PLOS ONE

RESEARCH ARTICLE

Multi-Agent Patrolling under Uncertainty and Threats

Shaofei Chen[1,2]*, Feng Wu[3], Lincheng Shen[1], Jing Chen[1], Sarvapali D. Ramchurn[2]

1 College of Mechatronics and Automation, National University of Defense Technology, Changsha, Hunan, 410073, China, **2** School of Electronics and Computer Science, University of Southampton, Southampton, SO171BJ, United Kingdom, **3** School of Computer Science and Technology, University of Science and Technology of China, Hefei, Anhui, 230026, China

* chensf005@163.com, sc16g13@ecs.soton.ac.uk

Citation: Chen S, Wu F, Shen L, Chen J, Ramchurn SD (2015) Multi-Agent Patrolling under Uncertainty and Threats. PLoS ONE 10(6): e0130154. doi:10.1371/journal.pone.0130154

Academic Editor: Yong Deng, Southwest University, CHINA

Received: January 19, 2015

Accepted: May 17, 2015

Published: June 18, 2015

Data Availability Statement: All relevant data are within the paper and its Supporting Information files.

Funding: SC is supported in part by China Scholarship Council for sponsoring his visiting study in University of Southampton, Hunan Provincial Innovation Foundation for Postgraduate (No. CX2013B013), and National University of Defense Technology for Outstanding Graduate Innovation Fund (No. B130302). SC, LS, and JC acknowledge support by National Natural Science Foundation of China (No. 61403411). The funders had no role in study design, data collection and analysis, decision to publish, or preparation of the manuscript.

Abstract

We investigate a multi-agent patrolling problem where information is distributed alongside threats in environments with uncertainties. Specifically, the information and threat at each location are independently modelled as *multi-state Markov chains*, whose states are not observed until the location is visited by an agent. While agents will obtain information at a location, they may also suffer damage from the threat at that location. Therefore, the goal of the agents is to gather as much information as possible while mitigating the damage incurred. To address this challenge, we formulate the single-agent patrolling problem as a *Partially Observable Markov Decision Process* (POMDP) and propose a computationally efficient algorithm to solve this model. Building upon this, to compute patrols for multiple agents, the single-agent algorithm is extended for each agent with the aim of maximising its marginal contribution to the team. We empirically evaluate our algorithm on problems of multi-agent patrolling and show that it outperforms a baseline algorithm up to 44% for 10 agents and by 21% for 15 agents in large domains.

Introduction

Unmanned Aerial Vehicles (UAVs) are increasingly becoming essential tools to carry out situational awareness tasks in a number of real-world applications ranging from disaster response [1–3] and security surveillance [3–5]. In these scenarios, multiple UAVs may be deployed to gather information at specific locations as quickly as possible in order to support an ongoing operation. However, such problems are often liable to a high degree of dynamism (e.g., fires may spread, wind direction may change) and uncertainty (e.g., it may not be possible to completely observe the causes of fires or the location of casualties may not be exactly known), and may also contain a number of hazards or threats for the UAVs (e.g., UAVs may fly close to buildings on fire or debris may fall on the UAVs).

In this paper, we consider the scenario where a set of UAVs aim to patrol the area to gather as much information as possible while minimising the negative impact of threats. Crucially, they aim to do so within an environment that is *partially observable* (i.e., the features of the

Competing Interests: The authors have declared that no competing interests exist.

locations are only fully observable where the UAV is located and partially observable at other locations). Hence, when planning the sequence of locations to visit, UAVs have the difficult task of estimating the information to be gained and threats to be encountered at these locations. This problem is compounded by the fact that the dynamism inherent to the environment may cause the information and threats at each location to change over time (i.e., the environment is stochastic). For example, when UAVs visit a building in a disaster area, the building states (intact, about to collapse, collapsing, or collapsed) may correspond to threat states (levels) for UAVs, and the threat at each location may be changing stochastically, such that it switches between "about to collapse" to "collapsed" due to an aftershock [6]. The information in the environment may also change dynamically (e.g., a victim may get out of danger or the fire may get close to a victim).

To date, a number of approaches to information gathering with teams of UAVs have been proposed. However, most of the work [3, 7, 8] focus on developing algorithms for UAVs gathering information in dynamic environments where the model of the features of the environment is *fully observable* and *stationary* (see Related Work section for more details). Furthermore, none of these approaches have considered how threats may affect the information gathering process while the environment is partially observable and non-stationary. Unless such issues are tackled, we believe it is unlikely that large UAV deployments in real major disaster will be feasible.

In recent years, *agent-based modelling* has been effectively used to formulate and solve the problems of planning in environments characterized by uncertainties [9]. In agent-based models, an agent is an encapsulated computer system that is situated in some environment and that is capable of flexible, autonomous action in that environment in order to meet its design objectives [10]. Such agents are either software or hardware (e.g., robots or unmanned autonomous systems (UAS)). In particular, operating in uncertain environments, autonomous agents have to deal with executing actions that may not have the intended results, with environments that change while the agent is operating, and with making observations that might not be completely accurate.

Against this background, we propose a agent-based model for patrolling under uncertainty and threats and go on to develop a novel algorithm to solve the planning problem that it poses. In more detail, we first model the information and threats on a graph representing the environment, where the information and threat at each location are independently modelled as *multi-state Markov chains* (which captures the non-stationary feature), whose states are not observed until the location is visited by an agent (which captures the partially observable feature). Then, we cast the single-agent patrolling problem as a *Partially Observable Markov Decision Process* (POMDP), which provides a rich model for planning and acting in partially observable stochastic domains [11]. Unfortunately, existing POMDP solvers are very inefficient to solve our POMDP formulation due to the exponential growth of the number of possible paths of agents in the size of the graph and the number of the possible observations along each possible path (see Related Work section for more detail). Hence, we propose an online algorithm to solve the patrolling problem for one agent at a time. (In computer science, an online algorithm is one that can process its input piece-by-piece in a serial fashion, i.e., in the order that the input is fed to the algorithm, without having the entire input available from the start. In contrast, an offline algorithm is given the whole problem data from the beginning and is required to output an answer which solves the problem at hand.) In particular, the algorithm utilises a predictive heuristic that only refers to the possible paths (looking ahead several steps) from the current position of the agent. Building upon this, to compute patrolling policies for multiple agents, the single-agent algorithm is extended for each agent with the aim of maximising its marginal contribution to the team. In summary, this paper advances the state of the art in the following ways:

- We propose the first algorithm for multi-agent patrolling under uncertainty and threats. Our formulation does not only capture the partially observable and non-stationary features of the dynamic environment, but also accounts for the health status of the patrolling agents.
- We design a predictive heuristic to estimate the value of each possible path from current position of the agent and provide an online algorithm to solve the patrolling problem for one agent at a time. Moreover, we propose a multi-agent algorithm that sequentially computes policies for individual agents. In particular, we also show that our multi-agent algorithm scale to larger environments (i.e., more than 10 agents) than existing solutions.
- We evaluate our algorithms in simulations and show that our algorithm outperforms a baseline algorithm up to 44% for 10 agents and by 21% for 15 agents.

The remainder of this paper is structured as follows. First, we review the literature on patrolling problems. We then present our model for the problem of multi-agent patrolling under uncertainty and threats. Given this, we formulate the single-agent patrolling problem as a POMDP and provide an algorithm that computes policies for individual agents. Finally, we propose our multi-agent algorithm and evaluate it in the simulations of multi-agent patrolling in a large environment.

Related Work

In this section, we review related work on agent based model and approaches for multi-agent patrolling problems.

In general, methods to gather situational awareness without considering threats are typically categorised as a class of *information gathering problem*[3], in which agents aim to continuously collect and provide up-to-date situational awareness. For these dynamic environments, previous work [3, 7, 8] consider fully observable (agents can directly observe the underlying state of the environment) stationary models (joint probability distribution of its states do not change when shifted in time). A partially observable model has been proposed in [12], where an agent can only perceive the exact state at its current position. Game-theoretic approaches [13–18] have focused on patrolling to guard important targets in the presence of strategic evaders or intruders; a problem that is characterised by (possibly multiple) attackers attempting to avoid capture or breach a perimeter. The agents' main challenge in such cases is to detect and capture these attackers in an effort to minimise loss. However, these approaches do not consider the health status of the agents and the damage that agents can suffer while patrolling.

