Adversarial Patrolling with Reactive Point Processes

Benjamin N. Hefferan[*], **Oliver M. Cliff**[†], **Robert Fitch**[‡]
[*]School of Aerospace, Mechanical and Mechatronic Engineering, The University of Sydney, Sydney, NSW, Australia, `bhef8151@uni.sydney.edu.au`
[†]Australian Centre for Field Robotics, The University of Sydney, Sydney, NSW, Australia, `o.cliff@acfr.usyd.edu.au`
[‡]Centre for Autonomous Systems, University of Technology Sydney, Sydney, NSW, Australia, `rfitch@uts.edu.au`

Abstract

Adversarial patrolling is an algorithmic problem where a robot visits sites within a given area so as to detect the presence of an adversary. We formulate and solve a new variant of this problem where intrusion events occur at discrete locations and are assumed to be clustered in time. Unlike related formulations, we model the behaviour of the adversary using a stochastic point process known as the reactive point process, which naturally models temporally self-exciting events such as pest intrusion and weed growth in agriculture. We present an asymptotically optimal, anytime algorithm based on Monte Carlo tree search that plans the motion of a robot given a separate event detection system in order to regulate event propagation at the sites it visits. We illustrate the behaviour of our algorithm in simulation using several scenarios, and compare its performance to a lawnmower planning algorithm. Our results indicate that our formulation and solution are promising in enabling practical applications and further theoretical extensions.

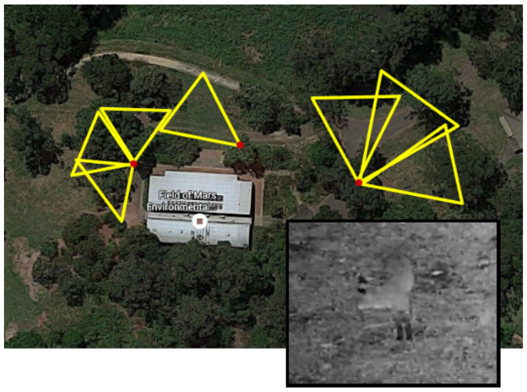


Figure 1: A motivating example of pest monitoring to minimise intrusion in a nature reserve. The surveillance network is a set of cameras (red dots with field of view (FOV) highlighted in yellow) to capture detection events. Inset: fox detected by one of the surveillance cameras. The task is to minimise fox intrusion by a robot visiting the sites in the FOV of the cameras.

1 Introduction

Patrolling is a task where an area must be continually monitored in order to detect undesirable events. Applications of patrolling are broad and include security surveillance, environmental monitoring for predators (such as shown in Fig. 1), and pest/weed control in agriculture. Algorithmically, the *patrolling problem* is to find a path, often a cycle, for a robot (or agent) that maximises an objective related to the probability of detection [Robin and Lacroix, 2016]. We are interested in a variant known as *adversarial patrolling*, where the behaviour of an adversary is affected by the actions of the robot. We consider the case where the likelihood of an event at a particular site increases contagiously over time when the robot is not present, and the action of a robot in making an observation reduces the likelihood of event occurrence at that site. Our objective is to plan a path that minimises the expected number of undesirable events within an area.

Patrolling problems are often studied in the non-adversarial case where events are independent [Smith and Rus, 2010; Yu *et al.*, 2014; 2015]. This model is useful for maintaining currency or minimising time-to-detect, but does not account for time-varying event likelihoods. Self-exciting point processes, such as the Hawkes process [Hawkes, 1971], are appealing in this regard because they naturally model phenomena that are locally contagious and form spatial and temporal clusters of events, e.g., criminal activity [Mohler *et al.*, 2012]. However, these models do not account for the adversarial scenario where an agent interacts with the environment.

Our approach is to employ *reactive point processes (RPPs)* in the context of adversarial patrolling [Ertekin *et al.*, 2015]. Unlike standard self-exciting point processes, the advantage of the RPP model is the ability to incorporate both the self-exciting nature of events and self-regulating property of making an observation. RPPs were originally developed within the domain of electrical grid reliability to predict catastrophic failures in the presence of inspection events [Ertekin *et al.*, 2015]; however, this general model is applicable to a broad spectrum of complex systems often studied in patrolling problems. In particular, we formulate our problem under the assumption that a fixed sensor network detects events, and a mobile robot patrols to regulate these events (decreases event vulnerability).

