International Journal of Geographical Information Science
Vol. 20, No. 9, October 2006, 945–964

Taylor & Francis
Taylor & Francis Group

Research Article

Swarming methods for geospatial reasoning

H. VAN DYKE PARUNAK*, SVEN A. BRUECKNER, ROBERT MATTHEWS and JOHN SAUTER
NewVectors LLC, 3520 Green Court, Suite 250, Ann Arbor, MI 48105-1579, USA

(*Received 30 September 2005; in final form 4 April 2006*)

Geospatial data are often used to predict or recommend movements of robots, people, or animals ('walkers'). Analysis of such systems can be combinatorially explosive. Each decision that a walker makes generates a new set of possible future decisions, and the tree of possible futures grows exponentially. Complete enumeration of alternatives is out of the question. One approach that we have found promising is to instantiate a large population of simple computer agents that explore possible paths through the landscape. The aggregate behaviour of this swarm of agents estimates the likely behaviour of the real-world system. This paper will discuss techniques that we have found useful in swarming geospatial reasoning, illustrate their behaviour in specific cases, compare them with existing techniques for path planning, and discuss the application of such systems.

Keywords: Swarming; Software agents; Path planning; Movement prediction

1. Introduction

The physical structure of the environment constrains the movements of robots, people, or animals ('walkers'). Ecologists may be interested in how changes to the landscape will affect migratory patterns. Emergency preparedness workers may need to evaluate the flow of evacuees through a road network. Security personnel may want to track the likely location of suspected terrorists. Roboticists may want autonomous vehicles to find their way through a landscape populated by threats and targets. In each case, one or more walkers move in response to stimuli, constrained by the topology and topography of their environment.

The set of choices available to a walker in such a system is, in the worst case, exponential in the length of the walk. A walker on a square lattice can turn in eight possible directions at each step, so that a walk of length k can follow one of 8^k possible trajectories. Complete enumeration and analysis of the alternatives for a single walker is clearly impractical.

Yet in practice, the structure of the environment often constrains reasonable choices. For example, there is very little chance that a walker at the bottom of a steep valley will wander up the valley wall. In fact, the value of geospatial analysis is greatest in just those domains where such constraints exist. If the landscape does not constrain movement, migration will be unimpeded, evacuees and robots can go wherever they want, and security personnel have no choice but to monitor the entire space. The task of geospatial reasoning is often to discover and identify the very constraints that restrict the combinatorial explosion of the search.

*Corresponding author. Email: van.parunak@newvectors.net

International Journal of Geographical Information Science
ISSN 1365-8816 print/ISSN 1362-3087 online © 2006 Taylor & Francis
http://www.tandf.co.uk/journals
DOI: 10.1080/13658810600830525

Rather than reason sequentially about the exponential tree of possible movements open to a single walker, we instantiate a large population of simple computer agents, each of which explores one possible path through the landscape. These agents diversify their behaviour through a stochastic element in their decisions. Given a walk of length k, we replace a branching search of a single walker's trajectory (of complexity $O(c^k)$, where c is the number of choices available at each step) with a search over multiple parallel non-branching trajectories (of complexity $O(kn)$, where n is the size of the swarm). The computational burden can be adjusted to balance the accuracy of the results against the resources available. In addition, in some applications each member of the swarm moves independently of the others. In these cases, the swarming approach lends itself to implementation on parallel hardware, while the sequential exploration of a branching trajectory is difficult to parallelize due to the dependence of later steps on those that have gone before.

This swarm of agents performs a Monte Carlo sampling of the possible paths through the space. Its aggregate behaviour estimates the likely behaviour of the real-world system. If the space does in fact highly constrain movement, the trajectories of the members of the swarm will converge, and in the process identify the constrained path. A space that does not constrain movement identifies itself through the dispersion of the swarm.

Such a system can be used in two ways. First, it can *predict* the behaviour of walkers that move autonomously (e.g. for studies of animal migration or the movement of adversarial troops). Second, it can *plan* movements that walkers execute under external control (e.g. path planning for robots or search teams). Because the same mechanisms support both prediction and planning, the same system can do both at the same time. We regularly use these mechanisms in military applications, to *plan* the movements of one population of walkers (representing friendly troops) in a way that anticipates and responds to the *predicted* movements of another population (representing adversaries).

This paper introduces these swarming methods for geospatial reasoning. Section 2 outlines several techniques that we have found useful in such systems. Section 3 gives concrete examples of three such systems, two for planning and one for prediction, and shows how planning and prediction can be combined. Section 4 discusses some practical issues in deploying such systems, including their relation to conventional methods for geospatial analysis in geographic information systems (GISs), speed of convergence, support for the computational environment on which our methods rely in distributed robotics, and ways to exploit the information generated by these methods in predictive systems. Section 5 concludes.

2. Modelling techniques

Swarming geospatial reasoning is an application of agent-based modelling. Instead of reasoning logically about how entities might behave in a geospatial environment, we create models of the entities, situate them in a model of the environment, simulate their behaviour, and observe what they do. The collective behaviour of the model answers the same kind of question that one might put to more conventional methods (e.g. predicting an entity's movement, or planning a desirable course of action), so we describe it as 'reasoning'.

To set the context for our methods, we characterize agent-based models in contrast with other computational models, distinguish swarming models from other

agent-based models, and discuss the criteria by which such models may be evaluated.

2.1 *Agent-based vs. equation-based modelling*

At some risk of oversimplification, computational models can be divided into two broad classes. In agent-based modelling (ABM), the model consists of a set of agents that encapsulate the behaviours of the various individuals that make up the system, and execution involves emulating these behaviours. In equation-based modelling (EBM), the model is a set of equations, and execution involves evaluating them.

Equation-based models, such as systems of differential equations, were the only practical form of mathematical model in the days before computers. They are quite mature (Sterman 2000) and can be executed extremely rapidly using numerical integration. Agent-based models have become popular only with the advent of inexpensive computers. They offer significant benefits over equation-based models in respect to the underlying structure of a model, the naturalness of its representation of a system, and the verisimilitude of a straightforward representation (Parunak *et al*. 1998), and have rapidly grown in popularity in many fields, including geographical information systems (Gimblett 2002).

2.2 *Distinctives of swarming agent-based modelling*

The most straightforward application of agent-based modelling to a geographical scenario would be to assign one agent to each entity, model the agent's behaviour on the entity's as closely as possible, then execute the set of agents and observe their behaviour.

In a swarming approach, the agents that explore the landscape differ from real agents (and from a naïve agent representation) in several ways. Swarming methods are inspired by mechanisms exhibited by social animals, notably insects (Parunak 1997), and these mechanisms often rely on characteristics that differ from those associated with real-world people or robots. These differences include the number of walkers, their internal logic, stochasticity, and stigmergic information exchange. The publications by Parunak, Brueckner, and Sauter cited in this section provide further details, lessons from experience, and methodological recommendations for those wishing to adopt our techniques.

