Hindawi Publishing Corporation
Advances in Artificial Intelligence
Volume 2014, Article ID 932485, 23 pages
http://dx.doi.org/10.1155/2014/932485

Research Article

Reinforcement Learning in an Environment Synthetically Augmented with Digital Pheromones

Salvador E. Barbosa and Mikel D. Petty

University of Alabama in Huntsville, 301 Sparkman Drive, Huntsville, AL 35899, USA

Correspondence should be addressed to Salvador E. Barbosa; seb0005@uah.edu

Received 1 October 2013; Revised 19 January 2014; Accepted 31 January 2014; Published 13 March 2014

Academic Editor: Ozlem Uzuner

Reinforcement learning requires information about states, actions, and outcomes as the basis for learning. For many applications, it can be difficult to construct a representative model of the environment, either due to lack of required information or because of that the model's state space may become too large to allow a solution in a reasonable amount of time, using the experience of prior actions. An environment consisting solely of the occurrence or nonoccurrence of specific events attributable to a human actor may appear to lack the necessary structure for the positioning of responding agents in time and space using reinforcement learning. Digital pheromones can be used to synthetically augment such an environment with event sequence information to create a more persistent and measurable imprint on the environment that supports reinforcement learning. We implemented this method and combined it with the ability of agents to learn from actions not taken, a concept known as fictive learning. This approach was tested against the historical sequence of Somali maritime pirate attacks from 2005 to mid-2012, enabling a set of autonomous agents representing naval vessels to successfully respond to an average of 333 of the 899 pirate attacks, outperforming the historical record of 139 successes.

1. Introduction

Sequences of events resulting from the actions of human adversarial actors such as military forces or criminal organizations may appear to have random dynamics in time and space. Finding patterns in such sequences and using those patterns in order to anticipate and respond to the events can be quite challenging. Often, the number of potentially causal factors for such events is very large, making it infeasible to obtain and analyze all relevant information prior to the occurrence of the next event. These difficulties can hinder the planning of responses using conventional computational methods such as multiagent models and machine learning, which typically exploit information available in or about the environment.

A real-world example of such a problem is Somali maritime piracy. Beginning in 2005, the number of attacks attributed to Somali pirates steadily increased. The attacks were carried out on a nearly daily basis during some periods of the year and often took place despite the presence of naval patrol vessels in the area [1]. They were often launched with little warning and at unexpected locations. We would like to use the attributes of past attacks to anticipate and respond to future attacks. However, the set of attack attributes potentially relevant to doing so is quite large; it includes the relative position of patrolling naval forces, the rules of engagement of those patrols, the type of boats and armaments pirates use, the experience level of the pirates, the speed of the targeted ships, and the skill of their captains and crews, the counter-piracy rules of the shipping companies operating the targeted ships, the inclination of those companies to pay ransoms for hijacked ships, the weather and sea state, and many others. Moreover, because the number of ships vulnerable to these attacks are in the tens of thousands annually, the patrolling navies of many countries are not under a unified command but operate independently of one another, the shipping companies are reluctant to reveal the amount of any ransoms paid, and pirate networks are notoriously opaque as to their operations, much information that could be useful to a model of pirate attacks is unavailable. Finally, even if all the information relevant to the pirate attacks was known, the large number of combinations resulting from an even modest

number of options for each of the applicable attributes would likely make it infeasible to exhaustively evaluate all possibilities in a timely manner or to draw inferences from such a large state space in light of a comparatively small sequence of events.

The difficulty of this real-world problem is illustrated by the fact that patrolling navies in the area, equipped with highly sophisticated surveillance systems and staffed with expert military intelligence personnel only managed a timely and successful response to one in six pirate attacks over the period in question [1]. An alternate and appropriate question then is whether there are hidden patterns in the data, such as the frequency and number of attacks in a given area, the timing, and location of an attack relative to those that preceded it, or the penchant of pirates for returning to the general location of a previous attack, that may be detected and exploited using model-free methods to aid in positioning naval assets to defend against future attacks.

Definitions of computational agents differ in detail, but they generally agree that agents are computational entities that perceive and react to their environment with varying levels of autonomy [2, 3]. However, for an agent-based solution to be effective, the environment must provide enough information for the agent to perform its task. There are many levels of agents including deductive, reactive, and hybrid agents [3]. Of these, reactive agents are most reliant on information, as they generally operate solely by reacting to their (usually local) environment. These agents are typically simple and have only a few behaviors, but they may exhibit emergent behaviors, which are behaviors that are not explicitly programmed into individual agents but rather result from the interactions of the agents with each other or with the environment [4].

Along with supervised learning [5] and unsupervised learning [6], reinforcement learning is one of the primary branches of machine learning. Computational agents using reinforcement learning may be found in both model-based and model-free contexts [7]. In this paradigm, an agent interacts with its environment without necessarily having any prior training in it or knowledge of its dynamics. In any given state, the agent chooses an available action and upon executing that action is rewarded based on the effectiveness of the action. From the reward, the agent learns about taking that action in that particular state and possibly in similar states. Any time the agent encounters a previously unseen state, exploration of the environment is taking place. As state-action pairs are encountered, the agent builds a memory of optimal behaviors for future use [7]. Such value iteration agents are deemed by the authors to be in the continuum of reactive agents.

Biologically inspired, multiagent methods have been applied to a range of difficult optimization problems [8]. This approach, often known as swarm intelligence, is modeled on the simple behaviors of individual members (agents) of groups found in nature, such as ants (e.g., ant colony optimization by Dorigo and Stützle [9]) or birds and fish (e.g., particle swarm optimization by Kennedy and Eberhart [10]), which collectively result in emergent properties and features that lead to good solutions. While these techniques do not guarantee convergence to the problem's optimum solution, they have been applied successfully in many areas [8, 11]. Most agents in these approaches fit the definition of reactive agents [3].

The Somali piracy problem may seem a poor fit for a model-based approach, as attempts to implement the model may be hampered by a lack in the required information (pirate tactics or skill of transiting ship captains, for instance). When viewed from a model-free perspective, we would like to learn from the event sequence for this and similar problems, including crime, military operations, and other sequences that, either by nature or intent, may appear unpredictable. However, it could be difficult for an environment with only two states (event and no event) to provide sufficient, distinct, and consistently recurring patterns for the learning algorithm to exploit. We purport that the use of model-free reinforcement learning with reactive agents can be a useful approach to this class of problems, if sufficiently informative states can be constructed from the sequence of the events to position agents in anticipation of upcoming events.

To that end, we introduce a method to generate informative states from an event sequence and to control a set of reactive agents using model-free reinforcement learning, with the goal of positioning agents in proximity of impending events (in essence predicting the next event in time and space). We do this by synthetically augmenting the environment with digital pheromones to indicate both the location of events and areas occupied by agents. These and other information augmenters derived or calculated from the pheromones are used to create discrete signatures in the environment (states) that agents assess and react to (actions). We propagate these markers to a spatiotemporal neighborhood around the event. Regularities in the timing and location of events result in the same state signatures and may be exploited using reinforcement learning by relating them to the signatures of augmenters encountered at the site of past events.

To extend this augmentation concept and improve agent learning speed, we artificially increase the number of learning opportunities by giving agents the ability to learn from fictive or counterfactual actions. This is accomplished by assigning to each agent a "ghost" fictive agent who is colocated with the agent but selects (possibly different) actions using an alternate policy. Actions by the fictive agent that would have resulted in a success, had they been executed by the real agent, are used to update the real agent's learning table, thereby affecting its future action selections.

The overall method intentionally includes both domain-specific and domain-independent elements. The augmenters to be used for the Somali piracy problem are largely specific to the domain (e.g., an augmenter for Shipping lanes will be defined). The piracy augmenters will likely have to be adapted or replaced for other applications, such as criminal acts in an urban area. However, the other elements of the method, including the reinforcement learning algorithm, the use of fictive learning, and the method's central idea of applying reinforcement learning to a synthetically augmented environment are independent of the domain and could be used for other applications once the augmenters for that application have been developed.

This research is intended to address some fundamental questions regarding the feasibility of environment augmentation supporting reinforcement learning agents, the viability of the proposed fictive learning method, and the overall effectiveness of the approach in solving real-world problems. Specifically, we aim to address the following questions.

(1) Can an environment consisting of a sequence of events be synthetically augmented to allow the use of model-free reinforcement learning methods and algorithms in controlling agents?

  (1.1) Do the results of applying different types of information augmenters differ significantly from one another, and if so, which augmenters are most effective?

  (1.2) How does the proposed fictive learning method compare to the standard reinforcement learning approach?

  (1.3) How do different action selection policies compare to one another?

(2) Does the outcome of the proposed methodology compare favorably to the historical record of success or to current means of responding to the real-world events of interest?

This paper is organized as follows. Section 2 briefly introduces the Somali piracy problem and surveys relevant aspects and research in reinforcement learning, multiagent reinforcement learning, fictive learning, and pheromone-inspired computing, the key concepts utilized in this work. Section 3 motivates the need for this novel method and explains the information-augmentation approach by defining the environment of interest and specifying a set of digital pheromones intended to augment that environment so as to enable use of a reinforcement-learning algorithm. Section 4 describes the scenario, formulates the reinforcement learning problem, presents agent behavior and attributes, including the fictive learning ability, and introduces the research platform. Section 5 reports and analyzes the results of applying the proposed approach to positioning naval vessels to defend against Somali pirate attacks, in light of the research questions. Finally, Section 6 states the conclusions, briefly discusses the limitations of the research, and identifies possible avenues for future work.

2. Background and Related Work

In this section brief background introductions are given of the Somali piracy problem, reinforcement learning, multiagent reinforcement learning, fictive learning, and pheromone-inspired computing, and selected relevant research literature is referenced. Research related to maritime piracy is also summarized.

2.1. Somali Piracy. Data compiled and reported by the International Maritime Bureau over a seven and a half year period (2005 through mid-2012) detail nearly one thousand incidents of ships of all types having been fired at, chased, boarded, or hijacked by pirates [12–19]. The attacks generally occurred in the northwest portion of the Indian Ocean and were concentrated in the Gulf of Aden (Figure 1).

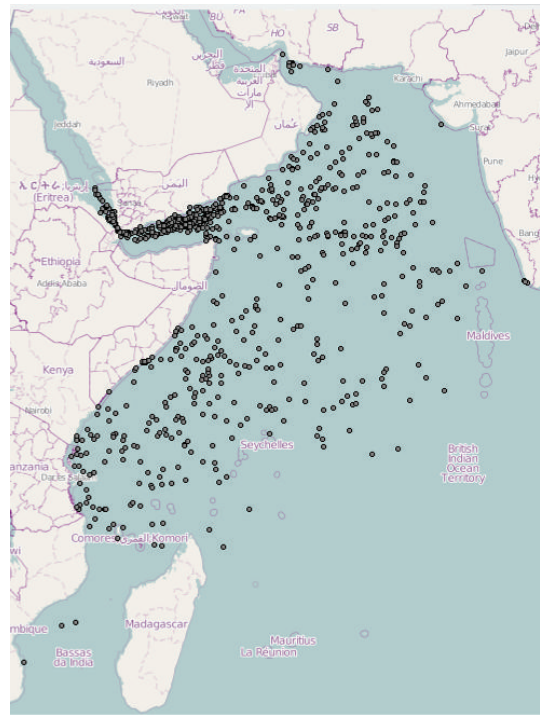


Figure 1: Attacks attributed to Somali pirates (January 2005 through June 2012).

Vessels hijacked by pirates are often ransomed by their owners and insurers for as much as tens of millions of dollars, a value that is usually a small percentage of the value of the vessel and its cargo. As a result, Somali piracy is a very profitable business with the pirates as the most visible aspect of a network that involves rich and powerful people at the top, an entire cast of middle men and enablers, and many others who profit from supplying the pirates, feeding or guarding hostages, or acting as lookouts [1, 20].

In light of the increasing multimillion dollar ransoms, insurance and shipping companies are willing to pay to get ships and crews back [21], the demand for ransom payouts likely continues to be high in the foreseeable future.

Since the hijacking of the Ukrainian ship MV *Faina* in September 2008, which was carrying a shipment of lethal military equipment to southern Sudan [22], three naval task forces purposely established to deal with the piracy problem have been patrolling the waters around Somalia and the Gulf of Aden, aided by vessels from other navies outside of their command and control structure [23]. The approximately 25 to 40 naval vessels available at any point in time [1] face the daunting task of stopping attacks in an area of over 6.5 million square kilometers (2.5 million square miles) [24]. The bulk of naval counter piracy efforts are centered on the Gulf of Aden.