Given an RPP model, finding a path that optimises our objective is also challenging. Making an observation decreases the event likelihood at a site, but the benefit of a given observation is not fully known at the time it is made. Future observations at the same site can mitigate the effect of previous observations. A Markov decision process (MDP) formulation, for example, would include the event likelihood at each site as part of its state space, leading to large problem sizes for reasonably sized areas.

We present a planning algorithm based on Monte Carlo tree search (MCTS), a well-known algorithm used successfully in game AI [Silver *et al.*, 2016]. MCTS handles large state spaces by estimating action values through Monte Carlo sampling [Browne *et al.*, 2012]. We formulate the adversarial patrolling problem such that MCTS can be applied, and its convergence properties are retained. Simulation results demonstrate the useful behaviour of our algorithm in two scenarios, including comparison with myopic information-gain-style planning.

The contribution of this work is a novel problem formulation for adversarial patrolling with reactive self-excitation, and an asymptotically optimal and anytime planning algorithm for this problem. The significance of our work is to enable new applications of robots in a range of patrolling tasks where events cluster in time, including pest monitoring and adaptive weed control.

2 Related work

The problem of continuously monitoring (or patrolling) targets of interest is a cross-disciplinary research topic with numerous definitions depending on the particular task. The motivation for studying these problems can vary from crime prevention [Kartal *et al.*, 2015] to environmental monitoring [Smith *et al.*, 2011; Cliff *et al.*, 2015]. Given the broad scope of this problem, we will restrict our attention to problems cast as either continuous sweep coverage, combinatorial optimisation, or adversarial patrolling algorithms.

The aim of sweep coverage planning is to maximise environment coverage using a mobile sensor. The problem definition can be to cover fixed, discrete locations, e.g., where the areas of interest remain constant and the goal is to find a tour that maintains currency of information [Smith and Rus, 2010], or continuous coverage problems, e.g., in the existence of sharply temporal covariant environment [Garg and Ayanian, 2014]. Tasks such as mission monitoring can be cast as a spatiotemporal optimal stopping problem where the agent maintains a stochastic model of the target [Best *et al.*, 2015].

Another promising approach for patrolling is to use techniques from combinatorial optimisation to find a path for the robot through the environment that maximises some criteria. When These approaches typically draw on classical results from the travelling salesman problem [Alamdari *et al.*, 2014; Yu *et al.*, 2015], the travelling repairman problem [Afrati *et al.*, 1986; Tulabandhula *et al.*, 2011], and the vehicle routing problem [Stump and Michael, 2011] (with dynamic and stochastic demands [Bertsimas and Van Ryzin, 1991; Novoa and Storer, 2009; Pillac *et al.*, 2013]). For example, Bopardikar et al. [Bopardikar *et al.*, 2014] find an optimal solution to the vehicle routing problem with stochastic demands and time constants assuming demands arrive as per a known Poisson distribution. However, the main focus of this literature is typically to consider the arrivals of events and demands as spatiotemporally independent.

In this work, we consider an adversary that identifies vulnerable areas of the environment as ones that have been visited recently. However, it is common to assume the adversary takes one of two general policies: semi-random movements or reacting optimally to the patrolling agent (i.e., worst-case scenarios) [Robin and Lacroix, 2016]. Notably, Agmon et al. develop theoretical penetration bounds for static and stochastic intruders on many different environment types assuming the intruder attempts to penetrate the weakest point in the patrol [Agmon *et al.*, 2011]. Similarly, Kartal et al. modified MCTS to allow cycles in order to 'capture' an intruder that arrives and departs the environment randomly, performing random movements during the intrusion [Kartal *et al.*, 2015]. The problem we consider resides between these two extremes as the adversary's policy is neither explicitly modelled, nor considered completely random; the RPP models complex macroscopic behaviour by assuming a contagious nature of events.

3 Background and problem statement

In this section, we formally introduce the RPP, a doubly stochastic point process. We then define the problem of adversarial patrolling for events modelled by RPPs.