2.2.1 Number of walkers. In a conventional multi-agent model, agents are in one-to-one correspondence with physical entities in the real world. Swarming systems achieve self-organization through the repeated interactions of many agents, and if the population of real agents is too small, a one-to-one correspondence will not yield the required dynamics. Thus, each physical agent may correspond to many computational agents. In some cases, we may even instantiate computational agents that do not correspond to any specific physical agent.

Using a many-to-one representation has another benefit in addition to enabling the dynamics of self-organization. Representing a single entity by a population of agents is analogous to representing a single particle by a wave function. We are shifting our focus from the unique behaviour of one individual to a collection of behaviours that sample the space of possible actions. It is sometimes helpful to interpret their movements as concurrent Monte Carlo. The resulting distribution of swarming agents can then serve as an estimate of the probability function of real walker distribution over the landscape. Our polyagent technology (Parunak and

Brueckner 2006a) captures this technique, using a single persistent avatar agent to manage the model's correspondence with an entity in the real world, and a swarm of ghost agents to explore the entity's possible alternative behaviours.

2.2.2 Walkers' internal logic. The conventional approach to ABM seeks to model the internal logic of each real-world entity. When those entities are humans, each agent becomes an independent artificial intelligence, most commonly modelled in terms of its beliefs, desires, and intentions, the so-called 'BDI' model (Rao and Georgeff 1991, Haddadi and Sundermeyer 1996, Müller 1996). The development of the complex symbolic knowledge bases needed to support BDI agents is an instance of the knowledge acquisition problem, which has long plagued the development of realistic AI applications.

The constraints imposed on the agents by the environment (including one another) often outweigh the effect of different decision algorithms, so that agents with different decision algorithms yield the same system-level behaviour. We call this phenomenon 'universality' (Parunak *et al*. 2004b), borrowing the term from its use in statistical physics to describe the emergence of identical critical exponents in widely different physical substances at their critical points. As in physics, so in multi-agent systems we do not fully understand what makes universality happen (or even how to predict when it will or will not apply). But knowing that it can happen encourages us to begin modelling with very simple rules rather than with the complete decision logic of a real-world agent. We focus on simple environmental clues ('prefer the path with the lowest gradient') and tropisms ('head in a general southerly direction'), and then enhance the agent sophistication only as long as it improves the performance of the system. The 'brain' of an agent is not a knowledge base, but a simpler (usually quantitative) structure such as a neural network or a polynomial. These structures can be tuned using synthetic evolution (Sauter *et al*. 2002, Brueckner and Parunak 2003, Parunak 2005), which is a much more efficient process than the knowledge acquisition required for BDI agents.

2.2.3 Stochasticity. Rational agents are typically deterministic, driven by computations that are modelled on theorem proving or optimization theory. For example, an agent will typically have an objective function that it seeks to maximize. One consequence of this approach is that agents with an identical state will make identical decisions, and the system will quickly fall into a stagnant state. To overcome such symmetries, one must invest in detailed knowledge engineering to capture the distinctions that always exist among real-world agents.

A simpler way to break symmetries is to have agents choose stochastically among alternative behaviours. Typically, we use a Boltzmann–Gibbs function. For example, at a point in its evolution, an agent might have n possible choices, each with perceived value v_i. Instead of making the choice with the highest value, the agent chooses among them, assigning each the probability

$$p_i = \frac{e^{v_i/T}}{\sum_j e^{v_j/T}} \tag{1}$$

In this equation, T is a temperature parameter that determines the degree of stochasticity in the decision. When T is small, the agent chooses the highest-valued option almost deterministically. When T is large, the choice becomes almost equal among the alternatives. This approach, inspired by simulated annealing

(Kirkpatrick *et al.* 1983), breaks symmetries among the agents without expensive knowledge engineering and avoids local minima. We have developed local measures that agents can use to estimate the degree of convergence of the system and thus adjust T dynamically as the model runs (Brueckner and Parunak 2003).

2.2.4 Stigmergic information exchange. Conventional agents interact primarily by sending messages to one another. These messages are typically symbolic, with a grammar based on standards such as KQML/KIF (Finin 1997) or FIPA (FIPA 2000). While the messages necessarily pass through a communication infrastructure such as the Internet, the agents are not aware of this infrastructure and behave as though they had direct telepathic capabilities.

Studies of insect behaviour have revealed the importance of indirect communication, mediated by a shared environment. The French entomologist Grassé (1959) coined the term 'stigmergy' from the Greek words *stigma* 'sign' and *ergon* 'action', to capture the notion that an agent's actions leave signs in the environment, signs that it and other agents sense and that determine their subsequent actions. A common form of stigmergy, and one that we imitate heavily, is the use by ants of chemical markers (pheromones) that they deposit and sense (Brueckner 2000). Stigmergic interaction has several benefits over message-based interaction, including simplicity, scalability, robustness, and the ability to take advantage of environmental noise to support the need for stochasticity in decision-making (Parunak 2003, Parunak and Brueckner 2004). In our systems, agents deposit and sense digital pheromones (named scalar variables) at their current locations in a computational environment.

The use of a distinct environment as the primary medium of interaction avoids computational paradoxes that can arise from direct agent-to-agent communication (Michel 2004). It is one thing for an agent to intend to change the world, but quite another for the change to succeed. Classical agent models often assume unrealistically that an agent's actions achieve their intended purpose (Ferber and Müller 1996). The environment provides a computational locus where the actions of different agents can be integrated and arbitrated, just as the laws of physics do in the real world.

2.3 *Evaluating swarming geospatial methods*

Other, non-swarming methods are commonly used to plan and predict behaviours under geospatial constraints. In deciding which method to use for a given application, three criteria must be taken into account: applicability, efficiency, and effectiveness. Applicability concerns the computational structure of the application. Efficiency concerns the computational effort needed to produce a solution. Effectiveness concerns the quality of the solution. The three are largely orthogonal to one another. Abstract discussions comparing different solutions usually focus on effectiveness, but real applications are often so constrained by issues of applicability and efficiency that they can sacrifice some degree of effectiveness.

2.3.1 Effectiveness. The first question one usually asks about a computational method concerns the quality of the answer it delivers. In prediction problems, effectiveness is estimated by the error between the prediction and what actually ends up happening. In planning problems, it is measured by the value achieved by executing the plan, compared with the maximum value achievable. Often, a relative

rather than an absolute measure is useful, comparing how well one mechanism does with other commonly used mechanisms. Relative measures are particularly important in planning problems, where the maximum value achievable in a scenario may be difficult to determine.