In the early years of Somali piracy, a group of four to eight pirates trolled the waters in small boats known as skiffs,

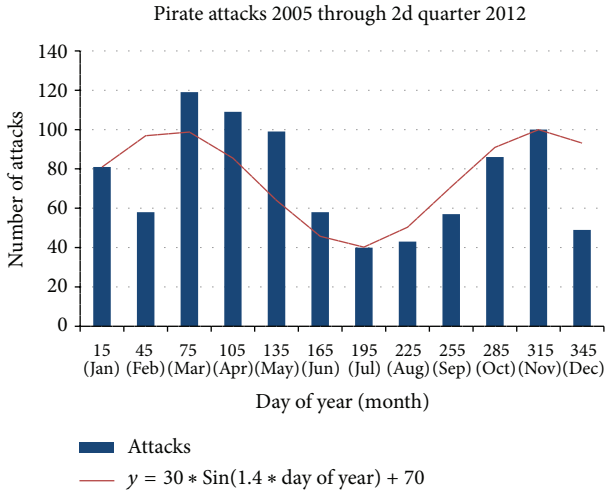


FIGURE 2: Pirate attacks by month appear to show a cyclical pattern.

looking for victim ships. As their success grew, pirates used commandeered medium-sized vessels to tow multiple skiffs farther into the ocean, allowing them to stay at sea longer and to conduct multiple attacks. These larger vessels became known as mother ships and their increased use is believed to be a significant enabler in the higher number of attacks seen in recent years [22]. While accurate counts of the number of ships transiting the entire area vulnerable to piracy are not available, more than 20,000 vessels are estimated to have passed through the Gulf of Aden in 2009 [20, 25].

The concentration in geography of pirate attacks is likely due to both the availability of victim shipping vessels, as well as the relatively narrow gulf waterway (when compared to the wide open ocean). However, an analysis of the number of attacks also revealed a pattern of approximately 8.5 months in duration as shown in Figure 2.

2.2. Reinforcement Learning. Reinforcement learning is a powerful computational paradigm employing well-understood algorithms [26] that may be used in a model-free context. Its applicability has been stretched to a vast class of problems and to multiple agents, through various modifications or restrictions. In its basic form, reinforcement learning involves a single agent, with all other actions (including those of an adversary) emanating from a stationary environment. A working definition of a stationary environment is that the true value of actions does not change over time [7].

As mentioned earlier, reinforcement learning is reward-based. At any time t, the agent seeks to maximize the total reward it will receive through some future time T. When T is finite, the task being performed is said to be *episodic*, with T denoting the end of an episode. When T is infinite, the task is said to be *continuing* [7].

Thus, the projected total reward at time t, denoted R_t, is defined as

$$R_t = r_{t+1} + \gamma r_{t+2} + \gamma^2 r_{t+3} + \cdots = \sum_{k=0}^{T} \gamma^k r_{t+k+1}, \quad (1)$$

where all r_i are the rewards received in successive time steps. The discount rate γ, with $0 \leq \gamma \leq 1$, serves to keep the result finite when T is infinite and to enable the calculation of the present value of future rewards.

To obtain a reward, the agent assesses its state and selects an action available in that state. The agent is then atomically rewarded (that term also applying to no reward or to negative rewards, which are understood as punishments) and transitioned to the next state. The selection of an action in any given state may be deterministic or probability based. This mapping from states to actions is known as the agent's policy and is normally denoted π. The goal of reinforcement learning is the determination of an action policy to maximize total reward.

A central focus of reinforcement learning algorithms is the estimation of value functions. The value function is a measure of the worth of being in a particular state or of taking an action while in a given state, while observing a specific policy. The two primary value functions used are $V^\pi(s)$, the state-value function, and $Q^\pi(s, a)$, the action value function [7]. These functions are defined as follows, where s is the state, a is the action, and $E_\pi\{\}$ is the expected value while following policy π:

$$\begin{aligned} V^\pi(s) &= E_\pi\{R_t \mid s_t = s\} = E_\pi\left\{\sum_{k=0}^{\infty} \gamma^k r_{t+k+1} \mid s_t = s\right\} \\ Q^\pi(s, a) &= E_\pi\{R_t \mid s_t = s, a_t = a\} \\ &= E_\pi\left\{\sum_{k=0}^{\infty} \gamma^k r_{t+k+1} \mid s_t = s, a_t = a\right\}. \end{aligned} \quad (2)$$

Temporal differences are the driving force that propel and enable online learning. Whereas Monte Carlo methods used in reinforcement learning are predicated on averaging all returns seen over an episode or trial, temporal difference learning updates knowledge at each time step, as experience is gained during an episode or trial [7]. The simplest form of temporal difference methods, TD(0), may be used in model-free contexts and is defined for the state-value function and action-value function, respectively, as

$$\begin{aligned} V(s_t) &\longleftarrow V(s_t) + \alpha\left[r_{t+1} + \gamma V(s_{t+1}) - V(s_t)\right], \\ Q(s_t, a_t) &\longleftarrow Q(s_t, a_t) \\ &\quad + \alpha\left[r_{t+1} + \gamma Q(s_{t+1}, a_{t+1}) - Q(s_t, a_t)\right]. \end{aligned} \quad (3)$$

In both cases, the value estimate for time t is updated by being summed with a scaling of the reward received upon transition to time $t + 1$ (as a result of the state and action at time t) plus the difference between the discounted value estimate for the state and action at time $t + 1$ and the original estimate for the value at time t. The scaling parameter α is known as the learning rate and determines how much of the difference should be applied at any given time step.

In this research we used the temporal difference algorithm using state values shown in Algorithm 1 with the ε-greedy policy (detailed in a later section). Note that the Q-Learning algorithm [27] is a temporal difference method that

initialize all $V(s)$ arbitrarily

for all episodes
  initialize s
  repeat
    choose the next state s' using policy
    observe r
    $V(s) \leftarrow V(s) + \alpha\left[r + \gamma V\left(s'\right) - V\left(s\right)\right]$
    $s \leftarrow s'$
  until s is terminal state

where $V(s)$ is the value of being in state s and $V(s')$ is the value estimate of the resultant state s', γ is the discount rate, and α is the learning rate.

Algorithm 1: Temporal difference learning algorithm (adapted from [7]).

learns the value of state-action pairs (each possible action in every state has a value and leads to an end state). Thus, when the agent makes a choice for its next move, it is guided by the value stored for the state-action pair. As an example, an agent on a *diving board* (the state) may have 3 actions available to it: (1) *jump into the pool*, (2) *jump onto the ground*, or (3) *climb down the ladder onto the ground*. The action is selected based on the value stored for each of the state-action pairs. Thus, while the latter two actions both result in a transition to the state *ground*, the climbing down option may have greater value than jumping down. In this research the state-value is used, meaning we are concerned with the utility, or worth, of being in a particular state and not the means by which we reach that state. Thus, parallel to the previously provided example, the agent is faced with two reachable end states *pool* and *ground* and selects its end state from the best of the two values (there are no explicit actions, only reachability). Therefore, references to the term "action" throughout this paper should be construed as the selection of a state transition available to the agent at any point in time, and not in the state-action pair sense.

In this work, two issues with conventional reinforcement learning must be acknowledged. First, reinforcement learning algorithms ordinarily assume that the environment is stationary (as defined earlier); if it is not, the agent must potentially keep relearning states and actions [28]. Research on the nonstationarity problem has shown that reinforcement learning can still be an effective tool even in such cases [7, 28, 29]. Second, reinforcement learning can be compromised in a multiagent environment. When multiple agents operate within the same environment, they exacerbate the nonstationarity problem as their actions affect other agents directly or indirectly via the environment, and they are in turn affected by the actions of others.

2.3. Multiagent Reinforcement Learning. In [3], an *agent* is defined as a computer system located in some environment and capable of autonomous action toward an objective. Agents have also been described as perceiving their environment through sensors and acting upon it [2]. A multiagent system is one where multiple agents act in response to their environment.

Because reinforcement learning is predicated on a single agent's interactions with the environment, having multiple agents presents three difficulties to its straightforward use: (1) it aggravates the lack of stationarity, as the actions of other agents alter the environment in an uncontrollable way, when viewed from the perspective of any single agent; (2) it raises the problem of how to distribute rewards, and (3) it introduces a possible requirement for some level of coordination and communication among the agents. Prior work overcomes some difficulties of multiagent reinforcement learning [30–32].

The agents in this work are simple, loosely coupled, homogeneous, and independent (in that there is no direct cooperation required in tasks) and utilize minimal coordination, which is achieved indirectly through digital pheromones. While they are multiple agents within the same environment and their behaviors impact those of other agents (sharing of rewards, for instance), agents have no awareness of one another. Given the agent simplicity and independence, the single agent approach to reinforcement learning is employed in this research.

2.4. Fictive Learning. Reinforcement learning enables learning through rewarding particular actions chosen by an agent in a given state. In general, the agent only learns from the actions it chooses and does not have the ability to learn from "what might have been" if it had selected a different action. However, some reinforcement learning research has focused on fictive learning (also referred to as counterfactual learning or learning through actions not taken). This concept is closely related to the notion of regret when making decisions under uncertainty. Fictive learning is usually adopted to speed up the learning process by either allowing the agent to learn from alternate actions it could have taken or from the actions of others [33]. We review some work on this alternate form of learning in this section.

A neuroscience imaging study of the human brain during decision making explored choice making in games with rewards [34]. The work reviews human behavior as subjects

are presented with various situations to identify areas of the brain associated with reward expectation. The model employed used standard Q-Learning [27] expanded by two factors: a component representing the outcomes of actions of others in comparable states and a factor indicating what should have been the best possible action in the situation.

Another Magnetic Resonance Imaging study was conducted on subjects making financial investment decisions [35]. In that study, subjects invested a sum of money and were then presented with the market results (gain or loss of the investment made). The fictive learning signal output was the difference between the best outcome that could have been possible and the actual outcome. The task was repeated multiple times to image areas of the brain as the subjects' decision making were affected by the gains and losses.

Another form of fictive learning is referred to as difference rewards and is intended for use in multiagent environments. The difference reward is defined as a system level reward that evaluates the performance of all agents collectively and subtracts from that the performance attained with all agents except one. This difference, calculated separately for each agent by excluding that agent's performance, in effect assesses the contribution of each agent to the overall team performance. Two application studies utilizing this method are an aviation traffic flow manager [36] and a variation of the El Farol bar congestion problem [37].

2.5. Digital Pheromones. Digital pheromones are a computational artifact patterned after the chemical substances deposited in the environment for communication by many social insects, such as ants [38]. Since the introduction of the ant colony optimization heuristic [9], the concept of digital pheromones has found a number of uses beyond its originally intended use for finding shortest paths. Those applications are quite diverse, including job scheduling [39], weapon-target assignment [40], and assembly line balancing [41]. In this section, some of that work that closely parallels our approach is reviewed.

In [42], pheromones are used in a multiplanar cooperative search application. The area of interest is subdivided for search by multiple unmanned vehicles in different media (air, sea, and land) and the goal is to search the entire area at the same rate. Since the vehicles have different speeds and sensor apertures, and the search area of a large sensor may overlap those of several smaller sensors, pheromones are used to mark locations as searching takes place. Agents are then able to calculate where their search services can best contribute to the global goal. Their technique relies on more than stigmergic signals for coordination in that visited grid cells and timestamps are transmitted to neighboring agents.

A patrolling problem of minimizing the duration between successive visits to subareas within a larger area requiring coverage is posed in [43]. In that work, the *Pants* algorithm (for probabilistic ants) is developed to use pheromones dropped by prior patrolling entities to calculate the best area to be visited by a given agent. The pheromone content is used to compute potential fields that "pull" agents toward areas of low pheromone concentrations. They use local neighborhoods to restrict both the state information flowing to agents and the decision making.

The problem differs slightly in [44], where the goal of minimal delay between patrol visits is applied to an area that may not be fully known. Their approach uses the dispersion of pheromone to neighboring, but possibly not previously visited grid cells. Agents descend to areas of lower pheromone and are restricted to moving to neighboring grid cells. Their technique enforces the single agent per grid cell policy to improve coverage.

Another use of pheromones, reported by Sauter et al. [45], is geared toward patrolling by unmanned vehicles. They use four types of pheromones, including one that identifies areas needing to be visited at a given frequency and one to mark areas that have been visited recently. The *Lawn* pheromone is emitted from sites requiring revisit at specified rates. These pheromones are considered to have been "cut" when the required visit is made by an agent and they begin "growing" again after some time. The *Visited* pheromone is dropped by agent and has the effect of repelling other agents, while its level is above a set value in order to avoid duplication of effort.

Monekosso and Remagnino use synthetic pheromones for multiagent reinforcement learning in their *Phe-Q* Learning algorithm [46]. Their basic premise relies on agent communication through the dispersal and diffusion of pheromones as many other approaches do. In that work, a belief factor is added to reflect the trust an agent has in pheromones deposited by other agents. This factor is controlled by the number of tasks that the agents successfully complete (rewards).

In a crime simulation reported by Furtado et al. [47], pheromones are used by agents representing criminals. That study exploits the preferential attachment mechanism associated with the predilection criminals have for committing crimes in places with which they are familiar. The pheromones placed in the environment by criminal agents are followed by other agents within their networks, based on communications patterns and criminals' experience level. That model also restricts agent sensing and decision making to a limited neighborhood.