3.1 The reactive point process (RPP)

A temporal point process is a random process with a realisation consisting of a list of discrete events localised in time, $(t_m)_{m\in\mathbb{N}*} \in \mathbb{R}^+$. The temporal point process is therefore a counting process $N_t = \sum_{m\in\mathbb{Z}^+} \mathbb{1}_{t_m<t}$, that records the number of events before time t. The left-continuous conditional process λ_t captures the probability of occurrence of a new event adapted to a filtration $\mathcal{F}_t$ (i.e., the amount of information available *up to* time t) [Daley and Vere-Jones, 2007]. The intensity is therefore defined as

$$\lambda_t = \lim_{\Delta\to 0} \Delta^{-1}\mathbb{E}[N_{t+\Delta t} - N_t \mid \mathcal{F}_t]. \tag{1}$$

For brevity, we omit the dependence on filtration $\mathcal{F}_t$ in notation unless necessary.

The most common point process is the Poisson process, which is memoryless and thus characterises processes that are independent of prior information $\mathcal{F}_t$ (homogeneity), however can be dynamic in time (inhomogeneity). That is, durations $(t_m)_{m\in\mathbb{N}*}$ of a homogeneous Poisson process are i.i.d. as an exponential distribution with rate parameter λ_t. The Poisson process is thus commonly used to model non-adversarial patrolling problems, where the events are considered independent (see Sec. 2). In this work we consider an adversarial process, where the adversary is more likely to attack areas that were previously vulnerable. As a result, we consider the Hawkes process to model adversaries.

The Hawkes process is a self-exciting point process model that has been used to model a number of relevant phenomena, e.g., earthquakes [Ogata, 1998], criminal activity [Mohler *et al.*, 2012], and financial data [Bacry *et al.*, 2015]. In its general form, the process captures the mutual excitation phenomena between events. For a K-variate Hawkes process, the couples $\{t_m, k_m\}_{m=1}^M$, where t_m denotes the time of event number m and $k_m \in [1,\ldots,K]$ indicates its component, the intensity is computed as [Bacry *et al.*, 2015]

$$\lambda_t^i = \mu^i + \sum_{t_m<t} \phi^{i,k_m}(t - t_m). \tag{2}$$

Thus each component of the process i can be coupled to any other component; however, in this work we assume each component is independent and thus remove the index k_m. The exponential kernel for the Hawkes process was originally defined as $\phi(t) = \sum_{j=1}^P \alpha_j e^{-\beta_j t}\mathbb{1}_{\mathbb{R}_+}$ for some order P [Hawkes, 1971].

However, the self-exciting point process defined in (2) is insufficient for patrolling. We model the behaviour of the adversary as less likely to attack a given area once it has been visited (e.g., on a patrol beat). Thus, we use the RPP model, where the intensity is given by [Ertekin *et al.*, 2015]

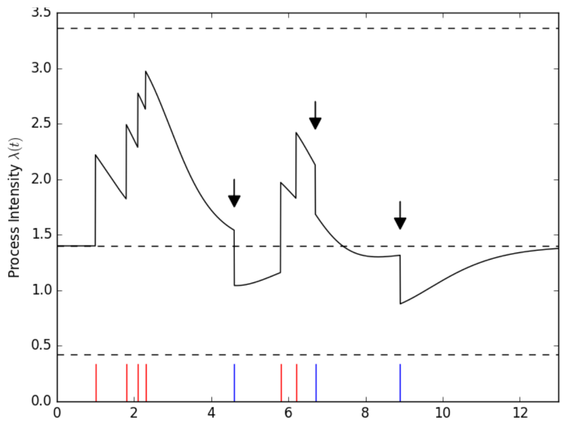


Figure 2: The RPP model; events (red bars) are temporally contagious, where one event will increase the vulnerability of another happening due to the *self-exciting* property; this likelihood then tempers over time to a base value. Once an observation (blue bar) is made, there is an instantaneous decrease in vulnerability due to the *self-regulating* property of the process.

$$\lambda_t^i = \mu^i \left[1 + g_1^i\left(\sum_{t_m<t}\phi^i(t-t_m)\right) + g_2^i\left(\sum_{\bar{t_n}<t}\psi^i(t-\bar{t_n})\right) + C_1\mathbb{1}_{[N_E\geq 1]}\right] \tag{3}$$