Stationary models of the information/threats are considered in previous work. The work on information gathering in dynamic environments [8] have focused on specific environmental phenomena (e.g., monitoring algal bloom growth in lakes and salt concentration in rivers) rather than stochastic events as in our scenarios. Markov models are widely used to model non-stationary stochastic states in the world, such as the specific ground targets for aircraft [12, 19, 20] and sensors [21], physical activities in wireless network [22], and channel memory in communication systems [23, 24]. However, a number of strict assumptions are made in these works in terms of the Markov models used. For example, each target at each period can be in one of only two states [12, 23] and the matrix of the Markov models must satisfy some special formations [24].

Among these works, a *Markov Decision Process* (MDP) based algorithm that computes policies for individual agents has been proposed in [3] to solve continuous information gathering in fully observable environments. Our formulation in this paper mainly extends [3] to patrolling under threats in partially observable and on-stationary environments and cast the single-agent patrolling problem as a POMDP. However, solving this formulation using current

POMDP solvers [25] for all but the smallest instances is impossible due to the exponential growth in the number of possible paths of agents that can be traced in the environment and the number of the possible observations along each path. The POMCP algorithm has been proposed in [26] and has been shown to generate good solution quality and scale to large POMDPs. However, to the best of our knowledge, developing scalable approaches that extend POMCP to solve multi-agent POMDPs is still a open problem. As these possible benchmarks are unable to scale to multi-agent instances of our formulation, we design a baseline algorithm that greedily select the policy for one time step as a benchmark.

Methods

In this section, we present the model for the problem of multi-agent patrolling under uncertainty and threats. Specifically, we first model the physical environment in which the agents operate and then go on to describe the decision problem faced by the agents.

The Patrolling Problem

We formulate the patrolling problem by defining the physical environment and patrolling agents. In particular, we present the Markov models of information and threat at in the environment to capture the non-stationary feature.

The physical environment. The *physical environment* is defined by its spatial, temporal and dynamic properties. In particular, in the aftermath of a disaster, a number of specific sites might need urgent attention and access to these sites may be limited to certain areas (e.g., due to trees, debris, or natural obstacles). Hence, we can capture such features in terms of paths along which agents can travel from one disaster site to another. Specifically, the spatial property of the environment is encoded by a graph, which specifies how and where agents can move.

Definition 1 (Graph) *We model an area of the environment as an undirected graph $G = (V, E)$, where each vertex V representing spatial coordinates are embedded in Euclidean space and edges E encode the movements that are possible between them. Here, we denote $N = |V|$.*

In disaster response, each disaster site is a vertex in the graph, and a traversable area between a pair of sites is an edge of the graph.

Definition 2 (Time) *Time is modelled by a set of time steps $\{1,2,\ldots,T\}$ and at each time step $t \in \{1,2,\ldots,T\}$ the agents visit some sites in the environment.*

To capture the dynamic attributes of the environment, we assume that each vertex holds two states: one for information and one for threats.

Definition 3 (Information State Variable) *An information state variable indicates different levels of the information at a given vertex.*

For example, how many people need help and what is the status of a bridge are information state variables in disaster response scenario.

Definition 4 (Threat State Variable) *A threat state variable reflects the level of damage an agent suffers when visiting a given vertex.*

For example, the level of fire and the degree of smog are typical threat state variables in disaster response.

Definition 5 (Markov Model of Information and Threat) *The two state variables at each vertex change over time according to discrete-time multi-state Markov chains.*

To capture the transitions of the state variables, we employ a Markov chain model. Specifically, for a Markov chain with K states $S = (S_1, S_2, \ldots, S_K)$, the matrix of transition probabilities

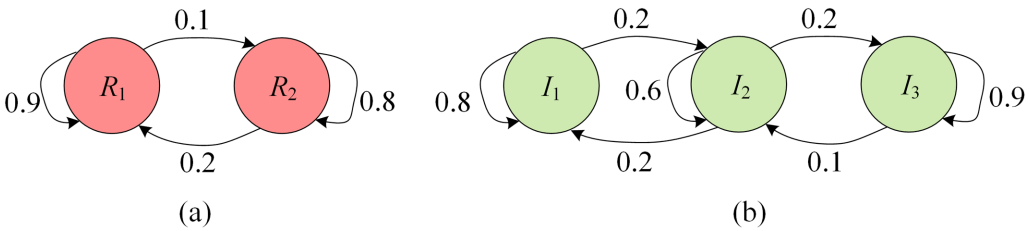

Fig 1. Example of information and threat models at a vertex. (a) A threat model with 2 states (i.e., R_1 and R_2) and (b) an information model with 3 states (i.e., I_1, I_2 and I_3), where the probabilities of each information/threat state changes to another over a time step are given (e.g., the probability of R_1 changes to R_2 is 0.1).

doi:10.1371/journal.pone.0130154.g001

for pairs of states is defined as:

$$P = \begin{bmatrix} p_{11} & \cdots & p_{1K} \\ p_{21} & \cdots & p_{2K} \\ \vdots & \ddots & \vdots \\ p_{K1} & \cdots & p_{KK} \end{bmatrix} = \begin{bmatrix} P_1 \\ P_2 \\ \vdots \\ P_K \end{bmatrix}$$

where p_{ij} is the probability that threat state S_i transitions to S_j in one time step and $S_i, S_j \in S$. An example of information and threat models at a vertex is shown in Fig 1. Thus, Fig 1(a) shows a threat model with 2 states (i.e., R_1 and R_2) and Fig 1(b) shows an information model with 3 states (i.e., I_1, I_2 and I_3), where the probabilities of each information/threat state changes over a time step are given (e.g., the probability of R_1 changes to R_2 is 0.1).

The set of information states $I^n = \{I_1^n, I_2^n, \ldots, I_{K_I^n}^n\}$ for location v_n correspond to an amount K_I^n of information which agents obtain when visiting v_n. The value of information is determined by the function f^n:$I^n \rightarrow \mathbb{R}^+$, and $f(I_k^n)$ increases monotonically with $k \in \{1, \ldots, K_I^n\}$, which indicates that the states of information are ordered in terms of their value. The information state at a given vertex independently evolves as a K_I^n-state Markov chain model with a matrix of transition probabilities P_I^n.

Similarly, the set of threat states $R^n = \{R_1^n, R_2^n, \ldots, R_{K_R^n}^n\}$ indicate the K_R^n threat levels of vertex $v_n \in V$. The "damage" that an agent suffers when visiting vertex v_n is captured by the function c^n:$R^n \rightarrow \mathbb{R}^+$, and $c(R_k^n)$ increases monotonically with $k \in \{1, \ldots, K_R^n\}$. The threat state at a given vertex independently evolves over time as a K_R^n-state Markov chain and the matrix of transition probabilities is P_R^n.

Having modelled the environment in which the agents operate, we next elaborate on the agents' goals.

Patrolling Agents. We define a *patrolling agent* (agent for short) as a physical mobile entity situated in the environment defined above, capable of gathering information, and maybe damaged by the threat when visiting a vertex. The set of all agents is denoted as $A = \{1,\ldots,|A|\}$. Then, the movement and visit capabilities of agents are formulated as follows. When patrolling in a graph G, each agent is positioned at a given vertex in G at each time step t. The movement of each agent is atomic, i.e., takes place within the interval between two subsequent time steps, and is constrained by G, i.e., agent m positioned at a vertex $v_i \in V$ can only move to a vertex $v_i' \in adj_G(v_i)$ that is adjacent to v_i in G. We assume that $\forall v_i \in V$, $v_i \in adj_G(v_i)$, i.e., an agent can

also stay at the same vertex. The speed of the agents is sufficient to reach an adjacent vertex within a single time step. Time can be discretised according to the speed of the UAVs. Thus if the UAVs can travel between sites in a five minutes, then a time step may be set at 5 minutes in the model.