2.6. Maritime Piracy Studies. A multiagent model of Somali piracy is documented in a sequence of studies [48–50]. That model focuses primarily on the Gulf of Aden, the geographic area within which most pirate attacks have taken place, and does not attempt to model the entire region where attacks have been experienced. The investigation is centered on game-theoretic routing of shipping vessels, pirates, and patrolling navies. Synthetic shipping traffic is generated and routes automatically planned for the cargo vessels. Risk maps, based on historical attack sites, are used to position defensive vessels to respond to pirate attacks. The pirate model is described as using "simple reinforcement learning" to avoid the defensive vessels, but details are not provided.

The Piracy Attack Risk Surface model is the subject of a thesis from the US Naval Postgraduate School [51]. The model predicts likely piracy attacks in the coming 72-hour period. It is implemented as a Monte Carlo simulation that factors in

weather and sea state, prior attacks, and military intelligence factors to determine the locations at most risk.

In a maritime counter-piracy study [52], an agent-based model is used to explore defense against pirate attacks and vulnerabilities of vessels during attack. The simulation is a tactical one that pits a single large commercial ship against a fast pirate attack boat. The analysis employs the MANA agent model from the New Zealand Defense Technology Agency.

Another tactical simulation of an attack on a high-value commercial ship was done by Walton et al. [53]. That work models an attack by a small suicide boat against the larger vessel, which is protected by armed sea marshals.

3. Information-Augmentation for Reinforcement Learning

This section motivates and explains the technical approach used in this work by defining the environments of applicability and by specifying a set of digital pheromones intended to augment that environment so as to support a reinforcement-learning algorithm.

The class of problems of interest consists of a sequence of events, caused or generated by a human actor, which happen in relative proximity of one another in both time and space, and for which a model-based approach may be practically intractable (with respect to timeliness) or incomplete (due to information requirements). A team of agents has an interest in being favorably positioned within a specified proximity of the events in order to respond to them.

Our approach is to create artificial patterns in the environment, directly related to the events in question, which may be exploited through use of reinforcement-learning algorithms. We employ reactive agents that use reinforcement learning to evaluate the state of the augmented environment and to select actions that would place them within a given proximity of an event. An envisioned application of this research is a real-time online controller or recommender system for positioning a set of agents tasked with being in proximity of the events of interest, using learning from prior events.

3.1. Motivation. Reinforcement learning is founded upon the concept of a Markovian state, which is to say that the state contains sufficient information and history to enable selection of the best action. Thus, reinforcement learning problems are often defined as Markov Decision Processes [7]. However, for most interesting and real-world problems, the Markovian property does not hold and there is insufficient information to select the best action. An additional complication is that the true value of actions tends to change over time, making the environment nonstationary.

When the Markovian state signal is not available, the environment is deemed to be only partially observable. This problem is then said to be a Partially Observable Markov Decision Process (POMDP). In a POMDP, the signals perceived from the environment are interpreted as observations that are mapped to true (but not fully observable) states via a probability distribution. There are many techniques used to solve POMDPs, including heuristic-based techniques, value iteration approaches, and approximation methods [54]. However, nearly all of these methods are model-based techniques that require *a priori* knowledge of the observations to state mapping functions, the state transition functions, and the rewards functions. Since these models are generally not available for real-world problems, an alternate option for obtaining them is to attempt to recover or to derive an approximation of them from observations. These methods have been deemed prohibitive in literature, due to time and difficulty [54].

Another approach available for handling a POMDP in a model-free manner is to treat observations as states, and to map those directly to actions [54]. It is that method we employ in this research, with the understanding that doing so does not directly address partial observability or nonstationarity.

3.2. Information Augmenters. Three categories of augmenters are defined for the augmentation process. A primary augmenter is a scalar quantity with maximum value at the time and location of an event and declining value over time and distance from the event. A map probability augmenter is also used and describes additional domain information pertaining to the geography of events. In this research, this augmenter does not change over time, although nothing precludes it from having a dynamic value to support multiple epochs. Finally, a set of secondary augmenters, which are derived from computations using one or more of the primary and map probability augmenters, are also used.

This research experimented with a total of seven information augmenters. The three primary augmenters are based on digital pheromones, the map probability augmenter represents geographically related information, and three secondary augmenters are calculated from one or more of the primary augmenters and/or the map probability. At this stage of the research only one of the augmenters, selected *a priori*, is used during a single simulation execution and no dynamic augmenter selection mechanism is in place. However, the *Weighted* augmenter described below enables a combination of multiple augmenters into a single measure that considers multiple facets at once. The details of the augmenters follow.

The primary augmenters are as follows.

(1) The *Event* pheromone is deposited at the occurrence of each event and is spread over a parameter-controlled radius of grid cells around the event's location. Its effect evaporates/decays at a medium rate over time, as optimized for the application domain. In the context of Somali piracy, an event is a pirate attack on a vessel. The rationale for this augmenter is to identify areas of previous events of interest, as both a region amenable to such events and one that potentially appeals to aspects of human psychology, such as the penchant of criminals to return to locations where they have successfully committed crimes [55]. The evaporation rate (herein both "evaporation" and "decay" are used interchangeably refering to the reduction of a pheromone's value over time) is set to

a medium level because it may be some time before a revisit to the vicinity of a previous event takes place.

(2) The *Cyclical* pheromone is released over a given radius of grid cells around the site of each event to capture any known cyclical aspects or frequency of the events' occurrence over time. The parallel to nature's pheromones are suspended with respect to this augmenter to allow both its decay and intensification over time. This repeating phenomenon is modeled using a sinusoidal wave with a given period in days and a linearly decreasing amplitude over time. Complete evaporation of this pheromone may be delayed for quite some time, in which case it serves as a long term memory of the events in an area. In the context of the Somali piracy application, this pheromone represents the seasonal pattern described earlier.

(3) The *Occupied* pheromone is deposited by the agents in the grid cells in which they are positioned and is dispersed to the nearby vicinity. This pheromone is designed to dissuade an agent from moving to a grid it has recently occupied. It has the added effect of improving map coverage by reducing "bunching up" near likely event areas, even though agents are unaware that there are other agents in the environment who also deposit this repelling pheromone. It is set to a high evaporation rate and is a computational aid for agent positioning that does not have a direct historical parallel in the Somali piracy domain.

The *Map Probability* augmenter is used to communicate domain information regarding the likelihood of the events of interest taking place at the various locations in the environment. It takes on a value between 0 and 1 for each grid cell represented. In the Somali piracy application, this value represents the shipping traffic levels in the region of interest, and reflects the fact that attacks are more probable in areas where more potential targets are present.

The secondary augmenters are as follows.

(1) The *Event Count* augmenter is a count of past events in a grid cell. It is linked to the *Cyclical* pheromone (which is deposited along with every *Event* pheromone but has a slower rate of decay) and is simply a count of the instances of that pheromone affecting previously attacked grid cells at any given time.

(2) The *Ratio* augmenter is the ratio of the value of the *Event* pheromone to that of the *Occupied* pheromone. This augmenter serves to indicate when a given grid cell may need to be visited, if it is high, or whether a visit may be postponed, if its value is low.

(3) The *Weighted Combination* augmenter is a weighted sum of any of the other augmenters, with weights summing to 1. This allows the agent to react to a single quantity that represents multiple facets of the environment.

3.3. Resultant Augmented Environment. The augmentation or transformation creates a synthetic "memory" in the environment, associated with each event, which persists over some time and distance. The values of the augmenters over time are sensed by an agent as the state and enable action selection using reinforcement learning. A contrast, with respect to time, between a nonaugmented environment and one that has been augmented is shown in Figure 3. The upper time line depicts an environment characterized solely by event occurrences, with little persistent and measurable information between events. The augmented environment "fills in" the gaps between events with changing levels of the pheromones and other augmenters, thereby providing dynamic and measurable quantities that persist over time and space and serve as synthetic states to which agents respond.

The spatial impact of an augmented environment is shown in Figure 4. Three events are shown (row, column coordinates) in grid cells (3, 7), (5, 3), and (7, 7) with the latter having taken place in an earlier time step. The nonaugmented environment presents a static snapshot in time and provides little information to help position agents. By contrast, the augmented environment presents a number of facts not discernible in the first case: dynamics that may be compared to those of prior events to exploit patterns, the fact that grid cells (3, 5) and (4, 5) have pheromone content that are equal to those at attack sites and thus may become candidates for future moves, and an indication that the highest pheromone content is cell (5, 5), at the intersection of the five cell by five cell pheromone fields of the three events.

4. Reinforcement Learning Problem Formulation

The information augmentation process was carried out for the Somali piracy historical sequence of attacks to demonstrate an application of the proposed method. We note here that the Somali piracy domain is challenging in several ways: the true values of state and actions do change over time, the proposed approach utilizes a limited sensor range and does not provide complete information, and states may be aliased because pheromone levels at a particular point are the same for an attack at a specific range but from varying directions or may be composed of deposits stretching across time and space that happen to combine to a specific value. As mentioned earlier, observations are treated as states in this application and agents react directly to them.

The experiments were conducted on a custom research platform with a discretized grid representation of the pirate attack area, the historical sequence of pirate attacks over a seven and one half year period, and a set of agents representing naval vessels. The simulation proceeds in a time-stepped mode, with each time step corresponding to one actual day in the historical account. No time steps are skipped because agents act at each time increment whether or not an attack occurs. The reinforcement learning problem formulation follows.

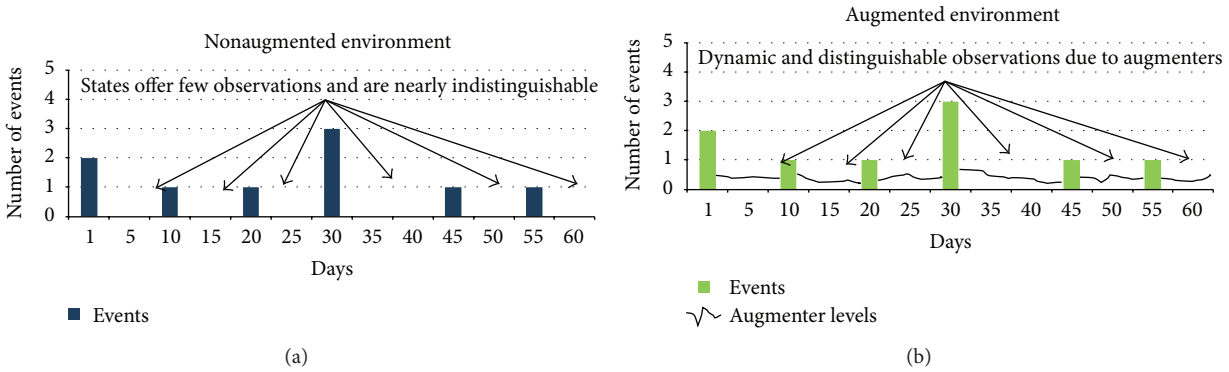


Figure 3: Contrast between nonaugmented and augmented environments (temporal).

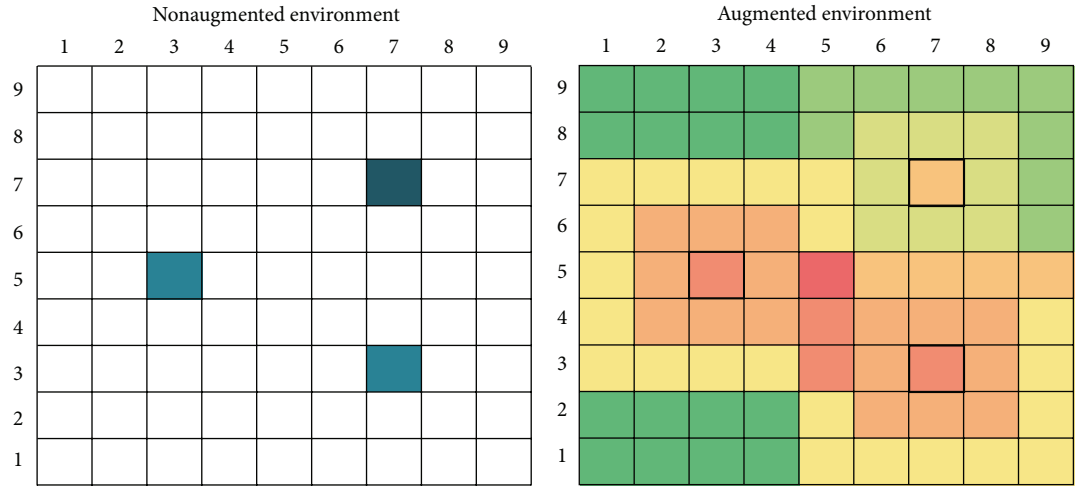


Figure 4: Contrast between nonaugmented and augmented environments (spatial).

