

Ant Hunt: Towards a Validated Model of Live Ant Hunting Behavior

Yu-Ting Yang and Andrew Quitmeyer and Brian Hrolenok
Harry Shang and Dinh Bao Nguyen and Tucker Balch
Georgia Institute of Technology
Atlanta, GA

Terrance Medina and Cole Sherer and Maria Hybinette
University of Georgia
Athens, GA

Abstract

Biologists seek concise, testable models of behavior for the animals they study. We suggest a robot programming paradigm in which animal behaviors are described as robot controllers to support a cycle of hypothesis generation and testing of animal models. In this work we illustrate that approach by modeling the hunting behavior of a captive colony of *Aphaenogaster cockerelli*, a desert harvester ant. In laboratory animal experiments we introduce live prey (fruit flies) into the foraging arena of the colony. We observe the behavior of the ants, and we measure aspects of their performance in capturing the prey. Based on these observations we create a model of their behavior using Clay, a Java library developed for coding hybrid controllers in a behavior-based manner. We then validate that model in quantitative comparisons with the live animal behavior.

Introduction

Biologists and cognitive scientists seek succinct, descriptive models of behavior for the animals they study. In this work we propose that robot programs, specifically hybrid controllers, can serve as *complete* and *testable* models of animal behavior. By *complete* we mean that the model fully explains the animal's behavior from sensation to action, and by *testable* we mean that the model provides an hypothesis that can be tested experimentally. We know that robot programs offer these properties because they fulfill exactly those purposes for the artificial creatures they were designed to control.

We are not the first to propose this approach. Others, including Webb and Mataric, have proposed robots as a testbed for animal behavior research (Webb 2002; Mataric 1998). Webb investigates the behavior of crickets, and she builds robots that mimic cricket behavior to validate her animal models. Mataric is a robotics researcher who has drawn inspiration from animals for her robot designs. We follow a similar approach in our work where we leverage the design philosophy of behavior based control (Arkin 1998). But rather than designing our programs to run on robots, we run

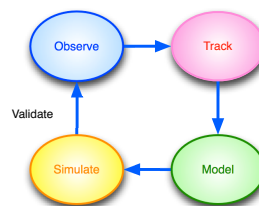


Figure 1: Our overall approach is to observe, track, and generate a model from observations, run the model in simulation and then finally validate the generated model.

them in simulation. We envision eventual implementation on robots, but we also welcome the ease simulation provides for prototyping the behavior of hundreds or thousands of agents.

Our overall approach is to *observe* and *track* a system (e.g., an ant colony) over time then from these observations generate an agent based model that runs in simulation (See figure 1). The last step includes constructing experiments of both the physical system (e.g., the observed live ants) and the corresponding modeled system (e.g., simulated ants) to verify whether the model corresponds to the physical system (i.e., validating the correctness of the generated model). In the work presented here we are focusing on the validation step, specifically in validating our model of live ant hunting or foraging behavior.

It would of course require quite a tremendous effort to create a single model that explains *all* aspects of behavior of a particular organism. Rather than attacking such a task all at once, we break the problem into smaller chunks by modeling particular aspects of an animals overall behavior individually. For this work we focus on the foraging behavior of *Aphaenogaster cockerelli* a desert harvester ant.

We follow a similar approach as myrmecologist Deborah Gordon in her work with another desert ant species: First, on the basis of observation she hypothesizes a quantitative model to predict or explain a measurable outcome. Next, she perturb the experiment in some way, perhaps by adding obstacles or additional food objects (Gordon 1999). If the model is accurate it will correctly predict the outcome of the experiments with live ants. In our work however, the model is a *program* instead of an equation. Furthermore our ap-

proach is more fine grained, dealing with individual behaviors (a behavior of a single ant in a colony, such as the sensor range of a particular ant), rather than broad stroked behaviors across an entire colony in Gordon’s approach, such as the average speed of the ants in the entire colony. Before discussing details of our approach we will give an overview of background material and review the literature.

