectangle to highlight the efficacy of the learned solutions. This zone represent distances smaller than the range of vision of the target, scaled to the limit of the tracker's range of vision $\left(\frac{D(a)-r_{a,b}}{D(a)}\right)$. Fitness values within that range must result from behaviors that stay within the outer range of the target's range of vision and receive no penalty from being seen (i.e., in the target's blind spot). The wider confidence wings on the 0.15 and 0.3 groups occur because some trials fail to discover this, while others succeed. The learning gradient for placing the virtual particle becomes particularly steep when the constraint is high.



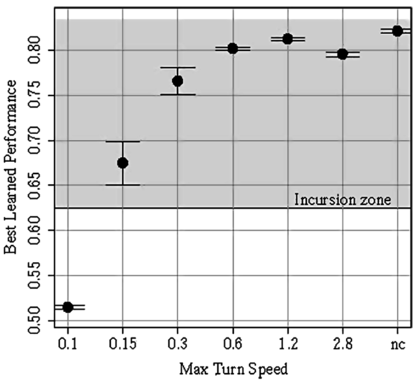


Fig. 3. *Left graph*: Suggested position difference for several experimental groups. *Right graph*: Best of run fitnesses for all groups. The grey area represents values where trackers remain (on average) in the blind spot of the target. Points and wings: means and 95% confidence intervals.

5 Related Work

There is an increasing wealth of literature that uses swarm intelligence for coordinated control of groups of agents, such as physicomimetics [2,3,4], methods based on social potential fields [7], or methods based on flocking and schooling behaviors in animals [8]. These approaches rarely focus on the effects on actuation constraints on agents under control, though the agents themselves are often nonholonomic.

Additionally, there are traditional control theory methods for formation control in agents, typically based on a leader / follower paradigm [9]. In an interesting middle-ground approach, [10] presents a method for designing cooperative formation control systems for groups of mobile robots (both holonomic and nonholonomic) based on potential functions. [11] examines three aspects of a behavior-based approach to coordinated multiagent control that includes control of robots under differential steering. These methods incorporate explicit notions of actuation constraint and (often) include formal justifications for which certain patterns will stabilize; however, they lack the intuition and simplicity of bottom-up, nature-based approaches.

6 Conclusions and Future Work

Though it is clear that actuation constraints can have potentially profound impacts on how a multiagent system must be controlled to produce useful coordinated behavior, little effort has been made to date to determine how swarm-based approaches are affected by such constraints. We believe a closer look at when and how actuation constraints affect swarm-based behaviors is therefore justified, and this paper presents a preliminary empirical look into these effects in two simple domains: hexagonal lattice formation and covert tracking. We examined a differential-like control system that allowed us to vary the degree of constraint on agent movement.

We found that while constraints do impact system performance, only in the most extreme cases were the physicomimetics control methods unable to accomplish the basic tasks at hand. The systems did not destabilize, nor did they alter the basic character of the behavior implemented in the unconstrained cases. Simple, high-quality hexagonal lattices were formed even when maximum turn speed was quite low, using the same parameters as successful unconstrained physicomimetics agents used to solve the same problem. Constraints affected learning performance only minimally, save when they were particularly severe. Our results provide some hope that for certain, simple problems physicomimetics is fairly robust to these kinds of mobility limitations.

We believe that even when the constraints are severe, learning difficulties can be mitigated using transfer learning. In the most extreme case of the covert tracking problem, the system never learns to place the virtual particle in a useful position and the agent learns simply to stay outside the vision range. Our approach is to first learn the control parameters in the traditional way, then use this to bias the search for behaviors when the tracker cannot move so freely. Early results are encouraging.

References

1. Bonabeau, E., Dorigo, M., Theraulaz, G.: Swarm Intelligence. Oxford University Press, Oxford (1999)
2. Spears, W., Spears, D., Hamann, J., Heil, R.: Distributed, physics-based control of swarms of vehicles. Autonomous Robots 17, 137–162 (2004)
3. Wiegand, R., Potter, M., Sofge, D., Spears, W.: A generalized graph-based method for engineering swarm solutions to multiagent problems. In: Parallel Problem Solving From Nature, pp. 741–750 (2006)
4. Zarzhitsky, D., Spears, D., Thayer, D., Spears, W.: A fluid dynamics approach to multi-robot chemical plume tracing. In: Auton. Agents and Multi-Agent Systems, pp. 1476–1477 (2004)
5. Guibas, L., Stolfi, J.: Primitives for the manipulation of general subdivisions and the computation of voronoi. ACM Trans. Graph. 4(2), 74–123 (1985)
6. Schwefel, H.P.: Numerical Optimization of Computer Models. Wiley, Chichester (1981)
7. Reif, J.H., Wang, H.: Social Potential Fields: A Distributed Behavioral Control for Autonomous Robots. Robotics and Autonomous Systems 27(3), 171–194 (1999)
8. Reynolds, C.: Flocks, herds and schools. Computer Graphics 21, 25–34 (1987)
9. Ogren, P., Egerstedt, M., Hu, X.: A control lyapunov function approach to multiagent coordination. IEEE Transactions on Robotics and Automation 18, 847–851 (2002)
10. Nguyen, D., Do, K.: Formation control of mobile robots. International Journal of Computers, Communications and Control I(3), 41–59 (2006)
11. Lawton, L., Beard, R., Young, B.: A decentralized approach to formation maneuvers. IEEE Transactions on Robotics and Automation 19(6), 933–941 (2003)