The RPP was initially used to predict power failures in underground electrical systems, whereby a *failure* excites the event likelihood and an *inspection* regulates it. We use the functions and kernels defined in the original setting of manhole inspection,

$$\begin{aligned}
g_1^i(\omega) &= a_1^i\left(1 - \frac{\log(1-e^{-b_1^i\omega})}{\log(2)}\right),\\
\phi^i(t) &= \frac{1}{1+e^{\beta^i t}},\\
g_2^i(\omega) &= a_2^i\left(1 - \frac{\log(1-e^{b_2^i\omega})}{\log(2)}\right),\\
\psi^i(t) &= \frac{-1}{1+e^{\gamma^i t}}
\end{aligned}$$

The RPP formulation includes saturation functions g_1 and g_2 on top of the standard self-exciting and self regulating properties which prevent the intensity from reaching unrealistic values under the self-exciting or self-regulating actions.

3.2 Problem statement

We study the problem of adversarial patrolling under the framework of MDPs. An MDP is a 5-tuple

$(\mathcal{S}, \mathcal{A}, \mathcal{P}^a_{ss'}, \mathcal{R}^a_{ss'}, \gamma)$ where $\mathcal{S}$ is the set of states; $\mathcal{A}$ is the set of actions; $\mathcal{P}^a_{ss'} = \Pr(s_{n+1} = s' \mid s_n = s, a_n = a)$ is the *transition probability* for moving to state s' from state s by action a; $\mathcal{R}^a_{ss'} = \mathbb{E}[r_{n+1} \mid s_n = s, a_n = a]$ is the *reward function* for a state transition; and γ is a discount factor. The goal of the MDP is to choose a policy $a_n = \pi(s_n)$ (i.e., a map from states to actions) that maximises the discounted return R_T for some horizon T.

We consider a discrete, graph-based representation of places (targets) of interest that have been *a priori* abstracted from the environment. A graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ is characterised by a node set $\mathcal{V} = \{v^1, v^2, \ldots, v^K\}$ and edge set $\mathcal{E} \subseteq \mathcal{V} \times \mathcal{V}$. Each edge in the edge set is denoted e^{ij} to denote the edge between node v^i and v^j. The environment (graph) can be considered to be a multivariate RPP, where each node v^i will have an associated intensity λ^i_t. We assume that node intensities are known at each decision point. This assumption is motivated by practical situations, e.g., where a surveillance system (e.g., cameras) is available and the role of the robot is interventional.

Let the state $s_n = (v_n, t, \{\mathbb{E}[\lambda^i_t]\})$ at decision index n comprise the node the agent is currently at $v_n \in \mathcal{V}$, the decision time t, and the expected intensities $\{\mathbb{E}[\lambda^1_t], \mathbb{E}[\lambda^2_t], \ldots, \mathbb{E}[\lambda^K_t]\}$ of all nodes at the decision time t. We allow the agent to move to any node excluding the state node $a_n \in \mathcal{V} \setminus v_n$. Denote $\mathbb{E}[N^i_t]$ as the number of events at node v^i assuming no interaction with the environment, and $\mathbb{E}_\pi[N^i_t]$ the expected number of events when following policy π (i.e., inspecting a node according to π). We then wish to minimise the total number of events that will occur, i.e., maximise the return at time t

$$R_T = 1 - \frac{\sum_{i=1}^K \mathbb{E}_\pi[N^i_{t+T} - N^i_t \mid \mathcal{F}_t]}{\sum_{i=1}^K \mathbb{E}[N^i_{t+T} - N^i_t \mid \mathcal{F}_t]}, \tag{4}$$

4 Planning algorithm

We propose to use MCTS as an anytime planning algorithm for finding the optimal action from the root node. Using the UCT child selection policy, the algorithm proposed is both asymptotically optimal and anytime.

4.1 Monte Carlo tree search

The goal of an MDP is to select an optimal (one-step) action. To achieve this, we employ MCTS, which performs random rollouts by Monte Carlo sampling to determine an optimal action from a root node, given a black box simulator of the environment [Browne *et al.*, 2012]. The four main stages of the MCTS algorithm are node selection, expansion, simulation and back-propogation (or backup), as illustrated in Fig. 3.