Swarming methods are seldom justified on the basis of effectiveness alone. Like neural networks and genetic algorithms, they fall in the general category of 'weak methods,' methods that do not rely on detailed knowledge of the problem domain. A method that embodies deep domain knowledge can sometimes be more effective than one that does not, but at a price. The effectiveness of a strong method typically falls off drastically when the assumptions of the domain are not met (a phenomenon known as 'brittleness'). In addition, strong methods often use logical manipulations (such as varieties of theorem proving) whose computational complexity is prohibitive and may not fit naturally into the computational structure of the application. These latter two concerns are addressed by efficiency and applicability, respectively.

2.3.2 Efficiency. In real-world applications, an approximate answer soon is often worth more than an exact answer later. This constraint impacts the choice between strong and weak methods in at least two ways, which are reflected in potential measures of efficiency.

First, strong methods that rely on logical manipulations of symbols often face combinatorial bottlenecks. These bottlenecks are most severe in NP-hard problems (Garey and Johnson 1979), in which processing time increases faster than polynomially in the size of the problem. Swarming approaches are one way to find an approximate solution to such problems in polynomial time. This insight suggests the usefulness of measuring how long an algorithm takes to solve a problem, and how that time varies as a function of size of the problem.

Second, even when a strong method is computationally tractable, it often provides no answer until the complete answer is ready. Swarming methods tend to be anytime. That is, they quickly yield an approximate solution, which then becomes more refined if more time is available for solution. Ideally, the improvement of quality over time should be non-convex, so that the greatest gains are realized early in the algorithm's execution. This insight suggests the usefulness of measuring the shape and convergence speed of a method.

2.3.3 Applicability. Even more fundamental than the speed and quality of solution is the issue of whether an algorithm is appropriate to the computational structure of an application. Swarming methods are motivated primarily by constraints of applicability. They have been found useful primarily in domains that may be characterized mnemonically as discrete, deprived, distributed, decentralized, and dynamic (Parunak and Brueckner 2004).

- Discrete problems involve the interactions of numerous entities. The nonlinearities of these interactions can lead to combinatorial explosion in methods that reason about those interactions explicitly. In swarming methods, each agent develops its own trajectory, interacting only with those other agents that it encounters, so the complexity of the problem is largely independent of the total population size.
- Deprived problems are those where resources are constrained. If the constrained resources are computational, the discussion of efficiency in section 2.3.2 applies. If communications bandwidth is the constrained resource,

this constraint deals with the same issue as Distribution, discussed in the next bullet.

- Distributed problems are those whose elements have different locations in some topology (in the case of geospatial reasoning, different geospatial locations). In such a configuration, many conventional approaches require a global analysis of the state of the agents. If the space being analysed is large, this constraint can require high-bandwidth long-range communications. Swarming approaches make extensive use of local interactions among agents that are near one another, and will succeed if these local interactions can propagate to produce the desired global effect.
- Decentralized problems are those that cannot rely on a central processor. Such a processor may be undesirable for several reasons. It may limit the scalability of the solution, or it may overload when the system becomes exceptionally busy, or it may pose a single point of failure that could limit the robustness of the system, or (in military applications) it may make the system vulnerable to a single attack. Because of the locality of swarming interactions, they avoid these problems.
- Dynamic problems are those that are changing fast enough that a solution based on a single estimate of the state of the system may not be useful. The issues here are those discussed under 'efficiency' in section 2.3.2.

An example of a scenario where applicability is more important than effectiveness or even efficiency is a network of unattended ground sensors monitoring for events that must be reported promptly. To keep the cost of individual sensors down and permit long battery life (deprived), communications must be severely limited (distribution), but the architecture must scale to large areas (decentralization). The need for prompt reporting precludes retrieving information from the sensors manually, and makes the problem dynamic. For researchers concerned with such problems, the ability of an algorithm to produce even approximate answers under the appropriate applicability constraints is a more important evaluation criterion than its effectiveness or efficiency in a limited laboratory setting.

3. Example applications

We describe three different systems that use swarming agents to solve geospatial problems. The first two, planning applications intended for robotic units, illustrate how swarming agents can develop paths that balance the influence of environmental threats and targets. The third, a predictive application, illustrates reasoning about more detailed topographical information. We comment on the behaviour of each example against the criteria of section 2.3.

3.1 *Path planning with threats and targets*

Consider an unmanned air vehicle (UAV) that must find its way around a network of surface-to-air missiles in order to reach a target. Figure 1 shows one possible configuration, in which a gauntlet of threats guards access to the target.

A common mechanism in robotics for path planning in this kind of problem is to define a loss function at each point in space on the basis of proximity to threats and targets, integrate it to generate a potential field, and then climb the field's gradient (Rimon and Kodischek 1992). Such methods require centralized computation, and so do not meet the applicability criterion for a distributed problem. They can also

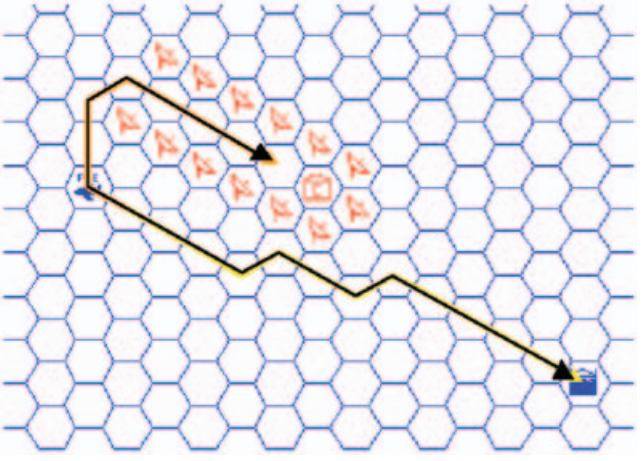

Figure 1. Path planning with threats (radar icons) and target (open house icon with 'C').

have difficulty solving configurations such as figure 1. The field can easily achieve a local maximum outside the gauntlet, trapping the hill-climbing search prematurely. In one experiment, researchers could only solve this configuration with standard potential methods by first manually defining a waypoint at the entrance to the gauntlet, and then planning the path in two segments, one from the base to the waypoint and the other from the waypoint to the target. Our approach exploits the recognized ability of stochasticity to break out of such local optima, as discussed in section 2.2.3. (In terms of section 2.3, our approach has greater relative effectiveness than the conventional potential field approach.)

This potential-based method is similar to the path-planning mechanisms used in current raster-based GIS systems (Douglas 1994). In general, these systems develop three successive layers:

1. a 'friction' or cost of traversal for each cell, analogous to the loss function used in robotic navigation;
2. the accumulated cost to reach each cell from a specified origin (the minimal path integral of the friction layer from the origin to each cell);
3. the least-cost direction from each cell toward the specified origin, developed by moving perpendicularly to iso-cost contours in the accumulated cost layer.