Background and Related Work

How Biologists Model Behavior

Understanding and predicting the behavior of a colony is a challenging task. Biologists approach this in at least two ways: 1) By creating mathematical models that predict measures of overall colony behavior, and 2) By creating functional models of behavior that illustrate flows or sequences of behavior. The second group of models are often graphical in nature.

Gordon (Gordon, Paul, and Thorpe 1993) developed experiments to explore the patterns of brief antenna contacts in the organization of ant colonies. She created a mathematical model to predict encounter rates as a function of their densities. Her hypothesis is that this is a non-quadratic law. Her work shows that contact rate was not random (Gordon 1996; Gordon and Mehdiabadi 1998). The behavior of the colony arise from the behavior of individuals. The internal and external factors outlined above contribute to individual decisions about task performance.

Theraulaz also explored encounter rates (in a different species). He developed a kinetic model of encounters between individuals. and conducted an experiment with different densities of animals in 2004 (Nicolis, Theraulaz, and Deneubourg 2005). He observed that encounter rate is amplified when the number of individuals involved in the aggregates increases.

When creating graphical models of behavior, ethologists look for commonly repeated activities which they classify as behavior primitives. For example, a duck might have behavior primitives such as wag tail feathers, swim and shake head. The sequence of these activities and proportion of time the animals spend performing each primitive are recorded in a table called an ethogram (Schleidt et al. 1984). An ethogram can be depicted graphically as a *kinematic graph*, which is similar to a discrete-event Markov Chain. The graph traces the probability of performing a next behavior given the behavior that the animal is currently engaged in.

A sample ethogram of ant behavior is provided in Figure 2 (Hölldobler and Wilson 1990). The nodes of this diagram represent the behavioral acts of individual animals. The links between the nodes show how behaviors are sequenced. The frequency of observed transitions is also recorded and represented.

Executable Models of Behavior

Ethograms are among the most effective and frequently used approaches by behavior ecologists in describing and predicting behaviors in a qualitative sense (Schleidt et al. 1984). But because they are not complete models they suffer from

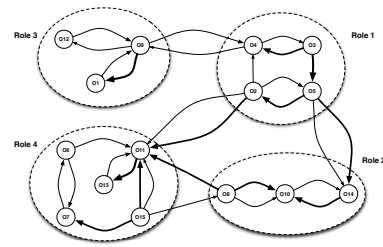


Figure 2: An ethogram of individual ant behavior, In this ethogram, behavioral acts are linked by arcs indicating transitions from act to act. Thicker lines indicate higher probability transitions (from (Hölldobler and Wilson 1990)).

difficulty in validation. In ethograms, for instance, the transition from behavior to behavior is represented as a probability. However, these transitions in real animals are usually a consequence of perceptions encountered by the animals.

Researchers in controls and robotics have begun to leverage a control-theoretic approach to modeling of behavior. Arkin, for instance simulates wolf hunting behavior with five behavioral states: *Search*, *Approach*, *Attack Group*, *Attack Individual* and *Capture*. He also built a probability table to express the transitions between the five states (Madden, Arkin, and MacNulty 2010). In his later work, Arkin et al built two probability transition table for heterogeneity aged wolves simulation (Madden, Arkin, and MacNulty 2010). He found that in the composition of a heterogeneous team, a high performing composition can be found.

Haque and Egerstedt have investigated the cooperative behaviors of the bottlenose dolphin *Tursiops truncatus* (Haque, Rahmani, and Egerstedt 2009). These dolphins utilize two different strategies for their foraging tasks: the “wall method” and the “horizontal carousel method.” The authors modeled these techniques with a hybrid controller and by using decentralized networked controls they successfully replicated the fish hunting scenario in simulation.

In this work we propose to leverage techniques from behavior-based robotics to model animal behavior. Behavior-based robotics, defined in previous work by (Brooks 1991) and (Arkin 1998), is built on the idea of combining primitive behaviors to define more complex behaviors in a bottom-up fashion. Computationally simple behavior primitives produce vectors of motion, for instance pointing away from an obstacle to represent repulsion, or pointing toward a goal to represent attraction. The ultimate behavior or decision of the robot is made by aggregating the results of the primitive behaviors. By applying behavior-based robotics techniques to animal modeling, we can program the models and run them within a multi-agent based simulation, which allows us to validate a model.