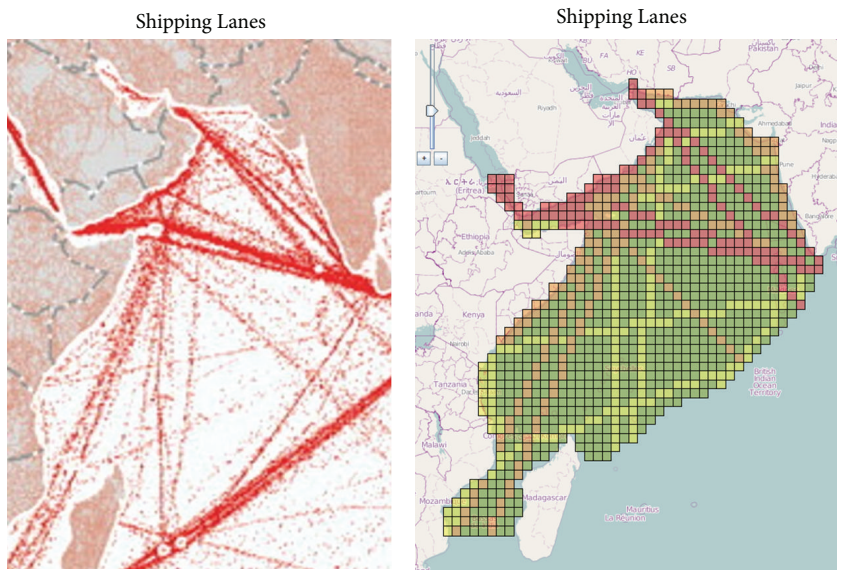


Figure 5: Shipping lanes in the piracy region (adapted from [56]).

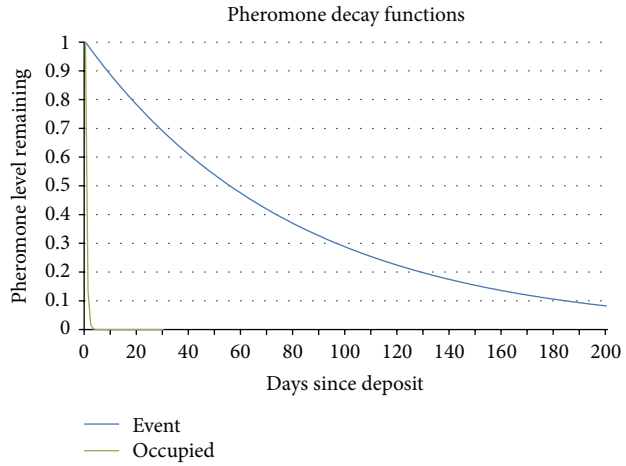


Figure 6: Pheromone exponential decay functions.

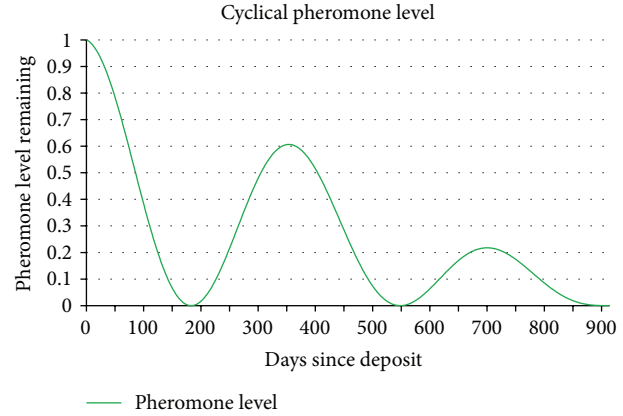


Figure 7: Decay function for the *Cyclical* pheromone.

4.1. Environment. The area of attacks was discretized into one degree of latitude by one degree of longitude grid cells and is represented by one thousand such grids, each of which has a preset *Map Probability* augmenter to represent one of four shipping traffic levels in the region: low, medium, high, and very high traffic (see Figure 5). The grid resolution was chosen to roughly correspond to the action radius of agents (described later).

4.2. Events. The pirate attacks in the region are the events of interest. The International Maritime Bureau (IMB) reports contain a mixture of statistics that are updated regularly, along with regional subsections that contain facts on the date, time, location, and victim vessel information for each attack, as well as a descriptive narrative that recounts how the attack unfolded and includes other details such as whether the military successfully intervened. Attacks are classified into four categories: *Hijacked*, *Boarded*, *Fired Upon*, or *Attempted*. Because the naval vessel agents' goal is to deter all attacks, we make no distinction between attack categories and consider attacks from any category to be an event of equal interest.

The research dataset was assembled from IMB annual reports for 2005 through 2011 and quarterly reports for the first six months of 2012 [12–19]. Table 1 shows an excerpt of an attack from an IMB report and its format in the research database. The IMB reports for the period January 2005 through June 2012 inclusive list 943 attacks under the sections for Somalia and Gulf of Aden. A number of these records are missing information, such as the attack's latitude and longitude or its time of day. In order to retain the highest possible number of historical events for this research, all of the attacks in the IMB reports were used except the 44 records missing the geographical coordinates of the attack. The data deemed relevant for the analysis was extracted, and the narrative portion was keyword searched to glean additional data regarding military intervention and use of pirate mother ships. This process resulted in a dataset of 899 attacks (events) over the 2,738-day period (1 January 2005 through 30 June 2012).

4.3. States. The information augmenters are each represented by a floating point value in each grid cell, along with globally applicable functions that determine changes in their value with respect to time and distance. The levels are calculated at each time step by applying the functions to the list of augmenters affecting the grid cell. The *Event* and *Occupied* pheromones each use a standard exponential decay function (e^{-kt} where k is each pheromone type's individual evaporation constant, and t is the number of time steps since its deposit). The *Event* pheromone is applied over a radius of four grid cells surrounding the attack site (approximately 240 nautical miles), and the *Occupied* pheromone is posted over a radius of one grid cell around the last position of the each agent (approximately 60 nautical miles). These pheromones are removed from affected grid cells once their value drops below a given threshold (0.1 in the research configuration). Figure 6 shows the pheromone decay functions.

The level of the *Cyclical* pheromone is computed using a sinusoidal wave with a gradually decreasing amplitude. The daily level calculation uses an annual cycle period and MAXD days to complete evaporation. The level is calculated as

$$0.5 \times \left[1 - \frac{t}{\text{MAXD}} \times \cos\left(t \times \text{YEARS PER DAY} \times 2\pi\right) + 1 - \frac{t}{\text{MAXD}}\right], \quad (4)$$

where t is the number of days elapsed since the pheromone was deposited, and YEARS PER DAY is 1/365 (a base period other than annual may be chosen and the parameters adjusted accordingly). Coincident with the *Event* pheromone, the *Cyclical* pheromone is spread over a radius of four grid cells surrounding the attack site. This pheromone yields a memory of MAXD = 913 days (two and one half years). The resulting sinusoidal wave is shown in Figure 7.

A single distance-effect function is used for all pheromone types to establish the pheromone's level in grid cells. It is a Gaussian function with a parameterized width that is centered at the location of the event. The result of the distance function is combined with that of the time decay functions previously described to determine

Table 1: Data conversion from IMB report to research database format.

Date Time	Name of ship/type /flag/Grt/IMO #	Position	Narration
02.02.2011 0830 UTC Steaming Fired upon	Duqm Tanker Panama 160160 9410387	20:16N—063:36E (around 225 NM ESE of Ras al Hadd, Oman), off Somalia	About eight pirates in two skiffs armed with RPG and automatic weapons chased and fired upon the tanker underway. The tanker raised alarm, increased speed, and contacted warship for assistance. The two skiffs kept firing with automatic weapons. Warship arrived at location and the skiffs stopped chasing and moved away. A helicopter from the warship arrived at location and circled the tanker. The helicopter contacted the pirates by VHF radio and ordered them to surrender their weapons. Pirates replied that they would kill the Iraqi and Pakistani hostages held onboard the mother ship if the warships attacked the skiffs.

Excerpt from IMB report [18]

Days since[1]	Lat D	Lat M	H[2]	Lon D	Lon M	H[2]	Mother ship[3]	Military aid[3]
2224	20	16	N	63	36	E	Y	Y

Resultant database entry

Notes:
Most columns are self-explanatory.
[1]This field was created for the simulation and indicates the number of days since December 31, 2004 (thus, simulation day 1 is January 1, 2005).
[2]These fields show the attack's north-south and east-west hemispheres, respectively.
[3]These fields represent whether a mother ship or a military intervention played a role in the attack; this information was gleaned from the free text narrative describing the attack (Y: yes and N: no).

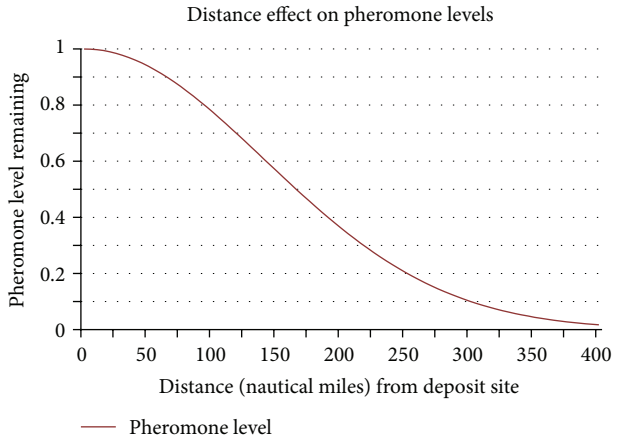


Figure 8: Effect of distance on pheromone levels.

the pheromone level at each affected grid cell. Figure 8 depicts the distance-effect function utilized in the research platform. The values of the evaporation constant k, chosen for the pheromones after some experimentation, yield a persistence of 180 days and 2 days for the *Event* and *Occupied* pheromones, respectively, at the deposit site (and fewer days in surrounding grid cells due to the distance decay).

The state of each grid cell in the simulated world is computed at each time step from the values of the augmenter chosen *a priori* for action selection during simulation execution, which we term the operational augmenter, and represents the following:

(1) The augmenter level discretized to one of four absolute levels.

(2) Whether the augmenter value has increased or decreased since the previous time step.

(3) Whether the augmenter value is the maximum or minimum within the agent's neighborhood.

The agents perceive and react to these states. Using the above descriptions, there are nine possible combinations for each of the four discretized augmenter levels, for a total of the 36 augmenter state signatures, as shown in Table 2.

4.4. Agents, Fictive Agents, and Actions. Agents are mobile entities that perceive the state of the operational augmenter of individual grids in a restricted local area referred to as its neighborhood, react to those states, and learn individually as they receive rewards. An agent is unable to sense outside its neighborhood and exploits the environment based solely on local information. At each time step, the agent selects as its action a transition to a grid cell containing the state signature with the highest state value, from the signatures present in the grid cells of its neighborhood of radius r. As such, it may choose an action that places it in any of up to $(2r + 1)^2$ grid cells in its neighborhood (a radius of 2 was used in

Table 2: Grid cell state signatures.

Operational augmenter characteristics
Level X, not minimum or maximum, not increasing or decreasing
Level X, not minimum or maximum, decreasing
Level X, minimum, not increasing or decreasing
Level X, minimum, decreasing
Level X, maximum, not increasing or decreasing
Level X, maximum, decreasing
Level X, not minimum or maximum, increasing
Level X, minimum, increasing
Level X, maximum, increasing

this research, representing a naval vessel's approximate range when traveling at 20 knots for eight to ten hours during a 24-hour period and resulting in a neighborhood size of up to 25 grid cells). The geographical layout of the environment may at times lead to neighborhoods of size less than $(2r + 1)^2$, such as when an agent is located adjacent to non-navigable grid cells (representing land in the Somali piracy case). If no distinct maximum state value exists, the agent applies a heuristic that biases its action to the highest operational augmenter level present in the neighborhood. We refer to this heuristic as the *Level-bias*.

Regardless of the operational augmenter used, when multiple cells in the neighborhood have the chosen signature, the grid with the highest concentration of the *Event* pheromone is preferred as the destination. This technique is termed the *Event-Bias* and is a key attribute of agent behavior, as it has the effect of keeping agents in, or drawing them to, areas with high event concentration. If all *Event* pheromone values are equal across the neighborhood, a grid is randomly chosen.

The action set used in our approach (a move to a desired end state, which is to say, a grid cell with a particular operational augmenter signature) is a domain-independent abstraction with no direct equivalent in the realm represented, since domains are unlikely to have actions like "Move to a grid cell where the operational augmenter is a Level 2, Not Minimum or Maximum, Decreasing." When reinforcement learning is combined with this method, the end result is selection of grid cells that have similar augmenter state signatures to those that have received the highest accumulated rewards, thereby exploiting patterns, if they exist.

In order to prevent multiple agents from clustering in the same grid cell, an auction system based on the state value of each agent's selected action decides the order in which agents choose their destination grid cell, the agent with the highest state value having first choice. Once a grid cell is selected by an agent, it is not available as a destination option to other agents, and agents vying for that cell must reenter the auction. This restriction is enforced to improve agent coverage of the geographical area and is removed only when, after repeated auctions, an agent has only one possible destination remaining, in which case a move to that grid cell is permitted regardless of whether the cell is occupied. The behavior of these simple, reactive agents may be summarized by the commented algorithm shown in Algorithm 2.