Node selection The node selection step chooses an unvisited leaf node of the tree. This node is determined by a UCT search from the root node in which the selection of each child node form the current node is considered to be an independent mulit-armed bandit problem [Kocsis and Szepesvári, 2006]. The expected benefit of expanding a child node is determined using the UCT algorithm where the expected reward for each child node is given by

$$UCT = \bar{R}_j + 2C_p\sqrt{\frac{2\ln n}{n_j}}$$

where $\bar{R}_j$ is the average return of patrolling policies (4) passing through this node, n_j is the number of times the child has been previously visited and n is number of time the parent(current) node has been visited. The tree policy terminates when a node with unexplored children is reached.

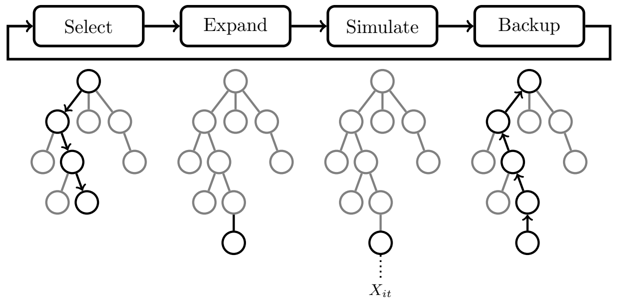


Figure 3: The four main stages in the MCTS algorithm: node selection, expansion, simulation, and backup.

Expansion Once a node is selected as described above, a child node is added corresponding to randomly selected available action form this state that has not been previously added to the tree. From this newly added node a default policy is applied to guide the actions taken by the robot while searching the graph. In this case random actions are selected from a list of possible actions for each state [Browne *et al.*, 2012].

Simulation The aim of the simulation stage of MCTS is to determine the utility of a possible action sequence that the agent could take. From Eq. (1), the expected number of events in (4) is given by integrating the intensity, i.e.,

$$\mathbb{E}[N_T - N_t \mid \mathcal{F}_t] = \int_{u=t}^{T} \mathbb{E}[\lambda_u \mid \mathcal{F}_t].$$

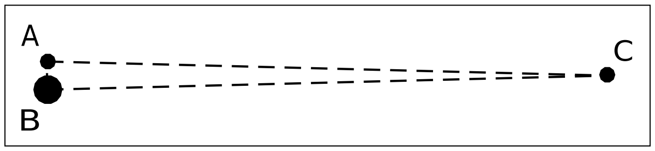
Thus, we can calculate the reward in (4) by either integrating (3) or sampling the number of events with and without interacting with the environment. As analytic solutions to the integrated intensity are difficult to find, we will use the observed number of events as an approximation to this integral. This approximation only holds while the number of events is large and thus in cases where the number of events is small, a numeric integration is performed on the conditional intensity function.

Once a node has been selected the expected reward R_T is calculated for this state. This is often achieved by applying default (rollout) policy until a terminal state is reached in which the outcome can be evaluated and a reward calculated. As patrolling is a task with no inherent end, a different approach is needed. In order to evaluate the expected reward, the default policy is applied up to a fixed horizon [Samothrakis *et al.*, 2011] time after which the reward for the rollout is calculated.

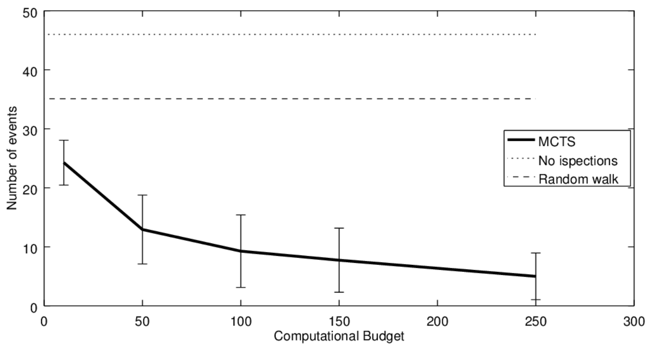
The reward is calculated by simulating the point processes (events) for all nodes in the environment using thinning algorithms [Ogata, 1981]. These procedures can be used to simulate any point process that is fully defined by a conditional intensity function. To perform the rollout, the simulation is generated from the decision time (i.e., the root state) of the search tree s_n. The simulation is conducted for the case with and without any robot intervention in the system. Inspections are considered to occur when the robot arrives at a location. The inspection times applied in the simulation include environment nodes visited during the tree walk nodes visited during the default policy. The reward R_T is calculated from the resulting process intensity from the decision time up until the simulation horizon time.