Agent Based Model (ABM) Simulation Systems

Agent-based modeling simulation systems (ABM) can provide a useful platform for evaluating behavior-based models. An ABM consists of multiple autonomous processes, or agents, that execute in parallel within a simulated environment. Each agent typically follows a sense-think-act cycle within the simulation that is, it first takes input from the sim-

ulated environment (senses), processes that input (thinks) and uses the result to modify the environment or its position within the environment in some way (acts).

ABMs have proven useful in simulating economic, social and biological systems, as well as complex queuing systems like airport runways and computer networks. A number of ABM simulation systems have been developed. We review a few of the most relevant here. The Swarm simulation system developed by the Santa Fe Institute (Minar et al. 1996) is a general purpose simulator that takes groups of agents, or *swarms*, as the fundamental unit of simulation. Notably, a swarm is defined as a set of agents which are unified through a schedule of events. If we consider an agent to be a set of primitive behaviors, as in behavior-based robotics, it is easy to see how a swarm can constitute not only a set of agents but also a set of primitive behaviors that define an agent. Furthermore, both an agent and its environment are defined as swarm objects. This notion of multi-level modeling is a powerful feature of the Swarm system. But Swarm also suffers from maintainability issues (Luke et al. 2005) and scalability limitation (Hybinette et al. 2006).

MASON (Luke et al. 2005) is an agent-based simulator that was inspired by early robotics simulators like Teambots (Balch 1998), but was developed as a general-purpose simulator similar to SWARM. Unlike SWARM, MASON is scalable up to a million of simple agents and maintains a distinction between agents and their environment. MASON's most notable feature is the separation of the model engine from the visualization of the model, which allows for many runs on multiple machines with dynamically configurable visualizations. However, MASON's application programmer interface (API) is not physically realistic or rich.

Finally, SASSY, developed at the University of Georgia, is an ABM that places importance on both scalability and runtime performance. SASSY provides middleware between a Parallel Discrete-Event Simulation kernel (PDES) and an agent-based API. SASSY's kernel is based on the Time-Warp algorithm, which allows multiple threads of events to play out optimistically, and then rolls them back to a previous state when a conflict between threads arises. This synchronization of multiple scheduling threads allows for a highly scalable simulator with significant performance improvement in distributed and multiprocessor environments as more processing elements are introduced.

In this work we use Clay (Balch 1998), a library for coding behavior-based controllers and BioSim an ABM simulation system extending Clay. As future work the agent based system can run without code modification on SASSY to improve scalability further by a provided plug-in (Sherer, Vulov, and Hybinette 2011). Currently, SASSY includes a MASON plug-in.

Hunting Behavior of Captive Ants

For this work we chose to focus on and model the behavior of *Aphaenogaster cockerelli* while they forage for, subdue, and collect *Drosophila melanogaster* (fruit flies). This particular aspect of their behavior is interesting for our study because it is not overly complex, but includes several distinct components, including aspects of collaboration. Addi-

tionally, this behavior can be observed rather easily in a laboratory setting. We refer to this activity as "hunting" rather than foraging because it involves collection of prey.

We observed the animals' hunting behavior over several weeks. Details of the experiments are outlined below. The behaviors the ants exhibited while hunting was similar among all the individual ants. Generally the ants wander about the arena, pausing to interact with other ants occasionally. When encountering a fruit fly, an ant will initially attempt to grapple it with its front legs and mandibles. If the fly escapes after the initial grapple attempt, the ant will usually quickly follow spiral path outward from the point at which the fly was lost - sometimes interacting with other ants as it encounters them - until it finds another (or possibly the same) fly.

We hypothesize that the spiral activity represents both a way to search for the prey, but that it may also represent a method for recruiting other ants into the hunt. Other ants encountered during this spiraling behavior tend to follow the recruiting ant.