Given this, an agent visits a vertex v_n when it is positioned at that vertex. On the one hand, a visit results in the agent being aware of the current information and threat state at v_n, such as I_i^n and R_j^n respectively. On the other hand, this agent obtains a reward $f^n(I_i^n)$ and suffers a loss $c^n(R_j^n)$ for a visit. The time it takes to visit a vertex is assumed to be negligible. We let $F^n = [f^n(I_1^n), \ldots, f^n(I_{K_I^n}^n)]$ denote the information value vector, where $f^n(I_k)$ is the information value that an agent could get if the information state is I_k^n (e.g., information at a vertex has 3 states and corresponds to 3 information values [0, 2, 5]). Similarly, we let $C^n = [c^n(R_1^n), \ldots, c^n(R_{K_R^n}^n)]$ denote as the damage value vector at vertex v_n, where $c^n(R_k^n)$ is the damage value that an agent will lose if the threat state is R_k^n (e.g., fire level at a position has 4 states which corresponds to 4 levels of damage [0, 4, 6, 10], and smog degree at a position has 3 states which corresponds to 4 levels of damage [0, 2, 5]). For each visit, the information at that vertex is obtained by the agent and we regard that the information state at a given vertex v_n will reset to I_1^n when an agent visits this vertex (I_1^n is the information state which means no new information was generated at v_n after last visit). As the states at each vertex change over time and agents can only access the exact states at the vertices that they visit, the patrolling environment can be considered non-stationary (i.e., joint probability distribution of its states may change when shifted in time) and partially observable.

Furthermore, in this paper, we make two assumptions about the communication and cooperation among agents as follows.

Assumption 1 *All the agents can share their collected observations with each other via communication. Such peer to peer communication is free of noise, costs, and delays.*

Consider a centralised station is organized to coordinate a team of UAVs for monitoring the continuously changing state of a disaster area, where each UAV can full communication with this station and Assumption 1 is satisfied in these domains. However, in some real scenarios, UAVs can only coordinate with each other using limited communication and decentralised approaches may be more appropriate (but this is beyond the scope of this paper and will be considered in future work).

Assumption 2 *When more than one agent is visiting a vertex, only one information value is obtained for the team but each agent suffers the same damage that may be generated at that vertex.*

This assumption is satisfied in the scenarios where the information gathering capability of one agent at a vertex is equal to that provided by multiple agents with the same sensors, and agents independently suffer the damage caused by threats. In future work, a model of information fusion for multiple (heterogeneous) agents will be considered. Thus, the team of agents need to coordinate with each other based on their observations while patrolling. Specifically, the goal of the agents is to gather as much information as possible while minimise the damage incurred.

We now provide a simple example to explain how the agents would operate in this scenario. Consider an agent that enters into a building on fire. In our setting, this is equivalent to the agent visiting a node in the graph. The fire level (threat state variable) and valuable information about victims and assets (information variable) changes over time. While exploring the building, the agent may acquire some information and suffer some damage due to the fire. At each time step, an agent selects one adjacent building to visit based on the estimated information value and the prior observation of threat states at each location. It then obtains a reward based

on the value of the information, and suffers a loss which is associated with the threat state. Then, the information state at the visited vertex is immediately reset.

Having defined the patrolling problem, we now need to plan the sequential patrolling actions for agents based on the history of actions and observations, and the model of the environment. Hence, in what follows, we first propose a POMDP formulation for single-agent patrolling within a graph and design an algorithm to solve it. Building upon this, we propose a scalable multi-agent patrolling algorithm.

Single-Agent Patrolling

In this section, we first formulate the POMDP based framework for single-agent patrolling problem. POMDPs imply that the agent does not know the exact state it is in, and the agent requires to keep track of each observation received, in order to maintain a probability distribution, known as the belief state, over the possible states [11]. Thus, we analysis that a standard representation of belief state makes the POMDP computational intractable and then present a compact representation of belief state for our POMDP formulation. Given this, we propose a predictive heuristic and an online single-agent algorithm.

The POMDP Framework. We now set up the single-agent patrolling problem as a POMDP $\langle \mathcal{S}, \mathcal{A}, \mathcal{O}, T, \Omega, r \rangle$ as follows:

- $\mathcal{S}$ is the set of states. A *state* is defined as a tuple $s = [v, (s_R^1, \ldots, s_R^N), (s_I^1, \ldots, s_I^N)] \in \mathcal{S}$, where v is the current position of the agent, $s_R^N \in R^n$ and $s_I^N \in I^n$ are the threat and information states at vertex $v_n \in V$. We denote $s_e = [(s_R^1, \ldots, s_R^N), (s_I^1, \ldots, s_I^N)] \in \mathcal{S}_e$, as the state that captures the information and threat states at all of the positions. Given this, the number of the states in $\mathcal{S}$ increases exponentially with the number the vertices.
- $\mathcal{A}$ is the set of all actions. The agent select an adjacent vertex to visit as an *action*.
- $\mathcal{O}$ is the set of observations. We define an *observation* $o = (v_i, s_I^i, s_R^i) \in \mathcal{O}$ as the current position v_i and the information and threat state at this position.
- $\mathcal{T}$ is the set of conditional transition probabilities. We assume that v is deterministic and only determined by the destination of movement of the agent. Based on the Markov models defined in the patrolling problem, s_e follows a discrete-time Markov process with $\prod_{n=1}^{N} K_R^n K_I^n$ states.
- Ω is the set of observation probabilities. As an observation o is directly a part of some states, the observation probability $\Omega(o|s', a) = 1$ if o is consistent with the corresponding part of s' and $\Omega(o|s', a) = 0$ otherwise.
- r:$\mathcal{A} \times \mathcal{O} \rightarrow \mathbb{R}$ is a reward function. $r(a, o)$ is the sum of the rewards obtained by the agent which associates to the action a and observation o:

$$r(a, o) = \alpha f^i(s_I^i) - (1 - \alpha) c^i(s_R^i) \tag{1}$$

  where α is a weight parameter of the two objectives.

The objective of the agent is then to choose the movement actions sequentially to maximize the total expected reward accumulated over T steps.

In this model, the states are not directly observable. Hence, a standard belief vector $B(t) = [b_1(t), \ldots, b_M(t)]$ is defined as the posterior probability distribution over the possible states S, where $b_m(t)$ is the conditional probability that the environment state is at the mth state at the current time step t. For any t, it has been shown in [27] that this belief vector is a sufficient

statistic for the design of the optimal action for each time step. A policy π specifies the action that will be executed in any given belief state and the optimal policy π^* is a policy by which the agent gets the maximum total expected reward accumulated over T steps. However, as each environment state is an joint state of the information and threat states at all of the vertices, the number of possible states S that defined in our POMDP is $\prod_{n=1}^{N} K_R^n K_I^n$, which increases exponentially with the number the vertices. Moreover, as the belief vector is defined as the posterior probability distribution over these possible states, the dimension of this belief vector also increases exponentially with the number the vertices.

To address this, we propose an online method by introducing a belief vector of reduced dimension and develop a predictive heuristic to reduce the search space and still produce high quality solutions (as we show later).

Compact Belief Representation. As the threat state and information state variables at each vertex evolve independently and v is deterministic, we can find a sufficient statistic for the optimal policy whose dimension linearly grows with N, similar to [23, 24]. We introduce a compact representation of belief state and its transition function in this section.

We define a sufficient statistic belief vector of the environment states at time t as the vector of the conditional probabilities (conditioned on the observation and decision history) $\Psi(t) = [\Psi_R(t), \Psi_I(t)]$, where $\Psi_R(t)$ is defined as:

$$\begin{cases} \Psi_R(t) &= (w_R^1(t), \ldots, w_R^N(t)) \\ w_R^n(t) &= (w_{R1}^n(t), \ldots, w_{RK_R^n}^n(t)) \end{cases} \tag{2}$$

where $w_{Rk_1}^n(t)$ is the probability that the threat state at vertex v_n is $R_{k_1}^n$, $k_1 = 1, \ldots, K_R^n$ and $\Psi_I(t)$ is defined as:

$$\begin{cases} \Psi_I(t) &= (w_I^1(t), \ldots, w_I^N(t)) \\ w_I^n(t) &= (w_{I1}^n(t), \ldots, w_{IK_I^n}^n(t)) \end{cases} \tag{3}$$

where $w_{Ik_2}^n(t)$ is the probability that the information state at vertex v_n is $I_{k_2}^n$ and $k_2 = 1, \ldots, K_I^n$. Then $\Psi(t)$ is a sufficient statistic of optimal decision making [23, 24]. By exploiting the statistical independence among vertices, we reduce the dimension of the sufficient statistic from $\prod_{n=1}^{N} K_R^n K_I^n$ to $\Sigma_{n=1}^{N}(K_R^n + K_I^n)$, which grows with N linearly. This allows us to reduce the computational and storage complexity of the optimal patrolling policy from exponential to linear.