In this form, the algorithm does not recognize direction-dependent differences in the cost of crossing a cell. This difficulty has been addressed in various ways, including graph-theoretic flow algorithms (Stefanakis and Kavouras 1995), and iteration (Collischonn and Pilar 2000).

Our agent architecture for solving this problem uses digital pheromones, and has three components:

1. A distributed network of *place agents* maintains the pheromone field and performs aggregation, evaporation, and diffusion. Each place agent is responsible for a region of the physical space. In our simulation, we tile the physical space with hexagons, each represented by a place agent with six neighbours but in principle both regular and irregular tiling schemes can be employed. Place agents ideally are situated physically in the environment using unattended ground sensors distributed over an area and connected to

nearest neighbours through a wireless network. They may also be located in a distributed network of command and control nodes.

2. Each physical entity is represented by a software *agent*. Red agents represent enemy targets and threats. Each blue entity (a vehicle being routed) is represented by a *polyagent* (Parunak and Brueckner 2006a). A polyagent consists of a single persistent *avatar* and a swarm of transient *ghosts* that the avatar generates. The avatar runs on the UAV and manages the path production process. The ghosts wander over the place agents, looking for targets and continually building a path from their avatar to the target. The avatars and ghosts continuously deposit pheromones at their current locations.
3. Different classes of agents deposit distinct *pheromone flavors*. Agents can sense pheromones in the place agent in whose sector they reside as well as the neighbouring place agents. Brueckner (2000) develops the underlying mathematics of the pheromone field, including critical stability theorems.

Both avatars and their ghosts follow the gradient of a function computed over the pheromones in their vicinity. We will shortly describe the stochastic algorithm that ghosts use to follow this gradient. The ghosts on average tend to climb the pheromone gradient, but each explores a slightly different path. A polyagent's multiple stochastic ghosts thus reason about alternative possible experiences of the vehicle as it moves through space. The world is not deterministic, and plans (such as pre-planned paths) are rarely followed completely. Particularly in military operations, it is a truism that 'no plan survives contact with the enemy'. A sudden wind shear may force an aircraft off-course. A robot travelling along the contour of a slippery hill may slide off its planned trajectory. A previously unknown adversary may begin attacking a convoy, requiring it to detour. Such variations can transfer a moving entity from its pre-planned path to a location from which the best path to the destination is no longer the same as that originally planned. The swarming ghost agents explore many such alternative paths, and the density of the aggregate pheromone field that they generate is a probabilistic balance between the theoretical optimal path and the variation that may be forced on the entity as it travels.

Battlefield intelligence from sensors and reconnaissance activities causes the instantiation of red agents representing known targets and threats. These agents deposit pheromones on the places representing their location in the battlespace. The field they generate is dynamic, since targets and threats can move, new ones can be identified, or old ones can disappear or be destroyed. A blue avatar representing a UAV is associated with one place agent at any given time. It follows the pheromone path created by its ghost agents.

Ghosts initially wander through the network of place agents, attracted to pheromones deposited by targets and repelled by threat pheromones. Once they find a target, they return over the network of place agents to the walker depositing pheromones that help to build the shortest, safest path to the target. The basic pheromone flavors are *RTarget* (deposited by a Red target agent, such as the Red headquarters), *RThreat* (deposited by a Red threat avatar, such as an air defence installation), *GTarget* (deposited by a ghost that has encountered a target and is returning to its blue avatar, forming the path to the target), and *GNest* (deposited by a ghost that has left the blue avatar and is seeking a target).

A ghost agent chooses its next sector stochastically by spinning a roulette wheel with six weighted segments (one for each of its six neighbours). The size of each

segment is a function of the strength of the pheromones and is designed to guide the ghost according to the algorithm above. We experimented with several different forms of the function that generates the segment sizes. Manual manipulation yielded the current form (for outbound ghosts):

$$F_n = \frac{\theta \cdot RTarget_n + \gamma \cdot GTarget_n + \beta}{(\rho \cdot GNest_n + \beta)(Dist_n + \varphi)^{\delta + \alpha(RThreat_n + 1)} + \beta} \quad (2)$$

F_n is the resultant attractive force exerted by neighbour n, and *Dist* is the distance to the target if it is known. Table 1 lists the tuneable parameters in the equation and the effect that increasing the parameter has on the ghost's behaviour.

Though this table provides general guidance to the practitioner, in practice, the emergent dynamics of the interaction of ghost agents with their environment makes it impossible to predict the behaviour of the ghosts. Thus, tuning the parameters of this or any pheromone equation becomes a daunting task. We use synthetic evolution to adjust these parameters in real time, as the system is operating (Sauter *et al*. 2002, Parunak 2005). As the avatar emits new ghosts, it breeds them from the fittest ghosts that have already returned. Fitness takes into account three characteristics of those ghosts:

1. Ghosts have a fixed lifetime. Ghosts that complete their search faster have longer to breed, and generate more offspring. Thus, we favour ghosts that found shorter paths.
2. Ghosts encounter threats during their search. We favour ghosts that found safer paths.
3. Targets differ in value. We favour ghosts that found more valuable targets.

This system is extremely robust and adaptable (Parunak *et al*. 2004a) and has been deployed successfully on physical robots (Sauter *et al*. 2005). It can solve the scenario of figure 1 (among many others). It has several benefits in our application domain over the classical potential field algorithm and its GIS analogues. These benefits fall in the evaluation category of applicability (section 2.3):

- The swarming approach is *local*. It touches only the cells that the ghost agents actually visit, rather than computing fields across the entire space being modelled.
- Because it is local, the swarming approach naturally supports a *distributed* network of place agents (such as a network of unattended ground sensors with limited communications range and bandwidth). It does not require synchronized computation of successive layers of information, a feature of the

Table 1. Tuneable parameters and their effects on ghosts.

Parameter	Effect on ghost
α	Increases threat avoidance farther from the target
δ	Increases probability of ghosts moving towards a known target in the absence of RTarget pheromone
φ	Increases threat avoidance near target
ρ	Increases ghost exploration (by avoiding GhostNest pheromone)
θ	Increases attraction to RTarget pheromone
β	Avoids division by zero

potential field algorithm that makes it much better suited for centralized computation.

- The swarming approach is *dynamic*. Ghosts are continually emitted by their avatars as the avatars move, and any changes to the landscape during the course of the mission automatically vary the portion of the path not yet traversed. The potential field method plans a complete path based on a snapshot of the terrain being traversed. If the landscape changes, the user must decide whether to continue to use an old path that may no longer be optimal, or recompute a new path.
- The swarming approach is *stochastic*. The polyagent's ghosts explore a range of alternative trajectories for the robot, reflecting the uncertainty of movement in the physical world, and the path that is computed is a weighted combination of these trajectories. The potential field method is deterministic. It reflects the likely experience of the entity that is to follow the path only in the single loss, cost, or friction value that it assigns to its initial raster, and does not account for possible variation in the experience of the entity as it follows the path.