After a successful grapple, the ant will generally head towards the nest, but not necessarily in a straight line. While returning, it will try to avoid interacting head-to-head with other ants. If the ant with the captured fly does interact with another ant head-to-head, the other ant will occasionally try to take the fly from it. The longer an ant takes to return home, the less likely it is to avoid other ants, which increases the chance that the fly will be passed from one ant to another, which may be a mechanism to ensure that food will be returned to the nest even if the ant which captured the prey is "lost".

We describe more details of the live ant behavior in the next section. We have modeled this ant foraging behavior with Clay, a behavior-based robot architecture. We describe our model in detail next.

An Ant Hunt Behavior-Based Model

After carefully noting the hunting behavior of our live ant colony, we modeled their behavior for simulation as a hybrid robot controller using Clay (Balch 1998).

The Clay architecture defines "motor schemas" and "perception schemas" as its behavior primitives, which are combined to form "behavior assemblages" which may be reused and recombined. Clay integrates this motor-schema based control with reinforcement learning, which allows robot agents to select behavior assemblages based on accumulated reward values for past performance.

Clay's motor-schemas are computationally simple behavior primitives that produce vectors of motion, for instance pointing away from an obstacle to represent repulsion, or pointing toward a goal to represent attraction. The ultimate behavior or decision of the robot is made by aggregating the results of the primitive behaviors. For example, a set of vectors radiating around an obstacle combined with a greater magnitude vector pointing toward a goal on the opposite side of the obstacle would cause the robot to nudge or dodge around the obstacle while still making forward progress toward the goal. Perception schemas provide sensor input processing specific to each motor schema.

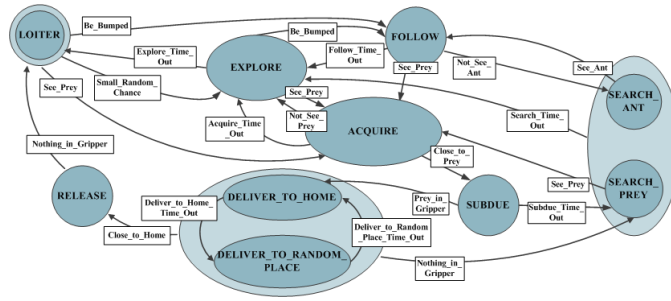


Figure 3: Hunting Behavior of Ants

A set of behavior assemblages constitute a set of states, and a modeled entity may be in one of these states at any given time. When we add transitions between states, we have the Finite State Machine (FSM) shown in Figure 3. Note that this is similar to the Markovian Kinematic diagram or ethogram used by ethologists, with the principal difference being that we include both probabilistic transitions (e.g., from the state LOITER to the state EXPLORE) and deterministic transitions based on external events (e.g., sensing a prey enters the ACQUIRE behavior), whereas the traditional ethological approach is to consider only probabilistic transitions between states that are completely independent of outside causes (Schleidt et al. 1984).

When the simulation is initialized, the ant starts out in the LOITER behavior. In the LOITER state, ants appear to be stuck in place or slowly rotating around. Ants sometimes leave this state suddenly and with no apparent reason, but most of the time they will leave the state because of some external influence, such as being bumped by another ant, which initiates the FOLLOW state or by sensing prey nearby, which causes the ant to ACQUIRE the prey. If neither of these happen, the ant may randomly enter the EXPLORE behavior, which simulates ants trying to find their prey. The ant leaves this state when it sees the prey, or is bumped by other ants. It may also leave this state when its biological timer runs out, which returns the ant to the LOITER behavior. The FOLLOW behavior simulates ants trying to follow an ant that has bumped into it. If the ant loses track of the ant it is following, it enters the SEARCH ANT behavior. If it finds the ant again it reenters the FOLLOW behavior, but if the search continues too long, then the ant will stop searching and go back to the EXPLORE behavior.