Back propagation Once a reward has been calculated, it is back propagated up the search tree. The expected reward is added to the average empirical reward $\bar{R_j}$ of the expanded node and all of its ancestors up to the root node and the visit count of all nodes n_j on this path incremented. This then informs subsequent UCT searches guiding the tree exploration and expansion towards regions of the action space expected to be of highest utility to the agent.

Action selection Once the allotted budget for the tree search has been consumed an action is selected by consulting the children of the root node of the tree. Action corresponding the most valuable child is then selected as the next action of the robot using the 'max child' criteria [Browne *et al.*, 2012]. Once an action has been selected the agent state is updated and the same simulation method is used to generate the environment processes up to the time the agent arrives at its next location at which time a the next decision making process begins.



(a)



(b)

Figure 4: Experimental validation with the three node environment. Figure (4a) illustrates the environment structure: two adjacent nodes, one with high intensity and one with low, and a third, further away node with low intensity. Figure (4b) illustrates the number of events as a function of the computational budget. The number of events are shown for MCTS (error bar), random movements (dashed line) and without intervention (dotted line).

4.2 Analysis

The planning algorithm is a specific implementation of the UCT algorithm [Browne *et al.*, 2012], and thus guarantees the bias at the root node converges at a rate $\mathcal{O}(\log(n)/n)$, where n is the number of rollouts. The only assumption for convergence of UCT is that the empirical averages converge. We can guarantee convergence of the mean values given the expected intensity is bounded, which is the case for RPP's due to saturation functions g_1 and g_2. In addition, the algorithm is anytime in that we can terminate the MCTS prior to convergence of the root node, where the agent selects the action that has the highest empirical average.

5 Experimental validation

In the previous section, we showed that the algorithm is asymptotically optimal and anytime. Thus, the main goal of this section is to both quantitatively and qualitatively validate the behaviour of the planning algorithm introduced above in simple but expressive toy environments.

Firstly, we validate the planning algorithm on a simple, three node graph ($K = 3$) shown in Fig. 4a. The

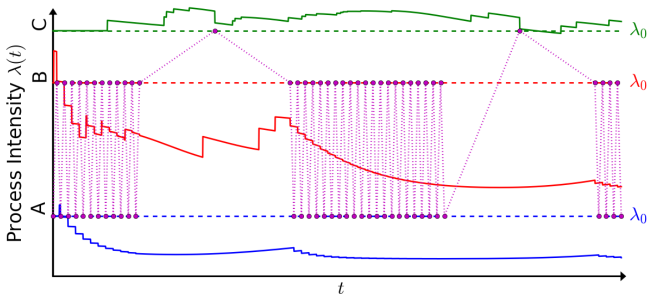
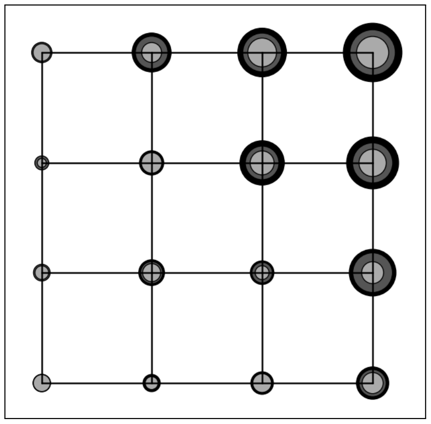


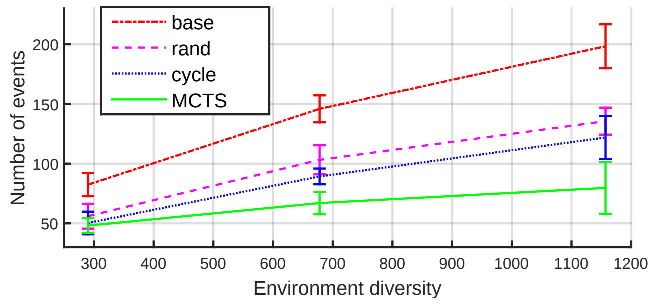
Figure 5: Path chosen by the MCTS planner in a trial on the three node graph. Horizontal dashed lines indicate the baseline rates of the three processes (nodes A,B and C) and solid lines show the process intensities. The magenta (dotted) line indicates the path chosen by the planner with circles indicating inspection times for each of the processes.