Theorem 1 *For any time t, $\Psi(t)$ is a sufficient statistic for the design of optimal policy for our POMDP formulation.*

Proof We show that when the information and threat at the N vertices evolve independently, each element $b_m(t)$ in the standard belief vector $B(t)$ can be obtained from $\Psi(t)$, where $b_m(t)$ is the conditional probability that the environment state is at the ith state. Without loss of generality, we consider $N = 2$. Let $\mathcal{I}(t)$ denote the history up to the beginning of slot t. Let τ_n denote the most recent time instant when vertex v_n is visited. We can thus write an entry of $b_m(t)$ as in

Eq (4). Quantities in Eq (4) are entries of $\Psi(t)$. Hence, $\Psi(t)$ is a sufficient statistics.

$$\begin{aligned} & \Pr[s_R^1(t)=i', s_R^1(t)=i'', s_I^1(t)=j', s_I^2(t)=j'' \mid \mathcal{I}(t)] \\ = \quad & \Pr[s_R^1(t)=i', s_R^2(t)=i'', s_I^1(t)=j', s_I^2(t)=j'' \mid s_R^1(\tau_1)=o_R', s_R^2(\tau_2)=o_R'', \tau_1, \tau_2] \\ = \quad & \Pr[s_R^1(t)=i' \mid s_R^1(\tau_1)=o_R'] \Pr[s_R^2(t)=i'' \mid s_R^2(\tau_2)=o_R''] \\ & \Pr[s_I^1(t)=j' \mid \tau_1] \Pr[s_I^2(t)=j'' \mid \tau_2] \end{aligned} \tag{4}$$

Initially, we assume that we have probabilistic information about the state of each of the N vertices $\Psi(0) = [\Psi_R(0), \Psi_I(0)]$. Then, the elements of belief vector $\Psi(t)$ are updated to $\Psi(t+1)$ upon action $a = v_i$ and observation $o = (v_i, s_I^i, s_R^i)$ as:

$$\begin{aligned} w_R^n(t+1) &= \begin{cases} \tilde{I}_k & \text{if } v_n = v_i, s_R^i(t) = R_k \\ w_R^n(t) P_R^n & \text{if } v_n \neq v_i \end{cases} \\ w_I^n(t+1) &= \begin{cases} \tilde{I}_1 & \text{if } v_n = v_i \\ w_I^n(t) P_I^n & \text{if } v_n \neq v_i \end{cases} \end{aligned} \tag{5}$$

where $\forall v_n \in V$, $R_k^n \in R^n$, $I_k^n \in I^n$, and $\tilde{I}_k$ is a unit vector with the k^{th} item is 1, P_R^n and P_I^n are respectively the matrices of transition probabilities of threat and information at position v_n. As shown in Eq (5), the threat belief vector $w_R^n(t+1)$ at one vertex v_n that some agent is visiting is updated to $\tilde{I}_k$ based on the observation $R_k^n(h)$ at this vertex, while $\mathbf{w}_R^n(t+1)$ at some other vertex that no agent is visiting is updated by the current threat belief vectors $\mathbf{w}_R^n(t)$ and threat Markov model P_R^n at this vertex. A similar explanation holds for the update to the information belief $w_I^n(t+1)$, as for $v_n \in V$.

Based on the transition function above, a policy π specifies a sequence of actions $\pi = [\pi(1), \pi(2), \ldots]$, where $\pi(t)$ is the position selected to visit at time t. Given this, the optimal policy can be computed as:

$$\pi^* = \arg\max_{\pi} \mathbb{E}^{\pi} \left[\sum_{t=1}^{\infty} \gamma^t \mathcal{R}^{\pi(t)}(\Psi(t) | \Psi(0)) \right] \tag{6}$$

where $\mathcal{R}^{\pi(t)}(\Psi(t))$ is the reward obtained when the belief state is $\Psi(t)$ and $\gamma \in [0, 1]$ is the discount factor.

Although the dimensionality of the belief state is reduced, the problem is still a POMDP and finding the optimal solution is intractable. Based on this reduced belief vector, we next develop a predictive heuristic and present the online single-agent algorithm that implements this heuristic.

The Predictive Heuristic. In order to develop a predictive heuristic for online policy selection, we first introduce the assumption that the Markov state transition matrices are monotone matrices, which means that the higher the information/damage value of the vertex's current state the higher is the likelihood that the next state of this vertex will be of high information/damage value. Then, we show how to define the predictive heuristic as the predictive expected future reward based on the monotonicity of the transition matrices.

Stochastic dominance is a central theme in a wide variety of applications in economics, finance and statistics [28]. Similar assumption has been made to model the states of the channels in communication systems [23, 24] and the states at targets for UAVs monitoring [12].

Stochastic dominance $\succ$ between two Z dimension probability vector x, y is defined as $x \succ y$, if:

$$\sum_{j=i}^{Z} x(j) \geq \sum_{j=i}^{Z} y(j), \quad \text{for } i = 2, 3, \ldots, Z \tag{7}$$

We assume that the Markov information model and Markov threat model are monotonic matrices, i.e., the matrix of transition probabilities P_R^n and P_I^n satisfies:

$$\begin{aligned} P^n_{RK_1} &\succ P^n_{RK_1-1} \succ \ldots \succ P^n_{R_1} \\ P^n_{IK_2} &\succ P^n_{IK_2-1} \succ \ldots \succ P^n_{I_1} \end{aligned} \tag{8}$$

If the matrix of transition probabilities P_R^n and P_I^n satisfy the assumption above, then P_R^n and P_I^n are *monotone matrices*[29]. Under this assumption, the higher the information value of the state of the current vertex the higher is the likelihood that the next state of this vertex will be of high information value, i.e., if $w_I^n(t) \succ w_I^{n'}(t)$, then $w_I^n(t)P_I^n \succ w_I^{n'}(t)P_I^{n'}$. From (5), we know that probability vectors for information states of two vertices keep the relationship of stochastic dominance when no agent visits any of them. Obviously, if $w_I^n(t) \succ w_I^{n'}(t)$, then $w_I^n(t)F^n \geq w_I^{n'}(t)F^{n'}$, which means that a stochastically dominant information belief vector is likely to have a higher information value. The same is true that a stochastically dominant threat belief vector is likely to have a higher damage value. In particular, as the information state at a given vertex will reset to I_1 when there is an agent visiting this vertex, the belief vector of information states (1,0,. . .,0) is stochastically dominated by the belief vector of any vertex which is not being visited, so the more recently visited vertex always has a lower expected information value.

To note, our monotonicity assumption is not a constraint that makes the information value (or the damage of the threat) increasing with time, but a model that the probability vector of the information (or threat) transition matrices satisfy the feature of stochastic dominance. We now provide a example of a 4-state Markov threat model at a vertex as follows:

$$P_R = \begin{bmatrix} 0.8 & 0.1 & 0.1 & 0 \\ 0.4 & 0.5 & 0.0 & 0.1 \\ 0.2 & 0.1 & 0.6 & 0.1 \\ 0 & 0.0 & 0.4 & 0.6 \end{bmatrix}$$

It can be seen that the Vectors of P_R satisfy the condition of Eq (8), i.e., $P_{R4} \succ P_{R3} \succ P_{R2} \succ P_{R1}$, where for $P_{R3} \succ P_{R2}$ as an example, the elements of P_{R2} and P_{R3} match the condition for stochastic dominance of Eq (7) as:

$$0.1 + 0.6 + 0.1 \geqq 0.5 + 0.0 + 0.1$$

$$0.6 + 0.1 \geqq 0.0 + 0.1$$

$$0.1 \geqq 0.1$$

For example, if the threat states at vertices v_1 and v_2 are respectively $w_R^1 = [0.1, 0.2, 0.5, 0.2]$ and $w_R^2 = [0.2, 0.4, 0.3, 0.1]$, i.e., $w_R^1 \succ w_R^2$. Then, v_1 is likely to have a higher next threat state than v_2. However, after a time step, it is possible that any threat state may switch to not only a higher state, but also a lower state.

Then given the monotonicity assumption, we can use the relationship between the belief states at different vertices in order to "predict" the belief state at an unvisited node. Hence, we can estimate the expected reward agents may get from one vertex of the graph when visiting it at a near future step. We denote a feasible policy of length D at time t as $\pi_D(t) = (\pi_{t+1},\ldots,\pi_{t+D})$, which consists of D consecutive deterministic vertices/actions.