To enable fictive learning, each of the real agents is paired with a fictive agent. The role of the fictive agent is to increase the experience of the real agent, by helping to populate the learning table more quickly. This fictive agent "ghost" begins each time step at the same location as its parent agent but chooses its action based on a separate policy. In this research, the alternate policy applied was the *Level-bias* technique, which biases the action to the highest operational augmenter level found in the neighborhood. Additionally, unlike the real agent, the fictive agent destination grid cells are not restricted to those that are not occupied. Finally, the fictive agent used in our methodology is a nonlearning agent that, in each time step, simply executes the best action available per the *Level-bias* heuristic. The fictive learning approach used in this research is novel and unlike others encountered in the literature. The behavior algorithm for fictive agents is shown in Algorithm 3.

The agents represent naval vessels seeking to be in position to deter, thwart, or mitigate the effects of an attack by responding within 30 minutes of a hijack in progress call. The 30-minute response is a stated goal of coalition naval forces [25–57]. The calculation of the effective ranges stems from parameters stated in those references, such as helicopter launch times and speeds, and is set at 36 nautical miles (67 kilometers) in the simulation. Each vessel agent has an action radius of two grid cells in any direction. This action range is the domain equivalent of 8 to 10 hours of a vessel moving at 20 knots during a 24-hour patrol period.

The individual agents do not correspond to specific actual naval vessels in the historic account, as the daily position of those vessels is not available. The agents are placed in random locations initially and move over time as they react to the augmented environment. The position of agents is coherent over time and is computed from the prior location and the size of the neighborhood. The aim of this study was to develop a methodology that might be of use in positioning such assets continually over time.

Data on the exact number of naval vessels patrolling the piracy region over time is not available. We have previously cited the estimate of 25 to 40 vessels mentioned in [1]. However, this may only have been the case in the latter period represented by the data, with additional ships being assigned to the area as the piracy problem grew. Because a precise historical number of naval vessels is not available, we conservatively used the lower number found in the literature (25 ships) as the agent count. Three different schemes for reaching that agent count were employed as follows.

(1) Initialize with 25 agents and maintain at 25 agents over the simulation.

(2) Initialize with 5 agents and increase to 25 agents linearly over time, with 1 agent added every 130 days in a random geographical location. This method results in a fleet that averages 15.03 agents over the simulation.

(3) Initialize with 5 agents and increase to 25 agents linearly over the number of attacks (events), with 1 agent added every 42 attacks in a random geographical

```
Algorithm: agentAct
Input: agent location, loc
Returns: destination grid cell, g

S ← get States In Neighborhood(loc)   // set of augmenter states in agent's neighborhood
s ← max(S)                            // augmenter signature with highest learned value
if s == null                          // no distinct maximum state value exists
   s = Level-Bias(S)                  // augmenter signature with highest augmenter level

G ← get Grid Cells(s)                 // set of all neighborhood grid cells having state s
g ← Event-Bias(G)                     // grid cell g from G with highest Event pheromone level
if g == null                          // all Event pheromone levels are equal
   g ← randomGrid(G)                  // choose g at random from G

ok ← auction(value(s), g)             // enter auction with state s value and destination grid g
                                      // auction returns true if the grid cell is unclaimed or if the
                                      // grid cell g is the last destination option for the agent
if not(ok) agent Act(loc)
return (g)
```

Algorithm 2: Agent action selection algorithm.

```
Algorithm: fictiveAgentAct
Input: agent location, loc
Returns: destination grid cell, g

S ← get States In Neighborhood(loc)   // set of augmenter states in agent's neighborhood
s = Level-Bias(S)                     // augmenter signature with highest augmenter level

G ← get Grid Cells(s)                 // set of all neighborhood grid cells having state s
g ← Event-Bias(G)                     // grid cell g from G with highest Event pheromone level
if g == null                          // all Event pheromone levels are equal
   g ← randomGrid(G)                  // choose g at random from G

return (g)
```

Algorithm 3: Fictive agent action selection algorithm.

location. This method results in a fleet that averages 11.68 agents over the simulation.

Note. Vessels are unable to remain at sea indefinitely and are routinely replaced. To simplify the model, the assumption was made that relieving vessels proceed to the location of the vessel being replaced and continue operations from that position, using its predecessor's amassed knowledge.

4.5. Rewards. Agents receive rewards according to their success against events. The determination of success is based on proximity, and requires that an agent be within a specified distance radius of an event at the time that event takes place. A reward of 1 is given for every success. In the event that more than one agent is successful against the same event, the reward is divided equally among those agents. All unsuccessful agents receive a reward of 0. This reward structure also applies to fictive agents as well, with two exceptions: successful fictive agents do not share rewards, and the update for fictive agent rewards is always made to the value table of its counterpart real agent, and only when the fictive agent is successful.

4.6. Policy. The selection policy used to choose agent actions is the ε-greedy technique [58]. From the available actions, the action selection mechanism previously described is employed with probability $1 - \varepsilon$. Alternatively, a random action from all possible actions is taken with probability ε. The research platform supports both a fixed ε and one that varies over the course of the simulation. Both methods were explored and the fixed ε results reported in this paper were marginally better.

4.7. Learning. During initialization, the simulation reads the scenario configuration values, including the number of time steps to simulate. The simulation then proceeds iteratively, repeating the same basic sequence. At each time step the agents mark their positions with the *Occupied* pheromone, assess their state, and select and execute an action. Events for that time step are then posted to the environment and the relevant event augmenters are applied. In time steps where

there are one or more events, all agents receive a reward per the method described earlier and update their reinforcement learning table accordingly. The process continues until the simulation ends at the last time step.

After some experimentation, the following values for reinforcement learning parameters were set for the reasons indicated as follows.

(1) Learning rate (α) = 0.1—the learning rate parameter is often varied from a high to a low value in reinforcement learning applications. However, in nonstationary environments, a fixed learning rate is recommended [7] as it gives greater weight to more recent rewards compared to those in the distant past. A low value was chosen to reduce variability in agent behavior by accepting only a fraction of the measured temporal difference.

(2) Discount rate (γ) = 0.75—this discount rate was chosen through experimentation and provided a good compromise between emphasis on immediate versus future rewards.

Learning is accomplished via the temporal difference learning algorithm previously described. Given the small size of the state space, tabular storage is used. The research platform provides the ability to evaluate joint learning, where agents update a common value table, as well as individual learning where agents have separate tables. Experimentation was carried out with both modes but the approach reported in this research employed individual learning.

Agents make continual use of state values to determine their action at every time step. However, learning takes place only in time steps in which one or more events take place, which is to say when the agent has an event to succeed against. In these time steps, the agent receives a reward greater than zero if successful or zero if unsuccessful, and the table is updated accordingly. In time steps without any events, agents continue to select actions based on the learning attained to date but no learning updates take place, as there is no possibility of success. As previously mentioned, the fictive learning mechanism is designed to help speed-up agent learning, and as such only successes (rewards greater than zero) are updated in the agent's table. The learning algorithm used by agents is depicted in Algorithm 4. The first statement is always executed and the conditional statement is executed when fictive learning is used.

4.8. Example Neighborhoods, States, and Transitions. An example of states, neighborhoods, actions, and rewards is provided here to illustrate those concepts in the context of a simulation time step. A state is a 6-bit integer value that indexes the agent learning table. It is defined as follows (the most significant bit is first): discrete augmenter level, 2 bits, values 0–3; increased since last time step, 1 bit, 0 = no 1 = yes; decreased since last time step, 1 bit, 0 = no 1 = yes; maximum in neighborhood, 1 bit, 0 = no 1 = yes; minimum in neighborhood, 1 bit, 0 = no 1 = yes.

Table 3: States by time step.

(a) Neighborhood states at time t

	1	2	3
3	Aug (Raw): 0.5 Aug (Level): 0 Increase?: N Decrease?: Y Nbhd Max?: N Nbhd Min?: N State: 4	Aug (Raw): 0.3 Aug (Level): 0 Increase?: N Decrease?: Y Nbhd Max?: N Nbhd Min?: N State: 4	Aug (Raw): 0.1 Aug (Level): 0 Increase?: N Decrease?: Y Nbhd Max?: N Nbhd Min?: Y State: 5
2	Aug (Raw): 1.7 Aug (Level): 1 Increase?: N Decrease?: Y Nbhd Max?: N Nbhd Min?: N State: 20	Aug (Raw): 1.1 Aug (Level): 1 Increase?: N Decrease?: Y Nbhd Max?: N Nbhd Min?: N State: 20	Aug (Raw): 0.7 Aug (Level): 0 Increase?: N Decrease?: Y Nbhd Max?: N Nbhd Min?: N State: 4
1	Aug (Raw): 2.1 Aug (Level): 2 Increase?: N Decrease?: Y Nbhd Max?: Y Nbhd Min?: N State: 38	Aug (Raw): 1.9 Aug (Level): 1 Increase?: N Decrease?: Y Nbhd Max?: N Nbhd Min?: N State: 20	Aug (Raw): 1.3 Aug (Level): 1 Increase?: N Decrease?: Y Nbhd Max?: N Nbhd Min?: N State: 20

(b) Neighborhood states at time $t + 1$

	1	2	3
3	Aug (Raw): 0.9 Aug (Level): 0 Increase?: Y Decrease?: N Nbhd Max?: N Nbhd Min?: N State: 8	Aug (Raw): 0.6 Aug (Level): 0 Increase?: Y Decrease?: N Nbhd Max?: N Nbhd Min?: N State: 8	Aug (Raw): 0.5 Aug (Level): 0 Increase?: Y Decrease?: N Nbhd Max?: N Nbhd Min?: Y State: 9
2	Aug (Raw): 1.7 Aug (Level): 1 Increase?: N Decrease?: N Nbhd Max?: N Nbhd Min?: N State: 16	Aug (Raw): 1.8 Aug (Level): 1 Increase?: Y Decrease?: N Nbhd Max?: N Nbhd Min?: N State: 24	Aug (Raw): 1.4 Aug (Level): 1 Increase?: Y Decrease?: N Nbhd Max?: N Nbhd Min?: N State: 24
1	Aug (Raw): 2.5 Aug (Level): 2 Increase?: Y Decrease?: N Nbhd Max?: N Nbhd Min?: N State: 48	Aug (Raw): 2.6 Aug (Level): 2 Increase?: Y Decrease?: N Nbhd Max?: Y Nbhd Min?: N State: 56	Aug (Raw): 2.2 Aug (Level): 2 Increase?: Y Decrease?: N Nbhd Max?: N Nbhd Min?: N State: 48

For simplification, we present a neighborhood of radius 1 across two time steps. Table 3(a) shows the states of the grid composing the neighborhood at time t. The *Event* pheromone

Algorithm: **agentLearn**
Input: Agent selected state s, agent next state s', agent reward r,
fictive agent selected state fs, fictive agent next state fs', fictive agent reward fr
$V(s) \leftarrow V(s) + \alpha[r + \gamma V(s') - V(s)]$
if (s != s' and $fr > 0$)
$V(fs) \leftarrow V(fs) + \alpha[fr + \gamma V(fs') - V(fs)]$

Algorithm 4: Agent learning algorithm (with fictive learning).

is the augmenter shown, and its levels are discretized as follows:

if raw augmenter < 1, Level = 0,

if 1 <= raw augmenter < 2, Level = 1,

if 2 <= raw augmenter < 3, Level = 2,

if raw augmenter >= 3, Level 3.

The neighborhood coordinates are given in row, column form. The agent is positioned in the center grid (2, 2) and can reach any of the other grids in the neighborhood. In this example, the agent's learning table (not depicted) may indicate that state 20 has the highest value, and thus grids (1, 2), (1, 3), (2, 1), and (2, 2) would be candidates for the agent's move. We note that the agent would prefer grid (1, 2) due to the *Level*-bias. However, it may end up in any of the candidate grids, based on its standing in the auction.

In time step $t + 1$ an event is posted to grid (1, 3) and the dispersal of the event's pheromones result in the new states in Table 3(b). Any agent moving to grid (1, 3) at time t shares the reward of 1 and updates its learning table for state 20, as it is the state the agent moved to immediately preceding the event (and can thus be considered an indicator of impending events). Agents who are not successful update their table for the state chosen at time t with a reward of 0.

4.9. Research Platform Description and Screenshots. The research platform, shown in Figure 9, uses the JMapViewer tool for visualization and OpenStreetMap maps (both open source products). The application window is subdivided into a control panel, a map display, and an output panel summarizing simulation results.

In the map overlay, the *Occupied* pheromone is depicted via the Blue component of the RGB color definition, with the brightness of the color expressing its value. The remaining augmenters are conveyed through the Red and Green channels either as the sole augmenter or as a composite value (*Weighted* augmenter) and are rendered with green indicating low levels and red high levels. Figure 10 illustrates the dispersion of pheromones following an event (shown as the small circle), and those left in grid cells previously visited by an agent (depicted as the diamond).

5. Experiments, Results, and Analysis

In this section, we describe the experiments conducted in a stepwise manner, and in light of the research questions posed earlier. We conclude by presenting and analyzing the results obtained from applying the information augmentation and learning approach to the Somali piracy domain.

5.1. Research Questions and Approach. Two related but distinct research questions were of interest in this work. We take up the first research question, which we subdivided into three subquestions, the first of which concerned the comparative effectiveness of different information augmenters. This subquestion was addressed in three steps as follows.