3.2 *Area surveillance*

A common task for uninhabited robotic vehicles is surveillance of a region of territory. Such surveillance must satisfy several characteristics. In this example, we focus on one: the vehicles should spread out over the area to avoid double coverage and reduce the time needed to cover the entire area. A convenient metric for such a system is how rapidly the agents initially cover the territory that they must monitor, tracking the fraction of the area that has been seen as a function of time.

A simple algorithm for this problem (Sauter *et al*. 2005) uses digital pheromones (Parunak *et al*. 2002a, b, 2004a). Unlike our other examples, each physical entity corresponds to only a single agent. The pheromone infrastructure represents the environment as a square grid, each cell of which has a place agent:

1. Once a second, each place agent deposits 20 units of attractive pheromone in its cell, propagates pheromone to the eight neighbouring cells, and evaporates the pheromone by a fixed proportion.
2. Every time a vehicle enters a new cell (on average, once every 4.8 s), it deposits two units of repulsive pheromone and zeros out the attractive pheromone in its current cell.
3. Once every 12 s, each place evaporates (but does not propagate) its repulsive pheromone by a fixed proportion.
4. Each vehicle moves to the neighbouring cell for which difference (attractive pheromone–repulsive pheromone) is greatest.

Agents' decisions use only the information available in their immediate vicinity and thus are local (though the propagation of attractive pheromone in step 1 provides some spread of information over time). In the absence of a vehicle, step 1 leads to an asymptotically constant level of attractive pheromone in each cell, drawing in vehicles. Step 2 causes the vehicles to spread out from one another and avoids repeat visits that are close to one another. Because step 1 repeats after step 2, and because the repulsive pheromone from step 2 evaporates over time, eventually each site will be revisited.

One could use a GIS to plan a shortest path that visits a sequence of locations with minimal repetition, a version of the travelling-salesman problem (Reinelt 1994), then follow this path repeatedly to keep the locations under surveillance. Our approach differs from this strategy in two important ways.

First, as with path planning, there is no guarantee that the surveillance vehicle will succeed in staying on the path. If it is forced off that path, it must find its way back and in the process may miss some locations that will remain unvisited until the next transit. If our approach misses a location, its attractive pheromone remains high, and continued pheromone growth raises its priority for visitation, reordering the vehicle's path dynamically.

Second, in an adversarial setting, the adversary can learn and evade a regular surveillance schedule. In our approach, the path changes in response to inevitable perturbations in the vehicle's movement. Thus, the surveillance is much more difficult to predict and evade.

In terms of section 2.3, these benefits are instances of applicability. In addition, our approach has highly desirable efficiency characteristics, discussed in more detail in section 4.2.

3.3 *Topographical prediction*

Our third application illustrates the use of swarming agents to predict the movements of entities that we do not control. In this example, insurgents are fleeing southward from friendly forces through a complex mountainous terrain. The task is to identify where in this terrain they are most likely to pass, so that surveillance assets can be deployed to detect and intercept them. Unlike the previous two examples, this example does not use digital pheromones but relies entirely on exogenous environmental variables (direction and gradient).

The sequence in figure 2 illustrates how swarming mechanisms can address this problem. The shading of the terrain indicates the steepness of the terrain. White terrain is level, while the steeper regions are successively darker shades of green, and the patch of red near the centre bottom is the steepest area.

The area under study is divided into a square lattice (200×200), and the gradient in each cell is computed on the basis of the elevations in its Moore neighbourhood (the eight adjacent cells). In addition, we compute a spatially smoothed gradient for each cell based on 5×5 Moore neighbourhoods.

We begin with a uniform distribution of 200 Red agents across the width of the battlespace along the northern edge, one per cell. At each time step, an agent assigns a weight to each of the cells in its 3×3 Moore neighbourhood, as the product of three values:

1. the inverse of the cell's gradient, normalized to sum to 1 over the neighbourhood;
2. the inverse of the cell's smoothed gradient, again normalized to 1;
3. a directional weight to encourage southward movement.

The directional weight is set heuristically, to reflect our estimate of how strong the insurgents' desire is to move southward. In the experiment reported here, we estimated that they would want to move southward 75% of the time, and 25% of the time might tolerate a lateral move. Empirically, we achieve this distribution of movement with weights (from NW to W clockwise) of (0, 0, 0, 0.13, 0.23, 0.28, 0.23, 0.13). In practice, we find that the behaviour of these systems is stable over a wide

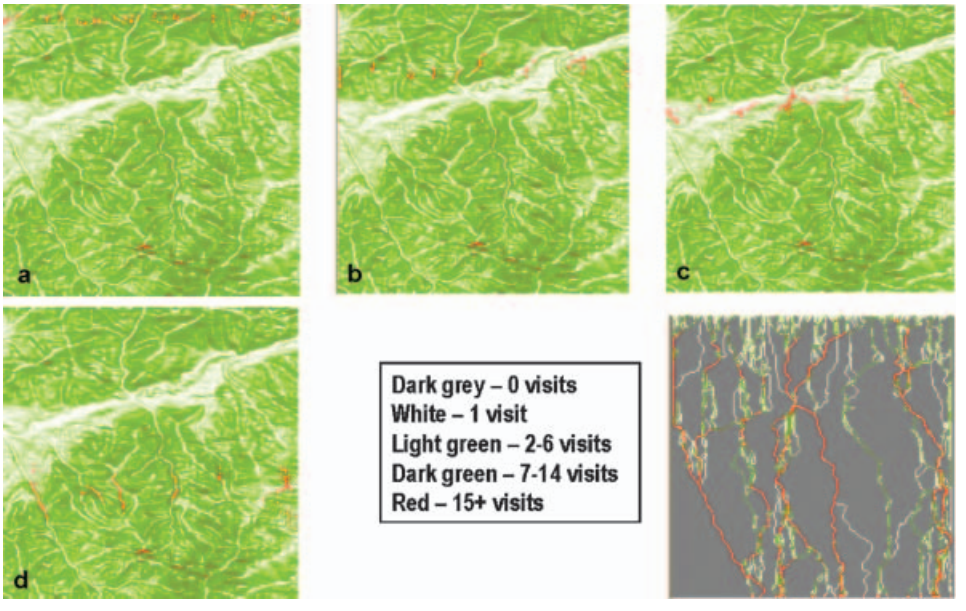

Figure 2. Successive stages in predicting movements in a complex topography.

range of such parameters. An evolutionary mechanism (Brueckner and Parunak 2003) can be used to search the parameter space efficiently for pathological regions.