Here, we define the predictive heuristic as the predictive expected future reward $\mathbb{E}[\hat{\mathcal{R}}(\pi_D(t))]$ for policy $\pi_D(t)$, which is the aggregate of the expected reward of each step in $\pi_D(t)$ as:

$$\mathbb{E}[\hat{\mathcal{R}}(\pi_D(t))] = \sum_{i=1}^{D} \gamma^t (\alpha \hat{w}_I^{\pi_{t+i}}(t+i) F^{\pi_{t+i}} - (1-\alpha)\hat{w}_R^{\pi_{t+i}}(t+i) L^{\pi_{t+i}}) \tag{9}$$

where, $[\hat{w}_I^{\pi_{t+i}}(t+i), \hat{w}_R^{\pi_{t+i}}(t+i)]$ is the predictive belief vector at the vertex π_{t+i} and time $t+i$. For the step $t+1$, we can get the predictive belief vector $[\hat{w}_I^{\pi_{t+1}}(t+1), \hat{w}_R^{\pi_{t+1}}(t+1)]$ by the current belief vector $\Psi(t)$, current action $a(t)$ and observations $\theta(t)$, i.e. $\Psi(t+1) = \delta\big(\Psi(t) \mid a_t^*, \theta(t)\big)$, which is the belief vector at $t+1$ and obtained from Eq (5). For $\{t+2,\ldots,t+D\}$, we get the predicted belief vector based on a transition which omits observations in Eq (5) as follows:

$$\begin{aligned} \hat{w}_R^n(\tau+1) &= \hat{w}_R^n(\tau) P_R^n \\ \hat{w}_I^n(\tau+1) &= \begin{cases} P_{I1}^n & \text{if } v_n = \pi_\tau \\ \hat{w}_I^n(\tau) P_I^n & \text{if } v_n \neq \pi_\tau \end{cases} \end{aligned} \tag{10}$$

where $\tau = \{t+1,\ldots,t+D-1\}$.

Given the predictive heuristic and policies that looks ahead D time periods, the agent compares all feasible paths of length D and chooses the next location to visit according to the path that gives the highest predictive expected reward gained over that path. The details of how to use the heuristic in our online single-agent algorithm is presented in the next section.

The Online Algorithm. Based on the predictive heuristic, we propose an online algorithm for single-agent patrolling problem (Algorithm 1) in this section.

Algorithm 1 Single-Agent Patrolling

Require: $\{P_R^n\}$: the Markov threat models
Require: $\{P_I^n\}$: the Markov information models
Require: $\Psi(t)$: the belief state of current time step
Require: $o(t)$: the observation at the current position
Require: $v(t)$: current position.
Ensure: $a^*(t+1)$: next action of the agent
▷ *Step 0: get all feasible policies* $\Pi_D(t)$;
▷ *Step 1: computing best policy:*
for $\pi_D(t) \in \Pi_D(t)$ **do**
▷ *Step 1.1: Get predictive belief state for next D steps:*
  $\Psi(t+1) \leftarrow \delta(\Psi(t) | v_t, \theta(t))$
  for $\tau \in \{t+1,\ldots,t+D-1\}$ **do**
    for $v_n \in V$ **do**
      $\hat{w}^n(\tau+1) \leftarrow \hat{\delta}(\hat{w}^n(\tau) \mid \pi_\tau(\tau))$
    end for
  end for
▷ *Step 1.2: Compute the predictive reward for* $\pi_D(t)$:
  $\mathbb{E}[\hat{\mathcal{R}}(\pi_D(t))] = \alpha w_I^{\pi_{t+i}}(t+i) F^{\pi_{t+i}} + \beta w_R^{\pi_{t+i}}(t+i) L^{\pi_{t+i}}$

▷ *Step 1.3: Compare $\pi_D(t)$ with the stored best policy:*
9: **if** $\mathbb{E}[\hat{\mathcal{R}}(\pi_D(t))] > \mathbb{E}[\hat{\mathcal{R}}(\pi^*_D(t))]$ **then**
10: $\pi^*_D(t) \leftarrow \pi_D(t)$
11: **end if**
12: **end for**
▷ *Step 2: return the next action from the best policy π^*_i*
13: **return** $a^*(t+1) \leftarrow \pi^*_{t+1}$

First, we compute $\Pi_D(t)$, which is the set of all the feasible policies that start from current position $v(t)$ (*step* 0), where we name the parameter D as the *maximum horizon*, i.e. the number of horizons we look ahead in the POMDP. Then, we compute the predictive expected reward for all the policies. For each policy, the belief state at t+1 is updated by the belief state, position and observations at t by Eq (5) (line 2) and the predictive belief state at {t+2,. . ., t+D} is computed by Eq (10) (line 3–7). Given this, we compute the predictive reward for the policy (line 8). Thus, the best policy is:

$$\pi^*_D(t) = \arg\max_{\pi_D(t)} \mathbb{E}[\hat{\mathcal{R}}(\pi_D(t))] \tag{11}$$

The best next action here is computed as $a^*(t+1) = \pi^*_{t+1}$, which is the first action of best policy (line 13).

Having defined the online single-agent algorithm for our formulation of patrolling under uncertainty and threats, we extend it to compute policies for multi-agent problems next.

Multi-Agent Patrolling

For multi-agent patrolling, we assume all the agents can share their collected observations with each other with full communication. Thus, team of agents may not only obtain more information about the environment, but each agent may also make decisions given observations are shared by other agents. Given this, we formulate the multi-agent patrolling problem as a *Multi-agent POMDP* (MPOMDP) and design an scalable online multi-agent algorithm to coordinate the actions of agents in their patrolling tasks.

A MPOMDP with complete communication can be reduced to a POMDP with a single centralised controller that takes joint actions and receives joint observations [30]. We now set up our problem of multi-agent patrolling in a graph as a POMDP $\langle \mathcal{M}, \mathcal{S}, \mathcal{A}, \mathcal{O}, T, \Omega, r \rangle$ as follows.

- $\mathcal{M}$ is the set of the agents.
- $\mathcal{S}$ is the set of states. A *state* is defined as a tuple $\vec{s} = [\vec{v}, (s^1_R, \ldots, s^N_R), (s^1_I, \ldots, s^N_I)] \in \mathcal{S}$, where $\vec{v}$ is the current positions of agents, $s^N_R \in R^n$ and $s^N_I \in I^n$ are the threat and information states at vertex $v_n \in V$. We denote $\vec{s_e}$=$[(s^1_R, \ldots, s^N_R), (s^1_I, \ldots, s^N_I)] \in \mathcal{S}_e$, as the state that captures the information and threat states at each position.
- $\mathcal{A}$ is the set of all joint actions. The agents select adjacent vertices to visit as an *joint action*.
- $\mathcal{O}$ is the set of joint observations. For current positions of the agents and the information and threat states of their current positions, we define a *joint observation* $o = \{\vec{v}, \{o^i | \forall v_i \in \vec{v}\}\} \in \mathcal{O}$, where $o^i = (s^i_R, s^i_I)$ is the observation of agent i.
- $\mathcal{T}$ is the set of conditional transition probabilities. We assume that $\vec{v}$ is deterministic and only determined by the destinations of the joint movement of agents. $\vec{s_e}$ follows a discrete-time Markov process with $\prod_{n=1}^{N} K^n_R K^n_I$ states.

- Ω is the set of observation probabilities. As an observation $\vec{o}$ is directly a part of some states, the observation probability $\Omega(\vec{o}|\vec{s}', \vec{a}) = 1$ if $\vec{o}$ is consistent with the corresponding part of $\vec{s}'$ and $\Omega(\vec{o}|\vec{s}', \vec{a}) = 0$ otherwise.
- r:$\mathcal{A} \times \mathcal{O} \rightarrow \mathbb{R}$ is a reward function. $r(\vec{a}, \vec{o})$ is the sum of the reward obtained by the agents which associates to the joint action $\vec{a}$ and observation $\vec{o}$:

$$r(\vec{a}, \vec{o}) = \sum_{v_i \in \vec{v}} \left(\alpha \frac{1}{n_{v_i}} f^i(s_I^i) - (1-\alpha) c^i(s_R^i) \right)$$

  where n_{v_i} is the number of agents who are visiting v_i.

The objective of the agents is then to choose the movement actions sequentially to maximize the total expected reward accumulated over T steps.