The ACQUIRE behavior simulates ants trying to approach and capture their prey. When the ant moves close enough to the prey, it enters the SUBDUE behavior. The prey is usually close to the ant, but if for some reason the ant misses the prey it goes back to the EXPLORE behavior to restart the search. If the ant catches the prey successfully, it enters the DELIVER TO HOME behavior, which simulates an ant carrying its prey directly to the home base. If the ant takes too long to deliver the prey (for instance, if it encounters obstacles), the ant enters the DELIVER TO RANDOM PLACE behavior, which makes the ant carry its prey in a random direction for a period time, then retry DELIVER TO HOME. If the ants biological time runs out before it has captured the prey, it goes back to the SEARCH PREY be-

havior. If the search time takes too long, then the ant will stop search and do the EXPLORE behavior. When an ant finally returns to home base with prey in mandible, it initiates the RELEASE behavior.

Experimental Methods

To generate and fine tune our simulation model we conducted experiments both with live ants and with simulated ants. We tuned both the behavioral parameters and the experimental configuration of the simulated ants in a base case scenario (live ants loitering without obstacles). After tuning, we have a hypothetical ant behavioral model. We test the hypothesis by perturbing the experiments with live ants and compare the performance in simulation of the perturbed model. In the next section we describe the setup and experiments with the live ants, both the base case environment (an environment with no obstacles) and a perturbed case environment (an environment with obstacles).

Live Ants

The core experiment consists of adding flightless fruit flies (*Drosophila melanogaster*) to an arena of the predatory ants (*Aphaenogaster cockerelli*) and observing the interactions between the two groups of agents. Metrics generated from this observed, real-world behavior will be used to determine the verisimilitude of a virtual simulation.

Setup The ants typically live in two, 270mm x 194mm, rectangular plastic boxes connected by short 20mm tubing. All boxes have edges coated in a Teflon paint (Fluon), to prevent escapes and maintain the two dimensionality of the observation fields. One chamber features a narrow glass ceiling and functions as the “nest”, while the other serves as a “nest entrance” and has an open top for generic feeding. A third box, with a single access point is added to the far end of the entrance arena to work as an empty “foraging arena.”

A Canon 550D camera (with 50mm f2.0 lens) is mounted directly above to observe the foraging arena. Lighting comes from two nearby heatless, dual-necked, fiber optic microscope illuminators, and a large incandescent flood light mounted farther away (to soften shadows). The outside bottom of the arena is painted matte white to provide stark contrast with the black ants and fruit flies.

Experiment : Live Ants without Obstacles The experimental ant colony is kept on a rigid feeding schedule, and

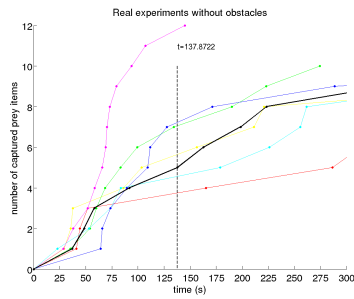


Figure 4: Live ants hunting fruit flies

experiments are run consistently within this schedule to minimize overall behavioral changes from one set of observations to the next. At the beginning of an experimental trial, five minutes of video are collected of the ants simply milling about in the empty foraging arena. This can allow us to further normalize between trials based on initial ant density and global activity level. Next, 10 flightless fruit flies are dropped from a vial into the center of the arena. The ants typically collect all fruit flies within 5 minutes, but the camera records their activity for a full 10 minutes. After a 5 minute break the same procedure is repeated but with 15 fruit flies. At the end of the experimental day, the three videos are saved to a centralized server for future processing. This experiment is performed for five days at the same time each day, resulting in 15 total videos.

Experiment : Live Ants with Obstacles The entire five day experiment is then repeated with one single alteration, six immovable obstacles are added to the foraging arena. The obstacles are 24mm diameter glass cylinders coated in Fluon to prevent ants from climbing.

The ants' hunting behaviors are analyzed between the two sets of experiments, and within each group of trials to determine acceptable metrics for analyzing the performance of a computer simulation of the predator/prey activity.