graph is designed such that two nodes (leftmost nodes in Fig. 4a) are placed adjacent to one another (0.1 units), where one node has high and the other low expected intensity. The rightmost node is situated further away (2 units) from this cluster, and given low expected intensity. In general, the agent performed an oscillating policy, where the two leftmost nodes were visited frequency, and the right node was visited at a significantly lower frequency as in Fig. 5. In addition, we show in Fig. 4b that increasing the computation budget significantly improves the performance of the planner. With a shorter budget, the planner effectively acts greedily. We also compare the performance of the planner with a random planner (effectively setting the horizon to 0) and base expected intensity (i.e., where there is no intervention from the agent at all). Increasing the planning horizon and budget demonstrates the asymptotic behaviour of the planner, and a comparison with the base cases illustrates the effectiveness of the optimal policy. Our next experiment, shown in Fig. (6) exemplifies the performance of the MCTS planning for non-uniform environments. We are interested in the utility of the algorithm as the number of events at each node becomes more varied, which we quantify with a measure termed *event diversity*. Event diversity is given by the standard deviation of the number of events at each node without intervention. To study this, we use a 16-node ($K = 16$) grid-world environment, illustrated in Fig. 6a. A naive coverage planner for these types of environments is to sweep the environment in a lawnmower style pattern. Thus, we contrast the MCTS algorithm with both a random and a lawnmower-style planner for validation. As illustrated in Fig. 6a, the event diversity is varied over a number of trials, and the number of events are captured for each trial and planner. The graph in Fig. (6b) shows that the number of events using the MCTS planner decreases with the baseline, whereas the events for random and lawnmower planners remain relatively constant. The improved performance of the MCTS algorithm is given by the ability to react to the processes instantaneous event probability (as an active planner).

When normalised to the base intensity of the environment (i.e. without any inspections), both random and lawnmower paths display approximately constant utility while the utility of the MCTS planner is seen to increase as the environment diversity increases for this environment.



(a)



(b)

Figure 6: Grid scenario used for experimental evaluation. The processes governing the grid were changed to vary the *event diversity*, i.e., how uniform the expected number of events were across the entire graph. In Fig. 6a the relative size of each node represents the average intensity across a trial, and the shade denotes three separate trials to illustrate which nodes were varied. Figure 6a illustrates the number of events plotted against event diversity without intervention (dash-dot line), with random movements (dashed line), with the lawnmower pattern (dotted line), and with MCTS (solid line).

6 Discussion and future work

We presented a novel formulation of adversarial patrolling, where intrusion events occur at discrete locations and are assumed to be clustered in time. An asymptotically optimal and anytime planning algorithm was then developed to solve this problem. We then validated the planning algorithm's performance both qualitatively and quantitatively by considering a number of environments.

There is broad scope for future work to consider for this type of problem formulation. We are currently investigating applying the model to a real system for pest monitoring problem (see Fig. 1) where the RPP parameters are learned from interventional data. Furthermore, a natural extension to this problem are multivariate RPPs [Bacry *et al.*, 2015], which are capable of capturing both temporally and spatially clustered events however would require additional structure learning procedures for these types of dynamical systems [Cliff *et al.*, 2016]. Finally, two important lines of inquiry are multi-robot decentralised planning [Best *et al.*, 2016] and partial observability (e.g., as a partially observable MDP [Silver and Veness, 2010]).

7 Acknowledgements

This work was supported in part by the Australian Centre for Field Robotics and the New South Wales Government. The authors would also like to thank Timothy Patten for discussions and for providing figure 3 and Graeme Best for discussions related to this work.

References

[Afrati *et al.*, 1986] Foto Afrati, Stavros Cosmadakis, Christos H Papadimitriou, George Papageorgiou, and Nadia Papakostantinou. The complexity of the travelling repairman problem. *Inform. Theor. Appl.*, 20(1):79–87, 1986.