Then, we note that, while the state variable described in Eqs (2) and (3) can be used to express the belief vector of the environment states for a multi-agent POMDP, the joint action space of the POMDP is the Cartesian product of the action and observation spaces of the individual agents. However, in so doing, the size of the joint action space and joint observation space grows exponentially with the number of agents $|\mathcal{M}|$, allowing only the smallest of problem instances to be solved. Instead, sequentially computing policies for individual agents as in our multi-agent algorithm avoids this problem of computing a joint policy for the team at the expense of solution quality. However, a bounded optimal of this multi-agent algorithm is guaranteed (we analyse this later).

Similar methods have been successfully used to solve multi-agent problems [3, 8]. As these formulations are different from our partially observable scenarios, a straightforward application of their methods is not possible. Hence, we consider how to sequentially compute policies for individual agents in partially observable problem using our online single-agent algorithm.

When sequentially computing policies for individual agents using our predictive heuristic, there implicitly exists an order in which the agents make actions; agent 1 completes D step actions of its best policy, agent 2 second, etc.. The expected future reward of a policy $\pi_D^i(t)$ of agent i is conditioned on position $v_i(t)$, belief vector $\Psi(t)$ and the best policies of the previously computed policies of agents $\mathcal{M}_{-i} = \{1, \ldots, i-1\}$.

The best online patrolling policy for agent i in a multi-agent setting is recursively defined as:

$$\begin{aligned} \hat{\pi}_1^* &= \arg\max_{\hat{\pi}_1} \mathcal{R}'(v_1(t), \Psi(t)) \\ \hat{\pi}_2^* &= \arg\max_{\hat{\pi}_2} \mathcal{R}'(v_2(t), \Psi(t), \hat{\pi}_1^*) \\ &\vdots \\ \hat{\pi}_i^* &= \arg\max_{\hat{\pi}_i} \mathcal{R}'(v_i(t), \Psi(t), \hat{\pi}_1^*, \ldots, \hat{\pi}_{i-1}^*) \end{aligned} \tag{12}$$

where we use $\hat{\pi}_i^*$ denotes the best policy of agent i.

To ensure the reward function only takes into account the marginal reward value, we need to exclude double counting. There are two types of double counting. First, *synchronous double* counting, which occurs when two agents patrol the same cluster within the same time step. In this case the reward for patrolling the vertex is received twice. Second, *asynchronous double counting*, which occurs when agent i decides to visit vertex v_n at t_1, and there was an action of visiting v_n by agent j ($j < i$) at t_2 ($t_1 < t_2$) during the D horizon i.e., the agent j will visit vertex v_n after agent i. For the situation where agent j visits vertex v_n before agent i (i.e. $t_1 \geq t_2$), it has been accounted when calculating $\mathbb{E}[\hat{\mathcal{R}}(\pi_D(t))]$ in Eq (9).

Here, we show how to deal with the asynchronous double counting, i.e., agent i decides to visit vertex v_n at t_1 and there was an action of visiting v_n by agent j ($j < i$) at t_2 ($t_1 < t_2$) during the D horizon. Without loss of generality, we consider the situation that only v_n in $\pi^i_D(t)$ of agent i has been visited by agent j. If more than one agent of $\mathcal{M}_{-i} = \{1,\ldots, i-1\}$ has an action to visit v_n, we assume the time t_2 is nearest to t_1 (only the nearest one needs to be taken into account and this can be deduced from the transition Eq (10)). Based on this assumption, we can see that the expected information reward of agent j for visiting vertex v_n is overestimated, as it is unaware that the i will reset the information at the time t_1. Thus, we introduce a penalty $\hat{p} \in \mathbb{R}^+$ for agent i that compensates for the reduction of reward of agent j, as follows:

$$\mathcal{R}'_i(v_i(t), \Psi(t), \hat{\pi}^*_1, \ldots, \hat{\pi}^*_{i-1}) = \mathbb{E}[\hat{\mathcal{R}}(\pi_D(t))] - \hat{p} \tag{13}$$

where $\mathbb{E}[\hat{\mathcal{R}}(\pi_D(t))]$ is the expected reward function defined in Eq (9), and $\hat{p}$ is the loss incurred by agent j that will visit the vertex v_n after i, which is defined as follows:

$$\hat{p} = \hat{r}_{\text{expected}} - \hat{r}_{\text{revised}} \tag{14}$$

where the $\hat{r}_{\text{expected}} \in \mathbb{R}^+$ is the expected reward that agent j computes for visiting vertex v_n and the $\hat{r}_{\text{revised}}$ is the revised expected reward of agent j visiting vertex v_n as computed by agent i considering only its action. We define the *revised expected belief states* at vertex v_n and between time $[t_1{+}1,\ldots, t_2]$ are $\{\tilde{w}^n(t_1+1), \ldots, \tilde{w}^n(t_2)\}$, which are obtained by the transition Eq (10) based on the predictive belief state $\hat{w}^n(t_1)$ and action $a(t_1) = v_n$. Then the *revised expected reward* is as follows:

$$\hat{r}_{\text{revised}} = \gamma^{t_2}(\alpha \tilde{w}^n_I(t_2)F^n - (1-\alpha)\tilde{w}^n_R(t_2)L^n) \tag{15}$$

Now, using the algorithm to compute the policy of length D as before, we obtain an action for each individual agent. A team action is formed by combining these individual actions. This team action is not optimal, as the policy of agent i is computed greedily with respect to the policies of agents $\mathcal{M}_{-i}$. However, we can still bound the the performance guarantees compared with the policy obtained by searching the joint action space.

We use the theorem from [31] to obtain a bound on the value of the greedily selected policies:

Theorem 2 *Let $g{:}2^E \rightarrow \mathbb{R}$ be a non-decreasing sub-modular set function. The greedy algorithm that iteratively selects the element $e \in E$ that has the highest incremental value with respect to the previously chosen elements $I \in E$:*

$$e = \arg\max_{e \in E \setminus I} g(e \cup I) - g(I) \tag{16}$$

until the resulting set I has the desired cardinality k, has an approximation bound $\frac{g(I_G)}{g(I^*)}$ at least $1 - \left(\frac{k-1}{k}\right)^k$, where $I^* \in E$ is the optimal subset of cardinality k that maximises f.

For the number of agents $|\mathcal{M}|$ in our formulation, the approximation bound of the greedy algorithm is $1 - \left(\frac{|\mathcal{M}|-1}{|\mathcal{M}|}\right)^{|\mathcal{M}|}$. It has been shown in [3] that this approximation bound is monotonically decreasing with $|\mathcal{M}|$, and thus as, for $|\mathcal{M}| \rightarrow \infty$, the multi-agent policy yields at least $\approx 63\%$ of the reward obtained using the best policy that searches the joint policy space for $|\mathcal{M}|$ agents.

Having formulated the problem and designed both single-agent and multi-agent algorithms, we will evaluate our methods in the next section.

Empirical Evaluation

To empirically evaluate our approach, we applied it to 10 and 15 agents continuously patrolling in a large graph, which contains 350 vertices and 529 edges. The online computing time limit is 0.5s because agents must decide which locations to visit at each time step within that time limit. As the single-agent algorithm in the paper can be seen as a special case of the multi-agent algorithm, we just present the results of the multi-agent algorithm here. In the aforementioned graph, we simulated two scenarios:

- **Scenario A:** we use the same Markov information and threat models for every vertex in the graph;
- **Scenario B:** we apply 3 different information and threat models to different vertices in the graph.

Notice that for Scenario A the information and threat models at different locations are homogeneous. However, as these information and/or threat are non-stationary, the information/or threat state are various among these locations. We use Scenario A aiming to capture the situation where the locations in the environment hold same types of information and threat. For Scenario B, the information and threat models at different locations are heterogeneous, i.e., different locations in the environment may hold various types of information and threat.