Adapting the Model

The simulated experimental setup is similar to the live ant setup, i.e., the arena is 270mm x 194mm, rectangular box. The simulated creatures have similar dimension to their live counterparts, where an ant is 8.6mm long with a moving speed of 0.007 m/sec and a fruit fly is 2.5mm long, with moving speed of 0.0005 m/sec. The right image of Figure 6 shows simulation environment of ants hunting, and the image to the left shows the live ants. The simulation is initialized with forty ants and ten fruit flies. Fruit flies, like the live experiments are placed in the center of the arena.

In our experiments we measure the average time to capture half of prey. We chose this metric as opposed to measuring the time to collect all the prey (or flies) because the latter value vary substantially when only a few flies can perturb the completion time by repeatedly escaping capture or finding a particular good hiding spots.

The simulation consists of two tunable parameters, one is called **subdued** and the other is called **successful delivery**. Subdued sets the probability of a successfully capture of a

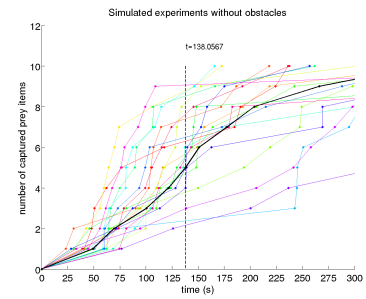


Figure 5: Simulated ants hunting fruit flies

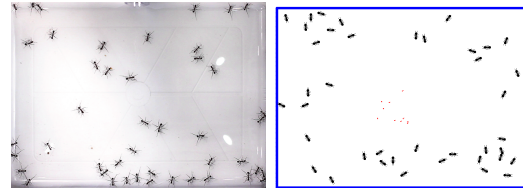


Figure 6: Live hunting ants on left and the simulated ants on right

fly, while the successful delivery parameter sets the probability whether the fly is successfully delivered to the ants' nest. In our experiments subdued is set to a 1% capture probability, and successful delivery is set to a .3% probability.

Figure 5's plot to the left shows the performance of live ants delivering prey to its nest and the plot to the right shows simulated ants delivering prey to its nest. Each point in the plots represent the time to deliver one fly from the arena to the nest (home). The left plot shows that live ants deliver half its prey to the nest in 137.87 seconds on average while the plot on the right simulated ants deliver half its prey is 138.06 seconds on average, less than 1% difference (after tuning). For now the comparison are at a qualitative level rather than a quantitative.

Before tuning our simulated environment we noticed that the average time it took to collect half of the prey was less than the time in the real animal experiment, without tuning simulated ants took 68.96 seconds on average to collect half the prey. We adjusted two parameters of the simulated ants to address this. First we reduced the sensing range of the simulated ants from 25.8mm to 7.16mm, this change had the effect of slightly increasing the a time to collect prey, but it did not make a significant difference.

We then looked for other parameters to calibrate the simulator. We noticed that in our simulation environment ants were roaming uniformly over the arena, while live ants tended to congregate near the walls of the arena. When we adjusted this factor by changing the ants initial location to near to the wall, while keeping the detection distance at 7.16mm. With this adjustment the simulated performance became close to that result measured of the live ants (i.e. 137.87 for live ants and 138.06 for simulated ants).

These data were analyzed as described above. For each ant, we noted that detection distance affect the total time of hunting, especially when the flies were at remote locations

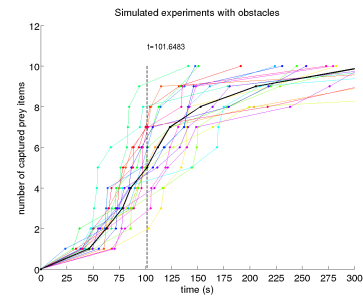
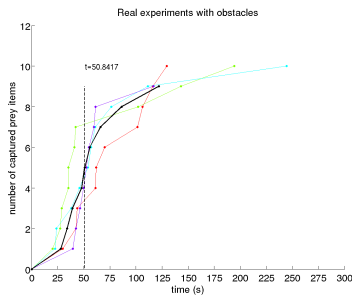


Figure 7: Simulation ants hunting with obstacles.