(1) *Evaluate individual augmenters.* Each of the augmenters (excluding the *Occupied* pheromone) was tested individually as the operational augmenter. Multiple trials were conducted for each and the averaged results analyzed.

(2) *Evaluate combinations of the Weighted augmenter.* To avoid overfitting, a few predetermined combinations and weightings of various augmenters as the composite *Weighted* augmenter were tested. From among the combinations evaluated, the best weightings were selected and defined as the standard for this augmenter. The composition reported here is made up of weightings of 0.5 *Ratio* augmenter and 0.5 *Shipping Lane* (map probability) augmenter.

(3) *Select the best performing augmenter.* The total number of successes using the individual augmenters were statistically assessed and the best one chosen for further evaluation.

The second subquestion addressed the relative effectiveness of the proposed fictive learning method over standard reinforcement learning. We expanded this evaluation by adding a third nonlearning mode, in order to complete a more thorough analysis. For each operational augmenter we evaluated the following.

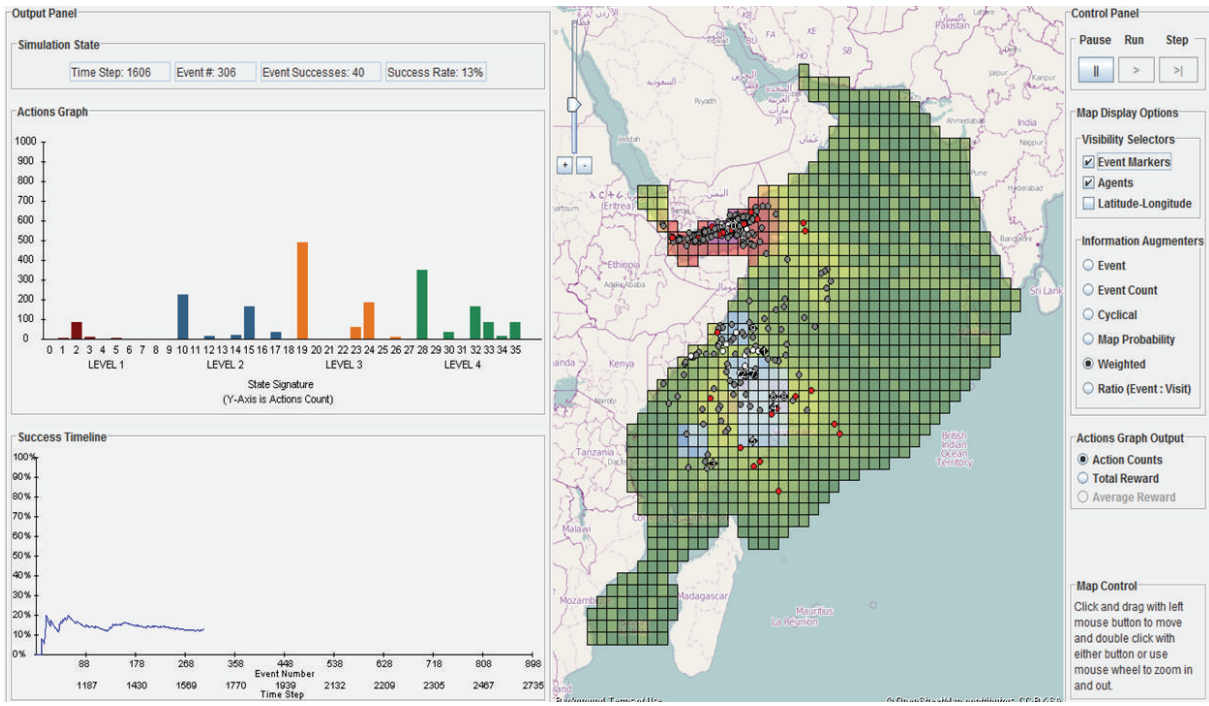
(1) The *Level-bias* heuristic in nonlearning mode.

(2) Standard reinforcement learning (using the temporal difference approach).

(3) The proposed fictive learning method.

The final subquestion was addressed by evaluating four policies for the best performing augmenter, using a complement of 25 agents. The first two techniques are nonlearning and the last two utilize fictive learning as follows.

(1) *Random Moves.* The absolute minimum expected number of successes was established by conducting



Simulation state: shows time step, event number, and success information	Map area: Displays events (circles) and agents (numbered diamonds) and their pheromone fields Event colors • Red—recent events • White—successfully mitigated events • Gray—nonrecent events	Simulation controls: allow the simulation to be run, paused, or stepped through
Actions graph: outputs the action counts or reward information		Visibility selectors: toggle visibility of event markers, agents, latitude-longitude
Success timeline: displays a line marking the level of success as events and time progress		Information augmenters—select which augmenter value is displayed in map area

Figure 9: Research platform screenshot.

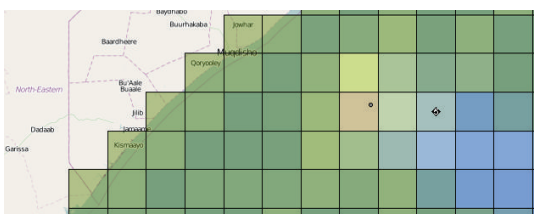


Figure 10: Pheromones dispersed by an event (circle) and agent (numbered diamond).

simulation runs where at each time step each agent simply moved randomly to any grid cell within its local neighborhood. These trials were conducted without the use of any information augmenters or reinforcement learning methods.

(2) *Random Actions.* This set of simulation trials was conducted using $\varepsilon = 1.0$, which is in effect a non-learning method. The approach amounted to agents randomly choosing any action possible from the operational augmenter signatures in grid cells within its neighborhood. This differed from the *Random Moves* case in that while the agents did not get to select the signature of the operational augmenter through a value table, the *Event-bias* was still used to determine the destination grid cell from the set returned by the randomly chosen end-state. In this case, it was expected that performance would improve since the *Event-bias* ensured that agents remained in, or were drawn to, areas affected by events. It was determined *a priori* that the proposed learning methods would have to outperform the number of successes obtained through this approach, in order to be considered viable.

(3) *Random Actions if State Values are Equal.* The method applied in this set of trials utilized $\varepsilon = 0.05$ and in it agents selected either the best (highest state value) action from prior learning (using fictive learning) or a random action with probability ε or when state values for the operational augmenter states within the neighborhood were all equal. The *Event-bias* was then applied to select the destination grid cell. Since this was approach which made use of learned values, the expectation was that it would outperform the simple *Random Actions* case.

(4) *Level-Bias if State Values are Equal.* Finally, we tested the viability of our proposed augmentation with

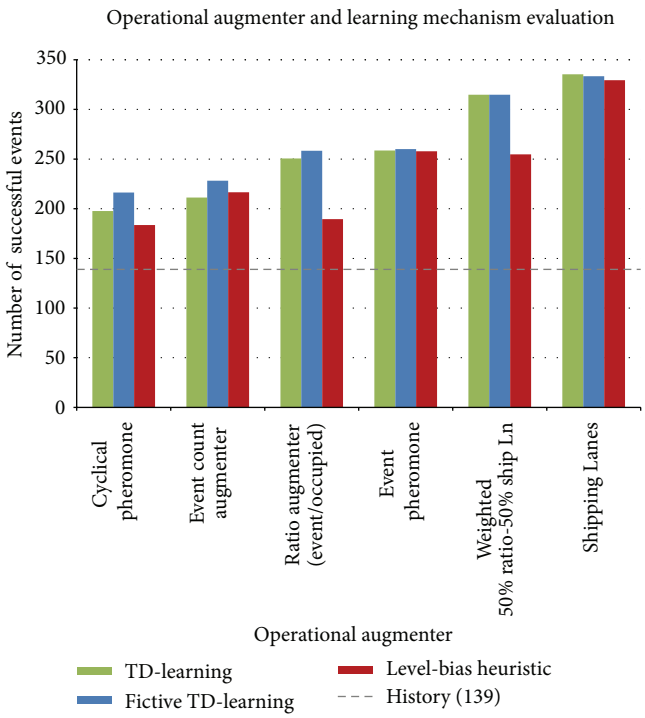


FIGURE 11: Summary of simulation results by augmenters and learning mechanisms.

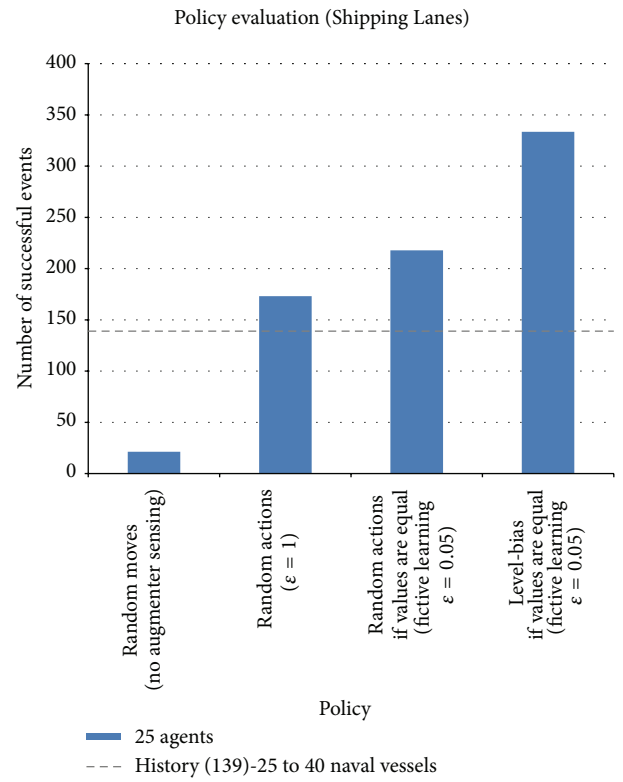


FIGURE 12: Summary of policy evaluation for the *Shipping Lanes* augmenter.

fictive learning and the *Level-bias*. The test scenario was executed multiple times, with agents selecting actions based on the best (highest state value) action from prior learning or by applying the *Level-bias* when no distinct best action existed. A random action was chosen with probability ε (0.05). The *Event-bias* was used to select the destination grid cell. Given the demonstrated effectiveness of the *Level-bias* heuristic, it was expected that this policy would outperform the *Random Actions if State Values Equal* policy.

The final research question concerned the effectiveness of the information augmentation and fictive learning approach, when applied to real-world problems. We addressed this in two steps as follows.

(1) *Execute simulations for different agent counts.* For the test application, the number of agents available at any particular time is only approximately stated in the available data. To account for this uncertainty, three different approaches to setting the agent population, all conservative with respect to the available data, were tested through multiple simulation trials for each of the augmenters, using the proposed fictive learning technique.

(2) *Compare historical and simulation outcomes.* The number of successes observed in simulation results was compared to the historical record of actual successes in defending against pirate attacks, using each of the different augmenters and agent counts tested. A comparison was considered favorable if a 95% confidence interval for the mean number of successes in simulation trials included or was greater than the number of historical successes.

5.2. Results. A set of 30 simulation trials were conducted for each of configurations described above. In each trial agents started without any knowledge and executed the appropriate policy and reinforcement learning algorithm with the described augmenters. Events in which one or more agents were positioned within the specified proximity were considered a success and the appropriate statistics calculated. The simulation results are presented here and analyzed in the next section. Figure 11 illustrates the output for combinations of learning mechanisms and individual augmenters.

Figure 12 depicts the outcome of action selection policies evaluation for the *Shipping Lanes* map probability augmenter, which was the operational augmenter with the best performance.

Figure 13 portrays the results of employing the proposed fictive learning method with the *Level-bias if State Values Equal* policy for combinations of augmenters and agent count schemes.

Table 4 summarizes the results shown in Figure 12 along with statistical details. The "Agents" column in Table 3 indicates how many agents were present during the simulation. In that column, "25" means that 25 agents were present during the entire simulation, "5 → 25 by time" means that the simulation started with 5 agents and ended with 25, with

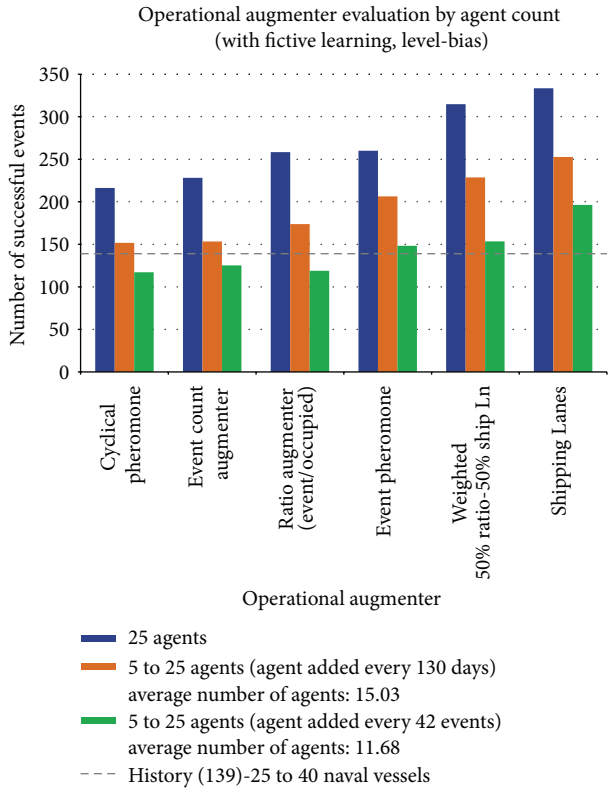


Figure 13: Summary of operational augmenter by agent count schemes.

one agent being added every 130 time steps, and "5 → 25 by event" means that the simulation started with 5 agents and ended with 25, with one agent being added every 42 events. For each of the combinations, 30 executions of the simulation were performed. The "Min", "Max", "Q1", and "Q3" show the minimum, maximum, first quartile, and third quartile values for the number of successes during the 30 executions. Similarly, the "Median", "Mean", and "Std Dev" columns report those statistics for successes during the 30 trials. Finally, the "Confidence interval" column reports a confidence interval for the mean number of successes during the 30 trials, calculated using the Student's t-distribution with $\alpha = 0.05$, confidence level $c = (1-\alpha) = 0.95$, and critical value $t_c = 2.05$. When examining the table, recall that the historical record had 139 successes with 25 to 40 naval vessels.