We set the parameters in reward function (i.e., Eq 1) and value function (i.e., Eq 6) as: the weight parameter $\alpha = 0.33$ and the discount factor $\gamma = 0.9$. More specifically, in Scenario A, we define the two Markov chains as follow:

$$P_R = \begin{bmatrix} 0.9 & 0.1 & 0 \\ 0.4 & 0.4 & 0.2 \\ 0.0 & 0.2 & 0.8 \end{bmatrix}$$

$$P_I = \begin{bmatrix} 0.8 & 0.1 & 0.1 & 0 & 0 \\ 0.2 & 0.7 & 0.0 & 0.1 & 0 \\ 0.1 & 0.1 & 0.7 & 0.1 & 0 \\ 0 & 0.0 & 0.1 & 0.8 & 0.1 \\ 0 & 0 & 0.0 & 0.1 & 0.9 \end{bmatrix}$$

Here, the transition function P_R and P_I satisfies monotonic assumption of Eq (7). For example, the first two rows in P_R satisfy the constraints in Eq (8) that $0.9 \geq 0.4$, $0.9+0.1 \geq 0.4+0.4$ and $0.9+0.1+0.0 \geq 0.4+0.4+0.2$. The information and threat value vectors are respectively set as $F = [0\ 1\ 2\ 3\ 4]$ and $L = [0\ 1\ 2]$. In Scenario B, we attribute several different Markov models to different vertices. A problem of 15 agents patrolling is shown in Fig 2, where the size of the circle of each location denotes the absolute value of instance reward of each vertex, the colour denotes its sign (black is positive and red is negative), the green circles are agents' current locations and "R" and "r" in lower right of each vertex denotes the threat state of this vertex is "2" and "1" respectively.

For standard POMDP solvers such as POMCP, the size of the joint action space and joint observation space grow exponentially with the number of agents, which makes them are intractable for our multi-agent patrolling problem with large number of actions and observations.

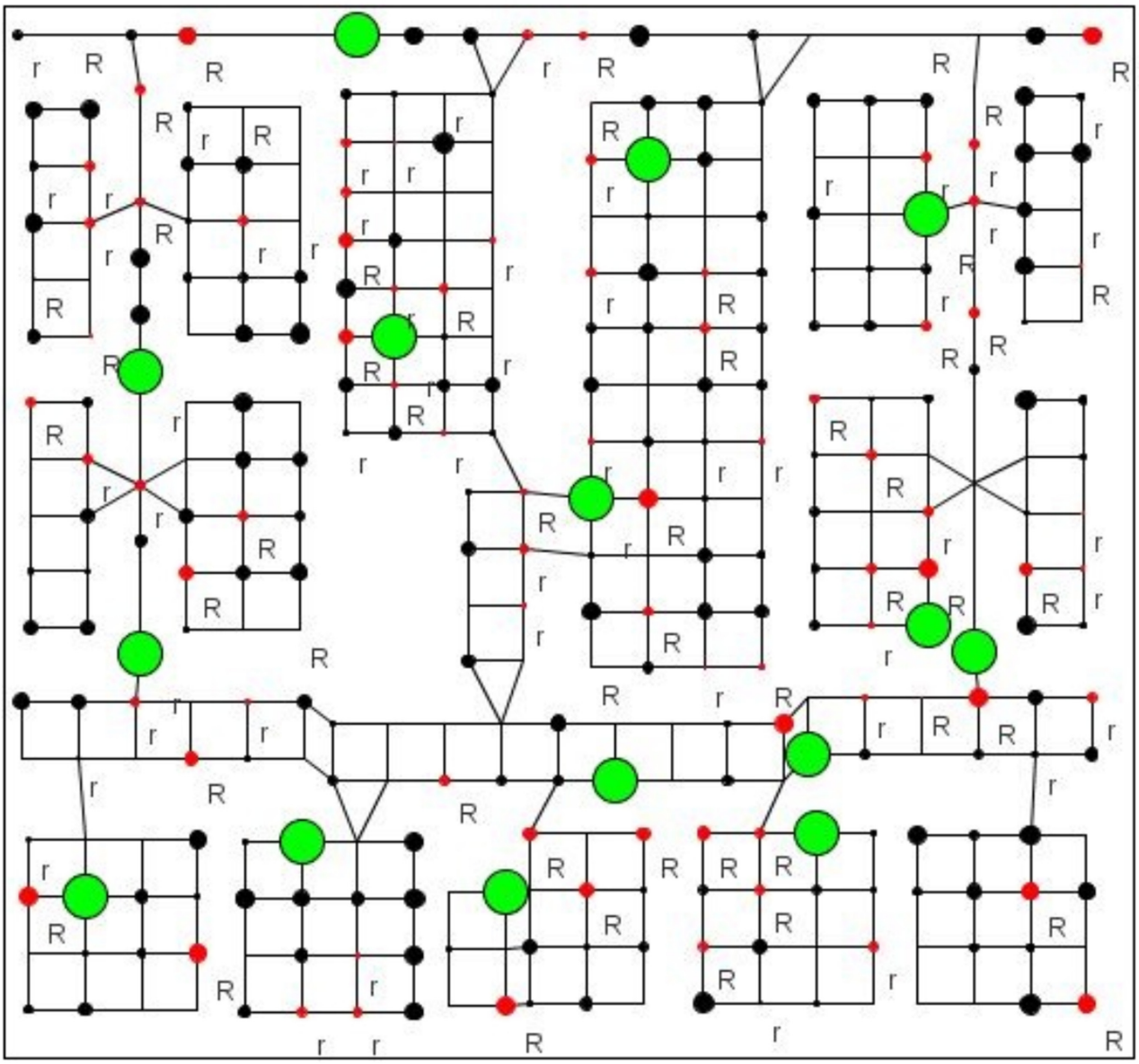

Fig 2. Scenario of 15 agents patrolling.

doi:10.1371/journal.pone.0130154.g002

Hence, we benchmark against a random algorithm (Random) and a baseline algorithm (Baseline), and measure the total reward of the information value and the damage suffered of agents using them. Specifically,

- **Random** moves the agents to a random location adjacent to the agents' current position.
- **Baseline** moves the agents to the adjacent location with the highest value in the next step. We assume the baseline algorithm sequentially computes policies for individual agents to avoid different agents selecting the same vertex, which is similar to PH-1.
- **PH-D** is our multi-agent patrolling algorithm, where D is the maximum horizon, i.e. the number of horizons we look ahead. We adjust maximum horizon D from the set {2,4,8} to

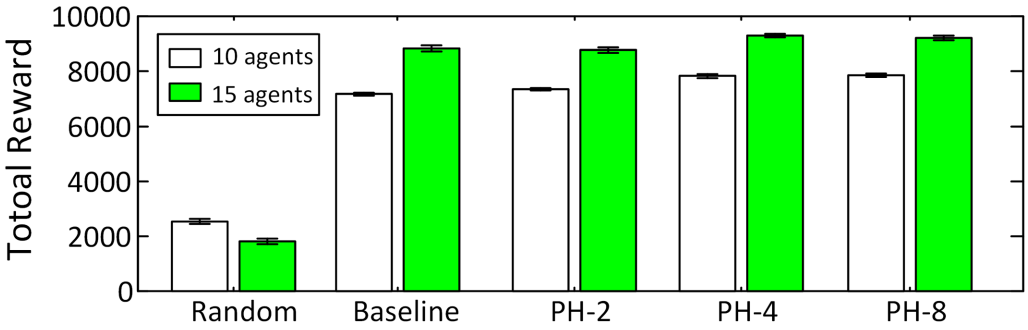

Fig 3. Rewards in Scenario A.

doi:10.1371/journal.pone.0130154.g003

investigate the extra computation involved for higher values of maximum horizon. We illustrated the results of our algorithms of different maximum horizon.

The initial locations of the agents are randomly distributed in the graph. Agents patrol continuously for 3000 time steps in the stochastically changing graph. For each scenario and each algorithm we ran 1000 rounds and plotted the results in Figs 3 and 4 where the error bars depict the 95% confidence intervals around the means. Non-overlapping error bars invalidate the null hypothesis with $\alpha = 0.05$. In both scenarios, Random performs poorly and its total reward never reaches more than 30% of the reward obtained by the other two algorithms. In Scenario A, both PH-8 and Baseline perform well, and PH-8 outperforms than baseline algorithm by at least 5%. However, for the graph with different Markov models in Scenario B, our algorithm is significantly better than all the other algorithms, and PH-8 outperforms the baseline algorithm by more than 44% for 10 agents and by 21% for 15 agents. In addition, with different maximum horizon D from {2,4,8}, the reward obtained by PH-D increases with D as well as its computation time increases with D exponentially. For $D > 8$, the computing time for each step is out of our time limit for online decision making. Thus, we can conclude that the use of our predictive

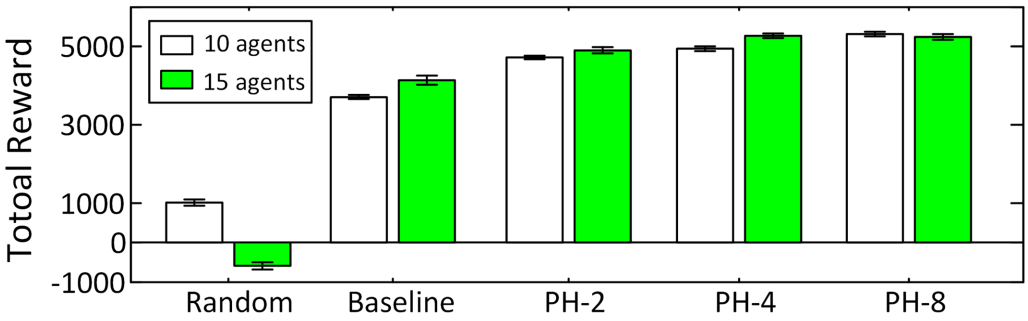

Fig 4. Rewards in Scenario B.

doi:10.1371/journal.pone.0130154.g004

heuristic in Ph-*D* has a significant impact on performance and that *D* can be adjusted to trade-off between quality and computation time while still outperforming baseline algorithms.