The agent's current cell and the northern cell are omitted, the products are renormalized, and then the agent selects from among the five eligible directions using the Boltzmann–Gibbs distribution (equation (1)). In these experiments, we used a t of 0.01.

After only a short time (figure 2a), the result of the gradient information and the southward pressure guides the agents into clusters. By the time they reach the southern third of the territory (figure 2d), all 200 agents have merged into six groups. The final (lower right) plot shows the history of their distribution, emphasizing how the initial broad distribution rapidly narrows into only a few likely tracks.

This application is the closest in spirit of our three examples to traditional GIS processing, relying as it does primarily on topological information. Since we are not trying to control the red units, only to predict their movements, we are not as concerned about real-time variations in their behaviour as we were in the previous two cases. However, taking an agent-centred view of path generation rather than a terrain-centred view offers several interesting benefits, even in this application:

- As before, the stochastic nature of swarming agents provides a more robust path than a deterministic algorithm. In principle, this should offer a benefit in effectiveness (section 2.3.1), though we have not done a quantitative comparison with other methods.
- Swarming red agents could deposit attractive pheromones that in turn would route surveillance assets to the regions through which the Red forces are most likely to pass. In a deployed system, it would be advantageous to house these pheromones on unattended ground sensors, motivating the swarming

approach on grounds of applicability (in particular, distribution and decentralization).

- A slight modification of this algorithm can model an intelligent adversary that is familiar with the terrain. The evolutionary mechanisms used in UAV path planning could evolve individuals that are able to make it from the North to the South in the shortest time with the least likelihood of being detected by blue. The paths followed by such an individual might not be strictly the low-gradient paths. For example, the red agent might evade areas of known blue surveillance, or its stochastic exploration might discover that if one ascended a particular steep cliff using a rope ladder, one could connect useful path segments that otherwise would not offer a complete path. Answering several questions with the same set of mechanisms promises efficiency benefits in comparison with multiple specialized algorithms, though we have not conducted a quantitative comparison.

Supported by such a system, a robotic swarm could automatically perform highly directed surveillance activities against a knowledgeable enemy anywhere in the world without having to explicitly direct the units where to survey and how often they should monitor the different possible paths or choke points.

4. Discussion

In this section, we compare our methods with traditional GIS mechanisms, discuss their convergence, and suggest how the results of such analysis can be used.

4.1 *Comparison with traditional GIS mechanisms*

In describing our examples, we have given special attention to how they would be solved in a traditional GIS, and the constraints that motivate our approach. The fundamental distinction between our approach and traditional mechanisms is that the latter are centred on the terrain, while ours are centred on entities that move in the terrain. This difference is responsible for the contrasts that we have noted. In terms of section 2.3, most of these are benefits of applicability:

- Agents move, while the terrain is relatively static. Our focus on agents leads to an emphasis on real-time computation that can adapt to changes as they occur, while a focus on the terrain suggests pre-planning of more static structures. The latter approach is appropriate for engineering projects such as roads and aqueducts. The former is essential for robotic control.
- Computation intended to guide individual agents can be limited to the areas that they are likely to touch, rather than addressing attributes such as aggregate cost for an entire quadrant of landscape. This local focus encourages us to think in terms of distributed computational support, such as a network of unattended ground sensors, and such a distributed architecture in turn urges us to prefer mechanisms such as digital pheromones (which can be managed locally at an individual processor) over those that require repeatedly scanning an entire raster layer.
- We are sensitive to the vagaries of an agent's experience in the real world. It may not be possible to execute a pre-computed plan with complete accuracy, so we favour mechanisms that take into account a probabilistic sampling of alternative trajectories, rather than a single deterministic computation.

- Different agents can perform in different ways. Some GIS researchers explicitly acknowledge the impact that differences among mobile entities can have on the computation of a path (Balstrøm 2002). Our methods can explore these impacts directly, by implementing agents with different behavioural characteristics and testing the sensitivity of our geospatial conclusions to those variations.

Neither an agent-centred nor a terrain-centred approach is intrinsically superior. The preference for one over the other depends on the application. An agent-centred approach has benefits for real-time robotic control, while a terrain-centred approach has advantages for developing relatively stable infrastructure (such as power lines or highways). In an ideal world, tools for geospatial reasoning would incorporate mechanisms to support both perspectives on problems, enabling users to tailor each implementation to its intended application.

4.2 *Convergence speed*

A major justification of swarming is its potential for overcoming the combinatorial complexity of classical methods. However, the use of stochastic decisions raises the question of how rapidly a swarming system can itself converge.

A general model for the convergence of swarming systems (Parunak *et al*. 2005) is based on an extension of the adaptive walk (Kauffman and Levin 1987). Consider a binary vector $S \in \{0,1\}^N$ of length N. Initially, all elements of S are 0. This system seeks to maximize N_1, the number of elements of S that are 1. The adaptive walk repeatedly takes the following actions:

- Randomly select an element of S.
- If the element is 0, set it to 1 with probability p_{01}. If it is 1, set it to 0 with probability p_{10}. p_{01} and p_{10} are independent and need not sum to 0.

Analysis of the master equation for this system shows that

$$N_1 = p_{01}\left(1 - e^{-\lambda t}\right)/\lambda \tag{3}$$

where

$$\lambda \equiv (p_{10} + p_{01})/N \tag{4}$$

This simple model has features shared by many more realistic systems:

- Each element of S is an agent.
- The system objective is global over the entire system.
- The agents do not have access to this global measure in making their decisions. In this simple model, they do not consider the state of any other agent in making their decisions but choose probabilistically. p_{01} reflects the probability that their local decision will advance the global goal, while p_{10} reflects the likelihood that it will oppose the goal.

Though simple, this model can analyse the convergence of real systems. Consider the system of section 3.2, with 15 vehicles responsible for surveillance of an area 200 cells square. Figure 3 shows three plots of coverage as a function of time for this system: an upper bound, observed performance, and our model's estimate.

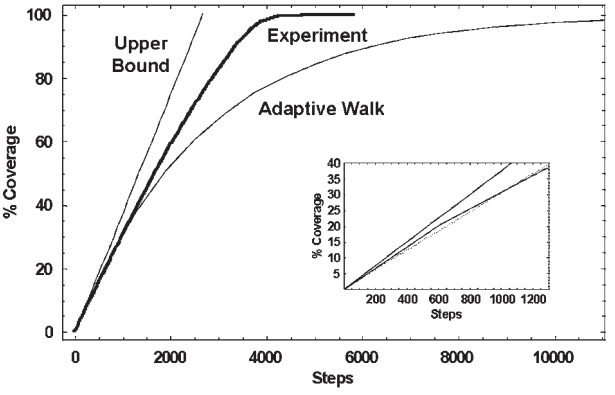

Figure 3. Comparison of adaptive walk with pheromone-guided area surveillance.