5.3. Analysis. We now analyze each set of results beginning with learning methods evaluation. The statistical significance reported here was established through conducting an analysis of variance (ANOVA) and applying Tukey's Test to establish differences (the less conservative Fisher test only changed one outcome from those reported here).

(1) *Level-Bias Heuristic.* As mentioned, this heuristic is a nonlearning approach that simply selects the highest level of the operational augmenter found in the neighborhood, to which the *Event-bias* is then applied. Given that description, the heuristic is likely to perform well when the distribution of operational augmenter signatures at the time of an event tend toward the higher discretized levels. Results using the heuristic were statistically indistinguishable from the learning methods when using the *Event* pheromone operational augmenter. Using the best performing *Shipping Lanes* augmenter, both learning methods (standard and fictive reinforcement learning) outperformed the heuristic by a small, but statistically significant, margin. When applying Tukey's test, the heuristic was statistically equivalent to both reinforcement learning methods, but Fisher's test showed a difference in favor of the fictive learning approach. These results may be explained by the fact that attacks tended to occur in area previously attacked (especially the case in the Gulf of Aden) and in places with very heavy shipping traffic (nearly half of all pirate attacks took place at the highest discretized *Shipping Lanes* level). Thus, simply biasing to those higher levels led to good performance. The drawback to this heuristic is illustrated by its performance when used with operational augmenters for which the distribution of states at attack time does not favor higher augmenter levels (*Cyclical*, *Ratio*, and *Weighted* augmenters). It's worth noting that the choosing of parameters for the proposed augmenters is more art than science, and a resultant state distribution not amenable to application of this heuristic is very likely. In spite of its performance, the greatest limitation of the heuristic is that it is a non-learning rule and is thus not adaptable to changes in the behavior of the human actors behind the events. In summary, the proposed fictive learning method statistically outperformed the heuristic for four or five of the six augmenters, depending on the statistical test used.

(2) *Standard Reinforcement Learning.* Use of nonfictive reinforcement learning was statistically superior or equivalent to the performance of the *Level-bias* in all augmenter cases.

(3) *Fictive Reinforcement Learning.* The analysis of the comparative performance of the *Level-bias* heuristic and the proposed fictive learning approach has been presented in the section analyzing the former. Here, we compare fictive learning to standard (nonfictive) reinforcement learning. Our proposed learning approach was statistically superior to using standard reinforcement learning for two of the operational augmenters. The statistically significant improvements however were modest (approximately 8% in both cases). Fictive learning was statistically equivalent to standard reinforcement learning in the other cases. Given these results, the fictive reinforcement learning approach was chosen as the learning method for further experimentation.

The comparative analysis of the action selection policies follows. In all cases, except for the *Random Moves* case which does not sense augmenters, the *Shipping Lanes* augmenter was used.

Table 4: Statistical results for the Somali piracy application.

	Method	Agents	Min.	Max.	Q1	Q3	Median	Mean	Std Dev	Confidence Interval
Policies	*Random moves (no augmenter)*	25	11	33	19.00	24.00	20.50	21.33	4.54	[19.64, 23.03]
	Random actions (shipping lanes)	25	146	194	164.25	181.75	175.50	173.10	11.39	[168.85, 177.35]
	Random actions if state values equal (shipping lanes)	25	169	261	206.75	229.25	218.50	217.87	19.99	[210.40, 225.33]
Operational augmenters (Fictive learn + level-bias)	*Cyclical*	25	179	259	203.75	222.75	216.00	216.30	18.76	[209.29, 223.31]
		5 → 25 by time	100	184	137.25	168.50	151.00	151.73	20.29	[144.16, 159.31]
		5 → 25 by event	76	153	106.75	130.50	118.50	117.23	20.36	[109.63, 124.84]
	Event Count	25	177	264	213.25	247.50	226.50	228.23	21.57	[220.18, 236.29]
		5 → 25 by time	94	204	127.75	173.75	158.50	153.33	31.76	[141.48, 165.19]
		5 → 25 by event	88	163	110.00	138.25	137.50	125.30	18.70	[118.32, 132.28]
	Ratio	25	221	285	248.25	270.25	258.00	258.37	16.73	[252.12, 264.61]
		5 → 25 by time	146	207	161.25	189.00	170.50	173.83	18.01	[167.11, 180.56]
		5 → 25 by event	55	170	104.00	131.00	118.50	118.97	25.75	[109.35, 128.58]
	Event	25	232	278	255.25	266.75	261.50	260.07	10.15	[256.28, 263.86]
		5 → 25 by time	181	232	200.00	213.25	205.00	206.40	13.31	[201.43, 211.37]
		5 → 25 by event	107	173	142.25	158.25	151.50	148.27	15.49	[142.48, 154.05]
	Weighted	25	295	333	305.50	323.00	316.00	314.73	10.73	[310.72, 318.74]
		5 → 25 by time	196	272	214.25	243.75	226.00	228.63	20.44	[221.00, 236.26]
		5 → 25 by event	83	187	141.25	169.00	154.50	153.43	21.75	[145.31, 161.55]
	Shipping Lanes	25	321	348	329.00	338.00	333.50	333.43	6.86	[330.87, 336.00]
		5 → 25 by time	220	273	250.00	259.00	254.00	252.73	11.81	[248.33, 257.14]
		5 → 25 by event	169	216	191.75	203.00	197.50	196.30	11.15	[192.14, 200.46]

(1) *Random Moves*. Given the lack of the *Event-bias* mechanism to keep agents in the areas experiencing attacks, the *Random Moves* policy resulted in a very low success rate, with a mean of only 21.33 successes.

(2) *Random Actions*. The *Random Actions* policy fared significantly better as the *Event-bias* mechanism ensures that agents remain in or are attracted to areas affected by events. This case illustrates the importance of that aspect of agent behavior, with a mean outcome of 173.10 successes, an over eight-fold increase over the *Random Moves* policy.

(3) *Random Actions if State Values are Equal*. In this case, actions learned using fictive learning were executed whenever a clear best action (highest value) was indicated in the learning table, and a random action was chosen otherwise. This policy resulted in a 25% increase in the number of successes, when compared to the *Random Actions* case (217.87 versus 173.10).

(4) *Level-Bias Heuristic with Fictive Learning*. Finally, the policy utilizing both the *Level-bias* heuristic and fictive learning was used. The mean of 333.43 successes demonstrated the significance of supplementing action selection with the *Level-bias* heuristic whenever a clear best action was not indicated by entries in the learning table. The number of successes posted an increase of 53% over the *Random Actions if State Values Equal* case.

The analysis of results for the operational augmenters versus fleet size, are now presented. As might be expected, the simulation results showed that for any given action selection basis, the fixed size fleet (25 agents throughout the simulation) performed best, followed by the time-based variable fleet (which averaged 15.03 agents over the simulation) and ending with the event-based variable fleet (which averaged 11.68 agents over the simulation). The latter two fleet count schemes are extremely conservative in light documented history.

For all fleet configurations, when using the best performing Shipping Lanes augmenter, the environment augmentation and fictive learning methods produced more successful responses to pirate attacks than the historical number of successful responses by the actual naval vessels (139). The patrol and search procedures and doctrine used by the naval vessels have not been made public, so a definitive

explanation of the difference in results is not possible. We conjecture that the commanders of the naval vessels were following doctrine that was either static or imposed by external superiors or both, and thus they could not benefit from learning the signatures of likely pirate attacks in the way that the algorithm did. Of the 18 combinations of augmenters and fleet configurations investigated, the 95% confidence interval for simulation results either exceeded or included the historical number of successes in all but three cases (Event Count, Cyclical, and Ratio augmenters with the event-based fleet size case). The time-based variable fleet and the fixed fleet exceeded the historical number of successes with all augmenter combinations.

Detailed analysis for the 25 agent fixed fleet using the different augmenters is shown here in order of increasing effectiveness.

(1) The *Cyclical* augmenter yielded the lowest number of successes (216.30), besting the historical record by almost 56%. The likely cause of this performance is that given time period it considers (two and one half years) this augmenter may increase agent actions to grid cells that may have experienced a lot of attacks in the past but are no longer favored by pirates.

(2) The *Event Count* augmenter's performance was close to that of the *Cyclical* augmenter as it also operates on information that may be two and one half years old. A possible explanation for the increase (228.23 versus 216.30) is that the *Cyclical* augmenter is more diffuse as it is applied over a radius around an attack site, whereas the *Event Count* augmenter applies only to attacked grids.

(3) The *Ratio* augmenter's results averaged 258.37 successes, an 86% outperformance of history. Given the composition of this augmenter, ratio of *Event* pheromone to *Occupied* pheromone, these results closely parallel those obtained with the former.

(4) The *Event* augmenter yielded a mean of 260.07 successes, an 87% improvement over the historical record. Since this method uses both the *Level-bias* and the *Event-bias*, this is the greedy case (choose the grid with the highest concentration of *Event* pheromone from those containing the highest discretized levels of that augmenter).

(5) The *Weighted* augmenter used in this case was composed of 0.5 *Ratio* and 0.5 *Shipping Lanes* weighting. It posted a mean of 314.73 successes, more than 226% of the historical record. We believe that the inclusion of the *Shipping Lanes* component is the driver for this performance.

(6) The *Shipping Lanes* augmenter yielded the best outcome, with a mean of 333.43 successes (240% of history). The likely cause for this performance is that the pirate attacks were more heavily skewed toward areas of very high maritime traffic. That, combined with the *Level-bias* in effect reduced the action space to areas of high levels of the augmenter containing high concentrations of the *Event* pheromone.

6. Conclusions, Limitations, and Future Work

With a mean of over 333 successes, our learning method with the *Shipping Lanes* augmenter enabled a successful response to over 37% of the historical sequence of 899 attacks. The success rate at various time points during simulation execution exceeded 50%. This performance substantially surpassed the actual historical record of 139 successful responses by naval vessels documented in the International Maritime Bureau reports, and was accomplished with a number of agents possibly smaller than the historical 25–40 vessel fleet reported in [1]. Use of the historic record of events, consisting only of dates and locations, combined with the performance attained lends credence to the viability and usefulness of our information augmentation and fictive learning approaches.

We acknowledge some limitations of this work as follows:

(1) Our focus in this research was to evaluate the use of model-free reinforcement learning methods in the maritime piracy domain. We therefore avoided the model-based and POMDP approaches. However, we conjecture that employing a mechanism robust to nonstationarity could result in improved performance.

(2) The number of events in the relatively confined Gulf of Aden likely contributed to the performance attained, since agents responding there were able to continually react to nearby attacks. Of course, our method did enable this geography to be exploited.

(3) This is not a two-sided game-theoretic model, and as such treats the sequence of events as deterministic; the pirates do not explicitly respond to defensive successes, for instance. However, the historical attack sequence already factors in at least some of the deterrent effect of thwarts due to historical interventions and thwarts. Many of the historical attacks took place despite successes by forces against piracy in the area in the recent past, that is, naval successes did not always dissuade the pirates. Adaptability by the pirates to naval tactics would likely take the form of a planned attack either not taking place at all, or being moved in time and/or space. Our methodology is robust to this adaptive behavior since only attacks that actually take place are marked with augmenters and factored into the reinforcement learning process. Thus, we always respond to the actual event sequence, without regard to whether the time and place of those events is a result of adaptive behavior.

(4) No limits were placed on where agents could move; that is, no ocean grid cells were considered to be off limits or impassable. This may not always reflect reality, for example, transient considerations such

as weather may prevent sea or air operations in particular areas. However, it is worth noting that any weather impacting naval vessels are likely to have a similar or greater impact on the smaller pirate skiff and mother ships. Nevertheless, this can be remedied by restricting the world at each time step to a subset of the grid cells.

(5) The agents are homogeneous and act on local information only. This often led to agents congregating at events in their neighborhoods, while other events remain unattended to. This shortfall may be addressed by specializing agents in order to enable a search function, or by subdividing areas of responsibility.