Conclusion

In this paper, we developed an online multi-agent patrolling algorithm for large partial observable and stochastic environment where the information are distributed with threats. Specifically, a predictive heuristic is defined to evaluate the policies of looking ahead several steps. For the multi-agent algorithm, we extended the sequential policy computation method for individual agents to deal with partially observable problems. We empirically showed that for 10 agents in a large graph, our algorithm outperforms the baseline algorithm by more than 44%. In our future work, on the one hand, as this is the first algorithm for patrolling with uncertainty and threats, we plan to devise a better heuristic and algorithms that provide theoretical performance guarantees in our future work. One the other hand, as our formulation is a basic model of UAVs patrolling under uncertainty and threats, we will consider that the communication system of the agents may locally break down by suffering from harms or some agents may get destroyed due to cumulated harms.

Supporting Information

S1 Video. Video to show the simulation of 15 agents patrolling problem.
(MP4)

S1 Code. Java code to implement the scenarios and computations in the paper.
(ZIP)

Author Contributions

Conceived and designed the experiments: SC LS SDR. Performed the experiments: SC FW SDR. Analyzed the data: SC JC SDR. Contributed reagents/materials/analysis tools: FW. Wrote the paper: SC FW LS JC SDR.

References

1. Maza I, Caballero F, Capitán J, Martínez-de Dios J, Ollero A (2011) Experimental results in multi-uav coordination for disaster management and civil security applications. Journal of intelligent & robotic systems 61: 563–585. doi: 10.1007/s10846-010-9497-5
2. Delle Fave FM, Rogers A, Xu Z, Sukkarieh S, Jennings NR (2012) Deploying the max-sum algorithm for decentralised coordination and task allocation of unmanned aerial vehicles for live aerial imagery collection. In: 2012 IEEE International Conference on Robotics and Automation. pp. 469–476.
3. Stranders R, de Cote EM, Rogers A, Jennings NR (2013) Near-optimal continuous patrolling with teams of mobile information gathering agents. Artif Intell 195: 63–105. doi: 10.1016/j.artint.2012.10.006
4. Pöllänen R, Toivonen H, Peräjärvi K, Karhunen T, Ilander T, et al. (2009) Radiation surveillance using an unmanned aerial vehicle. Applied radiation and isotopes 67: 340–344. doi: 10.1016/j.apradiso.2008.10.008 PMID: 19046635
5. Watts AC, Ambrosia VG, Hinkley EA (2012) Unmanned aircraft systems in remote sensing and scientific research: Classification and considerations of use. Remote Sensing 4: 1671–1692. doi: 10.3390/rs4061671
6. Balakirsky S, Carpin S, Kleiner A, Lewis M, Visser A, Wang J, et al. (2007) Towards heterogeneous robot teams for disaster mitigation: Results and performance metrics from robocup rescue. Journal of Field Robotics 24: 943–967. doi: 10.1002/rob.20212
7. Farinelli A, Rogers A, Petcu A, Jennings NR (2008) Decentralised coordination of low-power embedded devices using the max-sum algorithm. In: AAMAS. pp. 639–646.

8. Singh A, Krause A, Guestrin C, Kaiser WJ (2009) Efficient informative sensing using multiple robots. Journal of Artificial Intelligence Research 34: 707.
9. Schut M, Wooldridge M, Parsons S (2002) On partially observable mdps and bdi models. In: Foundations and Applications of Multi-Agent Systems, Springer. pp. 243–259.
10. Jennings NR (2000) On agent-based software engineering. Artificial intelligence 117: 277–296. doi: 10.1016/S0004-3702(99)00107-1
11. Kaelbling LP, Littman ML, Cassandra AR (1998) Planning and acting in partially observable stochastic domains. Artificial intelligence 101: 99–134. doi: 10.1016/S0004-3702(98)00023-X
12. Ny JL, Dahleh M, Feron E (2008) Multi-uav dynamic routing with partial observations using restless bandit allocation indices. In: American Control Conference, 2008. IEEE, pp. 4220–4225.
13. Agmon N, Kraus S, Kaminka GA, Sadov V (2009) Adversarial uncertainty in multi-robot patrol. In: IJCAI. pp. 1811–1817.
14. Basilico N, Gatti N (2011) Automated abstractions for patrolling security games. In: AAAI.
15. An B, Kempe D, Kiekintveld C, Shieh E, Singh SP, Tambe M, et al. (2012) Security games with limited surveillance. In: AAAI.
16. An B, Brown M, Vorobeychik Y, Tambe M (2013) Security games with surveillance cost and optimal timing of attack execution. In: AAMAS. pp. 223–230.
17. Vorobeychik Y, An B, Tambe M, Singh SP (2014) Computing solutions in infinite-horizon discounted adversarial patrolling games. In: ICAPS.
18. Qian Y, Haskell WB, Jiang AX, Tambe M (2014) Online planning for optimal protector strategies in resource conservation games. In: AAMAS. pp. 733–740.
19. Yost KA, Washburn AR (2000) The lp/pomdp marriage: Optimization with imperfect information. Technical report, DTIC Document.
20. Whittle P (1988) Restless bandits: Activity allocation in a changing world. Journal of Applied Probability 25: 287–298. doi: 10.2307/3214163
21. Castanón DA (2005) Stochastic control bounds on sensor network performance. In: IEEE Conference on Decision and Control. IEEE, pp. 4939–4944.
22. Zois DS, Levorato M, Mitra U (2013) Energy-efficient, heterogeneous sensor selection for physical activity detection in wireless body area networks. IEEE Transactions on Signal Processing 61: 1581–1594. doi: 10.1109/TSP.2012.2236320
23. Zhao Q, Tong L, Swami A, Chen Y (2007) Decentralized cognitive mac for opportunistic spectrum access in ad hoc networks: A pomdp framework. IEEE Journal on Selected Areas in Communications 25: 589–600. doi: 10.1109/JSAC.2007.070409
24. Ouyang Y, Teneketzis D (2014) On the optimality of myopic sensing in multi-state channels. IEEE Transactions on Information Theory 60: 681–696. doi: 10.1109/TIT.2013.2288636
25. Kurniawati H, Hsu D, Lee WS (2008) Sarsop: Efficient point-based pomdp planning by approximating optimally reachable belief spaces. In: Robotics: Science and Systems. volume 2008.
26. Silver D, Veness J (2010) Monte-carlo planning in large pomdps. In: NIPS. pp. 2164–2172.
27. Smallwood RD, Sondik EJ (1973) The optimal control of partially observable markov processes over a finite horizon. Operations Research 21: 1071–1088. doi: 10.1287/opre.21.5.1071
28. Sandholm WH (2010) Orders of limits for stationary distributions, stochastic dominance, and stochastic stability. Theoretical Economics 5: 1–26. doi: 10.3982/TE554
29. Keilson J, Kester A (1977) Monotone matrices and monotone markov processes. Stochastic Processes and their Applications 5: 231–241. doi: 10.1016/0304-4149(77)90033-3
30. Pynadath DV, Tambe M (2002) The communicative multiagent team decision problem: Analyzing teamwork theories and models. Journal of Artificial Intelligence Research 16: 2002.
31. Nemhauser GL, Wolsey LA, Fisher ML (1978) An analysis of approximations for maximizing submodular set functions—i. Mathematical Programming 14: 265–294. doi: 10.1007/BF01588971