The time step in all cases is 4.8 s (the average time it takes a vehicle to move from one cell to another).

Ideally, at each time step, each vehicle would move immediately to an unvisited cell. This strategy is physically impossible, because it would sometimes require vehicles to move directly between non-contiguous cells. But it provides an upper bound, visiting 15 new cells or 15/40 000=0.0375% on each time step.

The experimentally observed convergence is nearly linear, with a slope of 0.030, until the coverage is almost saturated, well over 95%.

To fit our model, observe that 15 vehicles are sampling cells of a 40 000-cell area. Thus, $|S|=40\,000/15\sim2667$. For $p_{01}=1$ and $p_{10}=0$, $\lambda=3.75\text{E}{-4}$ (the same as the slope of the upper bound, and $N_1=2667\left(1-e^{-3.75E-4t}\right)$).

The model provides an excellent fit to the observations up to about 40% coverage. The model rises slightly faster than the experiment, then falls below it (inset). The model initially rises faster than reality because it is not constrained to move between contiguous cells. The adaptive walk slows as more and more of the area is covered, and random selection of the next site increasingly chooses a site that has already been visited, resulting in the shortfall above 40%.

The superior performance of our system above 40% shows the efficiency of the pheromone mechanism in improving over random selection of the next site to visit. Pheromones reduce the locality of the decision process in two ways. First, the propagation of attractive pheromone makes information from one cell available nearby, reducing spatial locality by generating a gradient that guides vehicles. Second, the persistence of pheromone deposited by one vehicle for sensing by another reduces temporal locality, enabling later decisions to build on the results of earlier decisions.

This example illustrates how the adaptive walk model can provide a lower bound for estimating the achievable performance of a real system, and for measuring the efficiency of mechanisms for overcoming locality.

4.3 *Use of results*

The examples we have presented illustrate two different applications of swarming geospatial reasoning. In this section, we discuss some of the issues involved in deploying these results.

The path planning and surveillance examples show how swarming can develop a *plan* to control physical hardware. Avatars controlling physical vehicles continuously manage a population of ghosts whose interactions emergently yield a solution to the path-planning problem. In this case, although there are many more ghosts than vehicles, the output of the system is a single recommended path for each vehicle.

An important consideration in control applications concerns the implementation of the environment that maintains the digital pheromones. The pheromone variables need to be maintained on processors that can update them (to provide evaporation and propagation), while remaining accessible to the agents. There are at least three workable alternatives:

1. A single central computer can maintain the pheromone infrastructure. While this approach limits the distribution and scaling of the system, it is the simplest. Because pheromone computations are so simple, in practice we can handle a system of 24 000 ghost agents on a grid of 40 000 cells on a single off-the-shelf laptop computer in real time.
2. At the opposite extreme, each region of space can be assigned its own processor. The most elegant approach is to embed the processors in the space, as unattended ground sensors that have on-board storage, processing, and communications. Only nearest neighbours need to communicate with one another, so power requirements can be limited.
3. Each avatar can maintain a pheromone map for regions it has recently visited, and exchange maps with other avatars when it comes near them. Agents move continuously through space, so the areas of most interest are those close to an agent's current location, which is the region for which an agent-based pheromone map will be most accurate.

The topographical reasoning example shows how swarming can *predict* behaviour by generating a probability distribution over possible futures. The frequency with which the agents visit different regions of the landscape is proportional to the probability that a single agent would traverse that region. Such an interpretation is useful in guiding the search for walkers of interest, or in planning traffic networks, among other applications.

We have described the utility of evolutionary methods in tuning control applications of swarming. Synthetic evolution is also useful in predictive applications. We are currently applying swarming geospatial prediction to the movement of soldiers in urban combat (Parunak 2005, Parunak and Brueckner 2006b). We begin the swarming simulation in the past, and adjust the individual parameters of each ghost to fit the observed recent behaviour of the corresponding real-world entity. Then, we allow the fittest ghosts to run into the future to form our prediction. This mechanism allows us to base our predictions on a much richer model of the individual agent's behaviour than would otherwise be possible, without the need for time-consuming knowledge acquisition.

An important feature of our approach is that the same underlying mechanisms (emulation of situated agents) provide both planning and prediction. For the sake of human observers, it is useful to reduce the observed movements of the agents to a symbolic description of the prediction or the plan, but the agents themselves need no such description, In particular, military applications often have two populations of swarming agents. One population represents the adversary, and its movements

constitute a prediction about likely adversarial behaviour. The other population represents friendly troops, and its behaviour is a seminal plan for deliberate action. Human decision-makers may plan their actions based on an explicit articulation of the perceived enemy plan, but within the model, the population representing friendly troops needs no such articulation. It responds directly to the behaviour of the adversarial agents, and vice versa. By staying within the sub-symbolic domain, the model can execute extremely rapidly, avoiding the combinatorial bottleneck of requiring agents to reason symbolically over representations of plans.

5. Conclusion

Swarming methods are a fruitful resource for reasoning about the movements of entities constrained by topological or topographical features of the environment. By making disciplined use of large populations of agents with greatly simplified internal logic, appropriate application of stochastic decisions, and stigmergic information exchange, we can solve problems that would be intractable by classical enumerative techniques and prohibitively expensive to implement with more sophisticated agent-based simulation. These methods converge with reasonable speed and can support both robotic control and prediction of natural systems. Their agent-centric view of geospatial reasoning complements the terrain-centric view more common in traditional GIS systems and is particularly suited to applications that must run in real-time, on distributed networks of processors without central control.

References

Balstrøm, T., 2002, On identifying the most time-saving walking route in a trackless mountainous terrain. *Geografisk Tidsskrift*, **102**, pp. 51–58.

Brueckner, S., 2000, Return from the ant: synthetic ecosystems for manufacturing control. Dr.rer.nat. thesis, Humboldt University Berlin.

Brueckner, S. and Parunak, H.V.D., 2003, Resource-aware exploration of emergent dynamics of simulated systems. In *Autonomous Agents and Multi-Agent Systems (AAMAS 2003)*, Melbourne, Australia (ACM), pp. 781–788.

Collischonn, W. and Pilar, J.V., 2000, A direction dependent least-cost-path algorithm for roads and canals. *International Journal of Geographical Information Science*, **14**, pp. 397–406.

Douglas, D.H., 1994, Least-cost path in GIS using an accumulated cost surface and slope lines. *Cartographica*, **31**, pp. 37–51.

Ferber, J. and Müller, J.-P., 1996, Influences and reactions: a model of situated multiagent systems. In *Second International Conference on Multi-Agent Systems (ICMAS-96)*, Kyoto, Japan (AAAI), pp. 72–79.