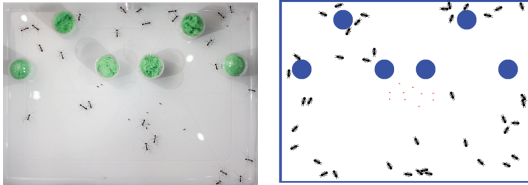


Figure 8: Real ants on left and simulated ants on right from the nest. The initial location affected the average time of capture the prey significantly.

Conclusion: Validating the Model With Obstacles

After tuning our model to match the performance of real ants we tested the validity of the model by perturbing the experimental environment. In particular we added obstacles between the area where the prey initially placed and the home base of the ants. We expected that that obstacles would have a net effect of slowing down the total collection time, but in experiments with the live ant colony we discovered that the collection time actually reduced.

We tested the model in simulation by replicating these conditions (adding obstacles) and we discovered that for the simulated ants deliver time was also reduced (from 137.8 sec to 50.8 seconds for real ants, and from 138.1 sec to 101 sec for simulated ants). So, in a qualitative sense the model was validated in these experiments as the time to collect prey with obstacles is reduced in both live and simulated ants.

References

Arkin, R. 1998. *Behavior-based robotics*. The MIT Press.

Balch, T. 1998. Behavioral diversity in learning robot teams. PhD Thesis, Georgia Tech, Atlanta, Georgia.

Brooks, R. 1991. Intelligence without representation. *Artificial intelligence* 47(1-3):139–159.

Gordon, D. M., and Mehdiabadi, N. J. 1998. Encounter rate and task allocation in harvester ants. *Behavioral Ecology and Sociobiology*.

Gordon, D.; Paul, R.; and Thorpe, K. 1993. What is the function of encounter patterns in ant colonies? *Animal Behaviour* 45:1083–1083.

Gordon, D. M. 1996. The organization of work in social insect colonies. *Nature* 380:121–123.

Gordon, D. 1999. *Ants at work: how an insect society is organized*. Free Pr.

Haque, M.; Rahmani, A.; and Egerstedt, M. 2009. A Hybrid, Multi-Agent Model of Foraging Bottlenose Dolphins. In *IFAC Conference on Analysis and Design of Hybrid Systems*.

Hölldobler, B., and Wilson, E. 1990. *The ants*. Belknap Press.

Hybinette, M.; Kraemer, E.; Xiong, Y.; Matthews, G.; and Ahmed, J. 2006. SASSY: A design for a scalable agent-based simulation system using a distributed discrete event infrastructure. In *Proceedings of the 2006 Winter Simulation Conference (WSC-06)*, 926–933.

Luke, S.; Cioffi-Revilla, C.; Panait, L.; Sullivan, K.; and Balan, G. 2005. "MASON": A multiagent simulation environment. *SIMULATION* 81:517–527.

Madden, J.; Arkin, R.; and MacNulty, D. 2010. Multi-robot system based on model of wolf hunting behavior to emulate wolf and elk interactions. In *Robotics and Biomimetics (RO-BIO), 2010 IEEE International Conference on*, 1043–1050.

Matarić, M. 1998. Behavior-based robotics as a tool for synthesis of artificial behavior and analysis of natural behavior. *Trends in cognitive sciences* 2(3):82–86.

Minar, N.; Burkhart, R.; Langton, C.; and Askenazi, M. 1996. The swarm simulation system, a toolkit for building multi-agent simulations.

Nicolis, S.; Theraulaz, G.; and Deneubourg, J. 2005. The effect of aggregates on interaction rate in ant colonies. *Animal behaviour* 69(3):535–540.

Schleidt, W.; Yakalis, G.; Donnelly, M.; and McGarry, J. 1984. A proposal for a standard ethogram, exemplified by an ethogram of the bluebreasted quail (*coturnix chinensis*) 1. *Zeitschrift für Tierpsychologie* 64(3-4):193–220.

Sherer, C.; Vulov, G.; and Hybinette, M. 2011. On-the-fly parallelization of sequential agent-based simulation systems. In *Proceedings of the 2011 Winter Simulation Conference (WSC-11)*.

Webb, B. 2002. Robots in invertebrate neuroscience. *Nature* 417(6886):359–363.