(6) The determination of weights for the *Weighted* augmenter was done *a priori*. An automated process to establish such weights and allow them to vary over time would be beneficial.

(7) Choosing a single operational augmenter *a priori* may limit the success of our approach. A method to vary the operational augmenter during execution, without exploding the state space could provide increased performance.

The research may be extended in a number of ways. The limitations listed earlier implicitly suggested opportunities for future work. Some additional areas for consideration include the following.

(1) Develop and apply a learning method better able to deal with non-stationarity.

(2) Convert agents from local to global behavior; Jones and Matarić provide an example process for this [59].

(3) Create distinct agent roles, such as patrollers and responders, with differing capabilities.

(4) Dynamically allocate agent roles during simulation execution.

(5) Optimize the number of agents for a given scenario.

(6) Further develop this approach towards its actual use as a decision support system for tasking naval forces.

(7) Define and test additional information augmenters, alone and in combination.

In spite of of these limitations, the obtained results justify further investigation into the use of model-free, fictive reinforcement learning with simple reactive agents, which through information augmentation are able to respond to patterns in a spatiotemporal environment.

Conflict of Interests

The authors declare that there is no conflict of interests regarding the publication of this paper.

Acknowledgments

The authors thank the anonymous reviewer of an earlier version of this paper, whose comments led to substantial improvements in the paper.

References

[1] J. Bahadur, *The Pirates of Somalia: Inside Their Hidden World*, Pantheon Books, New York, NY, USA, 2011.

[2] S. J. Russell and P. Norvig, *Artificial Intelligence: A Modern Approach*, Prentice Hall, Upper Saddle River, NJ, USA, 2nd edition, 2003.

[3] M. J. Wooldridge, *An Introduction to Multiagent Systems*, Wiley & Sons, West Sussex, UK, 2nd edition, 2009.

[4] G. P. Williams, *Chaos Theory Tamed*, Joseph Henry Press, Washington, DC, USA, 1997.

[5] P. Tan, M. Steinbach, and V. Kumar, *Introduction to Data Mining*, Addison Wesley, New York, NY, USA, 2006.

[6] H. Witten, E. Frank, and M. A. Hall, *Data Mining: Practical Machine Learning Tools and Techniques*, Morgan Kaufmann, New York, NY, USA, 3rd edition, 2011.

[7] R. S. Sutton and A. G. Barto, *Reinforcement Learning: An Introduction*, MIT Press, Cambridge, Mass, USA, 1998.

[8] D. Floreano and C. Mattiussi, *Bio-Inspired Artificial Intelligence: Theories, Methods, and Technologies*, MIT Press, Cambridge, Mass, USA, 2008.

[9] M. Dorigo and T. Stützle, *Ant Colony Optimization*, MIT Press, Cambridge, Mass, USA, 2004.

[10] J. Kennedy and R. Eberhart, *Swarm Intelligence*, Morgan Kaufmann, San Francisco, Calif, USA, 2001.

[11] Z. Michalewicz and D. B. Fogel, *How to Solve it: Modern Heuristics, 2nd Revised and Extended Edition*, Springer, New York, NY, USA, 2004.

[12] International Maritime Bureau, *Piracy and Armed Robbery Against Ships Annual Report 1 January-31 December 2005*, International Maritime Bureau, London, UK, 2006.

[13] International Maritime Bureau, *Piracy and Armed Robbery Against Ships Annual Report 1 January-31 December 2006*, International Maritime Bureau, London, UK, 2007.

[14] International Maritime Bureau, *Piracy and Armed Robbery Against Ships Annual Report 1 January-31 December 2007*, International Maritime Bureau, London, UK, 2008.

[15] International Maritime Bureau, *Piracy and Armed Robbery Against Ships Annual Report 1 January-31 December 2008*, International Maritime Bureau, London, UK, 2009.

[16] International Maritime Bureau, *Piracy and Armed Robbery Against Ships Annual Report 1 January-31 December 2009*, International Maritime Bureau, London, UK, 2010.

[17] International Maritime Bureau, *Piracy and Armed Robbery Against Ships Annual Report 1 January-31 December 2010*, International Maritime Bureau, London, UK, 2010.

[18] International Maritime Bureau, *Piracy and Armed Robbery Against Ships Report for the Period 1 January-31 December 2011*, International Maritime Bureau, London, UK, 2012.

[19] International Maritime Bureau, *Piracy and Armed Robbery Against Ships Report for the Period 1 January-30 June 2012*, International Maritime Bureau, London, UK, 2012.

[20] R. I. Rotberg, "Combating maritime piracy: a policy brief with recommendations for action," Policy Brief #11, World Peace Foundation, Medford Somerville, Mass, USA, 2010.

[21] Oceans Beyond Piracy, "The Economic Cost of Somali Piracy 2011," 2011, http://oceansbeyondpiracy.org/sites/default/files/economic_cost_of_piracy_2011.pdf.

[22] P. Eichstaedt, *Pirate State: Inside Somalia's Terrorism at Sea*, Chicago Review Press, Chicago, Ill, USA, 2010.

[23] A. Shortland and M. Vothknecht, "Combating maritime terrorism off the Coast of Somalia," Working Paper 47, European Security Economics, Vienna, Austria, 2011.

[24] Combined Maritime Forces, 2012, http://www.cusnc.navy.mil/cmf/cmf_command.html.

[25] R. Mirshak, "Ship Response Capability Models for Counter-Piracy Patrols in the Gulf of Aden," Technical Memorandum DRDC CORA TM, 2011-139, Maritime Operations Research Team, Defence R&D Canada, Ottawa, Canada, 2011, http://cradpdf.drdc-rddc.gc.ca/inbasket/DRP_CORA.111027_0949.TM2011-139_A1b.pdf.

[26] S. Marsland, *Machine Learning: An Algorithmic Perspective*, Chapman & Hall/CRC, New York, NY, USA, 2009.

[27] C. H. Watkins and P. Dayan, "Q-learning," *Machine Learning*, vol. 8, no. 3-4, pp. 279–292, 1992.

[28] B. C. da Silva, E. W. Basso, A. L. C. Bazzan, and P. M. Engel, "Dealing with non-stationary environments using context detection," in *Proceedings of the 23rd International Conference on Machine Learning (ICML '06)*, pp. 217–224, June 2006.

[29] V. Bulitko, N. Sturtevant, and M. Kazakevich, "Speeding up learning in reel-time search via automatic state abstraction," in *Proceedings of the 20th National Conference on Artificial Intelligence and the 17th Innovative Applications of Artificial Intelligence Conference (AAAI '05)*, pp. 1349–1354, July 2005.

[30] L. Panait and S. Luke, "Cooperative multi-agent learning: the state of the art," *Autonomous Agents and Multi-Agent Systems*, vol. 11, no. 3, pp. 387–434, 2005.

[31] L. Buşoniu, R. Babuška, and B. De Schutter, "A comprehensive survey of multiagent reinforcement learning," *IEEE Transactions on Systems, Man and Cybernetics C*, vol. 38, no. 2, pp. 156–172, 2008.

[32] L. Jing and N. Cerone, "Thoughts on multiagent learning: from a reinforcement learning perspective," Technical Report CSE-2010-07, Department of Computer Science and Engineering, York University, Ontario, Canda, 2010.

[33] L. Oliwenstein, "From dendrites to decisions," *Engineering and Science*, vol. 74, no. 3, pp. 14–21, 2011.

[34] P. R. Montague, B. King-Casas, and J. D. Cohen, "Imaging valuation models in human choice," *Annual Review of Neuroscience*, vol. 29, pp. 417–448, 2006.

[35] T. Lohrenz, K. McCabe, C. F. Camerer, and P. R. Montague, "Neural signature of fictive learning signals in a sequential investment task," *Proceedings of the National Academy of Sciences of the United States of America*, vol. 104, no. 22, pp. 9493–9498, 2007.

[36] A. Agogino and K. Tumer, "Regulating air traffic flow with coupled agents," in *Proceedings of the 7th International Joint Conference on Autonomous Agents and Multiagent Systems*, vol. 2, pp. 535–542, 2008.

[37] K. Tumer and N. Khani, "Learning from actions not taken in multiagent systems," *Advances in Complex Systems*, vol. 12, no. 4-5, pp. 455–473, 2009.

[38] D. M. Gordon, *Ants at Work: How An Insect Society Is Organized*, The Free Press, New York, NY, USA, 1999.

[39] K. L. Huang and C. J. Liao, "Ant colony optimization combined with taboo search for the job shop scheduling problem," *Computers and Operations Research*, vol. 35, no. 4, pp. 1030–1046, 2008.

[40] Z. J. Lee, C. Y. Lee, and S. F. Su, "An immunity-based ant colony optimization algorithm for solving weapon-target assignment problem," *Applied Soft Computing Journal*, vol. 2, no. 1, pp. 39–47, 2002.

[41] J. Bautista and J. Pereira, "Ant algorithms for a time and space constrained assembly line balancing problem," *European Journal of Operational Research*, vol. 177, no. 3, pp. 2016–2032, 2007.

[42] M. Gosnell, S. O'Hara, and M. Simon, "Spatially decomposed searching by heterogeneous unmanned systems," in *Proceedings of the International Conference on Integration of Knowledge Intensive Multi-Agent Systems (KIMAS '07)*, pp. 52–57, May 2007.

[43] J. G. M. Fu and M. H. Ang, "Probabilistic ants (PAnts) in multi-agent patrolling," in *Proceedings of the International Conference on Advanced Intelligent Mechatronics*, pp. 1371–1376, 2009.

[44] H. Chu, A. Glad, O. Simonin, F. Sempé, A. Drogoul, and F. Charpillet, "Swarm approaches for the patrolling problem, information propagation vs. pheromone evaporation," in *Proceedings of the 19th IEEE International Conference on Tools with Artificial Intelligence (ICTAI '07)*, pp. 442–449, October 2007.

[45] J. A. Sauter, R. Matthews, H. Van Dyke Parunak, and S. A. Brueckner, "Performance of digital pheromones for swarming vehicle control," in *Proceedings of the 4th International Conference on Autonomous Agents and Multi agent Systems (AAMAS '05)*, pp. 1037–1044, July 2005.

[46] N. Monekosso and P. Remagnino, "An analysis of the pheromone Q-learning algorithm," in *Proceedings of the 8th Ibero-American Conference on Artificial Intelligence*, pp. 224–232, 2002.

[47] V. Furtado, A. Melo, A. L. V. Coelho, R. Menezes, and R. Perrone, "A bio-inspired crime simulation model," *Decision Support Systems*, vol. 48, no. 1, pp. 282–292, 2009.

[48] O. Vaněk, B. Bošanský, M. Jakob, and M. Pěchouček, "Transiting areas patrolled by a mobile adversary," in *Proceedings of the IEEE Conference on Computational Intelligence and Games (CIG '10)*, pp. 9–16, August 2010.

[49] M. Jakob, O. Vanek, S. Urban, P. Benda, and M. Pechoucek, "Agent C: agent-based testbed for adversarial modeling and reasoning in the maritime domain," in *Proceedings of the International Conference on Autonomous and Multiagent Systems*, pp. 1641–1642, 2010.

[50] M. Jakob, O. Vaněk, and M. Pěchouček, "Using agents to improve international maritime transport security," *IEEE Intelligent Systems*, vol. 26, no. 1, pp. 90–95, 2011.

[51] L. A. Slootmaker, *Countering piracy with the next-generation piracy performance surface model [M.S. thesis]*, Naval Postgraduate School, Monterey, Calif, USA, 2011.

[52] J. Decraene, M. Anderson, and M. Low, "Maritime counter-piracy study using agent-based simulations," in *Proceedings of the Spring Simulation Multiconference (SpringSim '10)*, pp. 82–89, April 2010.

[53] D. Walton, E. Paulo, C. J. McCarthy, and R. Vaidyanathan, "Modeling force response to small boat attack against high value commercial ships," in *Proceedings of the 2005 Winter Simulation Conference*, pp. 988–991, December 2005.

[54] M. T. J. Spaan, "Partially observable markov decision processes," in *Reinforcement Learning State-of-the-Art*, M. Wiering and M. van Otterlo, Eds., Springer, Berlin, Germany, 2012.

[55] B. Weitjens, *Geopredict: Geographical crime forecasting for varying situations [M.S. thesis]*, Vrije Universiteit, Amsterdam, The Netherlands, 2010.

[56] P. Kaluza, A. Kölzsch, M. T. Gastner, and B. Blasius, "The complex network of global cargo ship movements," 2010, http://arxiv.org/abs/1001.2172/.

[57] M. West, "Asset allocation to cover a region of piracy," Report DSTO-TN-1030, Maritime Operations Division, Defence Science and Technology Organisation, Australian Government Department of Defense, Canberra, Australia, 2011.

[58] E. Alpaydin, *Introduction to Machine Learning*, MIT Press, Cambridge, Mass, USA, 2nd edition, 2010.

[59] C. Jones and M. J. Matarić, "From local to global behavior in intelligent self-assembly," in *Proceedings of the IEEE International Conference on Robotics and Automation*, pp. 721–726, September 2003.