Finin, T., 1997, *UMBC KQML Web*, T. Finin (Ed.). In University of Maryland, Baltimore Campus. Available online at: http://www.cs.umbc.edu/kqml/ (accessed 20 July 2006).

FIPA 2000, FIPA Agent Communication Language Specifications. Available online at: http://www.fipa.org/repository/aclspecs.html (accessed 28 September 2000).

Garey, M.R. and Johnson, D.S., 1979, *Computers and Intractability* (San Francisco, CA: W.H. Freeman).

Gimblett, H.R. (Ed.), 2002, *Integrating Geographic Information Systems and Agent-based Modeling Techniques for Simulating Social and Ecological Processes*, Santa Fe Institute Studies in the Sciences of Complexity (New York: Oxford University Press).

Grassé, P.-P., 1959, La Reconstruction du nid et les Coordinations Inter-Individuelles chez *Bellicositermes Natalensis et Cubitermes* sp. La théorie de la Stigmergie: Essai d'interprétation du Comportement des Termites Constructeurs. *Insectes Sociaux*, **6**, pp. 41–84.

HADDADI, A. and SUNDERMEYER, K., 1996, Belief–desire–intention agent architectures. In *Foundations of Distributed Artificial Intelligence*, G.M.P. O'Hare and N.R. Jennings (Eds) (New York: Wiley), pp. 169–185.

KAUFFMAN, S.A. and LEVIN, S., 1987, Toward a general theory of adaptive walks on rugged landscapes. *J. Theoret. Biol*, **128**, pp. 11–45.

KIRKPATRICK, S., GELATT, C.D. and VECCHI, M.P., 1983, Optimization by simulated annealing. *Science*, **220**, pp. 671–680.

MICHEL, F., 2004, Formalisme, méthodologie et outils pour la modélisation et la simulation de systèmes multi-agents. Doctorat thesis, Université des Sciences et Techniques du Languedoc.

MÜLLER, J.P., 1996, *The Design of Intelligent Agents*, Lecture Notes in Artificial Intelligence 1177 (Berlin: Springer).

PARUNAK, H.V.D., 1997, 'Go to the ant': Engineering principles from natural agent systems. *Annals of Operations Research*, **75**, pp. 69–101.

PARUNAK, H.V.D., 2003, Making swarming happen. In *Swarming and Network-Enabled C4ISR* (Tysons Corner, VA: ASD C3I).

PARUNAK, H.V.D., 2005, Evolving Swarming Agents in Real Time. In *Genetic Programming Theory and Practice (GPTP05)* (Ann Arbor, MI: Springer).

PARUNAK, H.V.D. and BRUECKNER, S., 2006a, Modeling Uncertain Domains with Polyagents. In *International Joint Conference on Autonomous Agents and Multi-Agent Systems (AAMAS'06)*, Hakodate, Japan (ACM).

PARUNAK, H.V.D., BRUECKNER, S. and SAUTER, J., 2004a, Digital Pheromones for Coordination of Unmanned Vehicles. In *Workshop on Environments for Multi-Agent Systems (E4MAS 2004)*, pp. 246–263 (New York: Springer).

PARUNAK, H.V.D., BRUECKNER, S. and SAVIT, R., 2004b, Universality in multi-agent systems. In *Third International Joint Conference on Autonomous Agents and Multi-Agent Systems (AAMAS 2004)*, pp. 930–937 (New York: ACM).

PARUNAK, H.V.D. and BRUECKNER, S.A., 2004, Engineering swarming systems. In *Methodologies and Software Engineering for Agent Systems*, F. Bergenti, M.-P. Gleizes and F. Zambonelli (Eds), pp. 341–376 (Dordrecht, Netherlands: Kluwer).

PARUNAK, H.V.D. and BRUECKNER, S.A., 2006b, Extrapolation of the Opponent's Past Behaviors. In *Adversarial Reasoning: Computational Approaches to Reading the Opponent's Mind*, A. Kott and W. McEneaney (Eds) (Boca Raton, FL: CRC Press).

PARUNAK, H.V.D., BRUECKNER, S.A. and SAUTER, J., 2002a, Digital Pheromone Mechanisms for Coordination of Unmanned Vehicles. In *First International Conference on Autonomous Agents and Multi-Agent Systems (AAMAS 2002)*, Bologna, Italy, pp. 449–450 (New York: ACM).

PARUNAK, H.V.D., BRUECKNER, S.A., SAUTER, J.A. and MATTHEWS, R., 2005, Global convergence of local agent behaviors. In *Fourth International Joint Conference on Autonomous Agents and Multi-Agent Systems (AAMAS05)*, Utrecht, The Netherlands, pp. 305–312 (New York: ACM).

PARUNAK, H.V.D., PURCELL, M. and O'CONNELL, R., 2002b, Digital pheromones for autonomous coordination of swarming UAV's. In *First AIAA Unmanned Aerospace Vehicles, Systems,Technologies, and Operations Conference* (Norfolk, VA: AIAA).

PARUNAK, H.V.D., SAVIT, R. and RIOLO, R.L., 1998, Agent-based modeling vs. equation-based modeling: a case study and users' guide. In *Multi-Agent Systems and Agent-Based Simulation (MABS'98)*, Paris, FR (New York: Springer), pp. 10–25.

RAO, A.S. and GEORGEFF, M.P., 1991, Modeling rational agents within a BDI architecture. In *International Conference on Principles of Knowledge Representation and Reasoning (KR-91)* (San Mateo, CA: Morgan Kaufman), pp. 473–484.

REINELT, G., 1994, *The Traveling Salesman: Computational Solutions for TSP Applications*, Lecture Notes in Computer Science, 840 (New York: Springer).

RIMON, E. and KODISCHEK, D.E., 1992, Exact robot navigation using artificial potential functions. *IEEE Transactions on Robotics and Automation*, **8**, pp. 501–518.

SAUTER, J.A., MATTHEWS, R., PARUNAK, H.V.D. and BRUECKNER, S., 2002, Evolving adaptive pheromone path planning mechanisms. In *Autonomous Agents and Multi-Agent Systems (AAMAS02)*, pp. 434–440, Bologna, Italy (New York: ACM).

SAUTER, J.A., MATTHEWS, R., PARUNAK, H.V.D. and BRUECKNER, S.A., 2005, Performance of digital pheromones for swarming vehicle control. In *Fourth International Joint Conference on Autonomous Agents and Multi-Agent Systems*, Utrecht, Netherlands (New York: ACM), pp. 903–910.

STEFANAKIS, E. and KAVOURAS, M., 1995, On the determination of the optimum path in space. In *The European Conference on Spatial Information Theory (COSIT 95)*, Semmering, Austria (New York: Springer).

STERMAN, J., 2000, *Business Dynamics* (New York: McGraw-Hill).