[Agmon *et al.*, 2011] Noa Agmon, Gal A Kaminka, and Sarit Kraus. Multi-robot adversarial patrolling: facing a full-knowledge opponent. *J. Art. Intell. Res.*, 42:887–916, 2011.

[Alamdari *et al.*, 2014] Soroush Alamdari, Elaheh Fata, and Stephen L. Smith. Persistent monitoring in discrete environments: Minimizing the maximum weighted latency between observations. *Int. J. Robot. Res.*, 33(1):138–154, 2014.

[Bacry *et al.*, 2015] Emmanuel Bacry, Iacopo Mastromatteo, and Jean-François Muzy. Hawkes processes in finance. *Mark. Microstructure Liq.*, 1(01):1550005, 2015.

[Bertsimas and Van Ryzin, 1991] Dimitris J Bertsimas and Garrett Van Ryzin. A stochastic and dynamic vehicle routing problem in the Euclidean plane. *Oper. Res.*, 39(4):601–615, 1991.

[Best *et al.*, 2015] Graeme Best, Wolfram Martens, and Robert Fitch. A spatiotemporal optimal stopping problem for mission monitoring with stationary viewpoints. In *Proc. of RSS*, 2015.

[Best *et al.*, 2016] Graeme Best, Oliver M. Cliff, Timothy Patten, Ramgopal Mettu, and Robert Fitch. Decentralised Monte Carlo tree search for active perception. In *Proc. of WAFR*, 2016.

[Bopardikar *et al.*, 2014] Shaunak D Bopardikar, Stephen L Smith, and Francesco Bullo. On dynamic vehicle routing with time constraints. *IEEE Trans. Robot.*, 30(6):1524–1532, 2014.

[Browne *et al.*, 2012] Cameron B Browne, Edward Powley, Daniel Whitehouse, Simon M Lucas, Peter I Cowling, Philipp Rohlfshagen, Stephen Tavener, Diego Perez, Spyridon Samothrakis, and Simon Colton. A survey of Monte Carlo tree search methods. *IEEE Trans. Comput. Intell. AI in Games*, 4(1):1–43, 2012.

[Cliff *et al.*, 2015] Oliver M. Cliff, Robert Fitch, Salah Sukkarieh, Debra L. Saunders, and Robert Heinsohn. Online localization of radio-tagged wildlife with an autonomous aerial robot system. In *Proc. of RSS*, 2015.

[Cliff *et al.*, 2016] Oliver M. Cliff, Mikhail Prokopenko, and Robert Fitch. An information criterion for inferring coupling in distributed dynamical systems. *Front. Robot. AI*, 3(71), 2016.

[Daley and Vere-Jones, 2007] Daryl J Daley and David Vere-Jones. *An introduction to the theory of point processes: Volume II: General theory and structure.* Springer Science & Business Media, 2007.

[Ertekin *et al.*, 2015] Şeyda Ertekin, Cynthia Rudin, Tyler H McCormick, et al. Reactive point processes: A new approach to predicting power failures in underground electrical systems. *Ann. Appl. Stat.*, 9(1):122–144, 2015.

[Garg and Ayanian, 2014] Sahil Garg and Nora Ayanian. Persistent monitoring of stochastic spatio-temporal phenomena with a small team of robots. In *Proc. of RSS*, 2014.

[Hawkes, 1971] Alan G Hawkes. Spectra of some self-exciting and mutually exciting point processes. *Biometrika*, 58(1):83–90, 1971.

[Kartal *et al.*, 2015] Bilal Kartal, Julio Godoy, Ioannis Karamouzas, and Stephen J. Guy. Stochastic tree search with useful cycles for patrolling problems. In *Proc. of IEEE ICRA*, pages 1289–1294, 2015.

[Kocsis and Szepesvári, 2006] Levente Kocsis and Csaba Szepesvári. Bandit based monte-carlo planning. In *Proc. of ECML*, pages 282–293. Springer, 2006.

[Mohler *et al.*, 2012] George O Mohler, Martin B Short, P Jeffrey Brantingham, Frederic Paik Schoenberg, and George E Tita. Self-exciting point process modeling of crime. *J. American Stat. Assoc.*, 2012.

[Novoa and Storer, 2009] Clara Novoa and Robert H. Storer. An approximate dynamic programming approach for the vehicle routing problem with stochastic demands. *Europ. J. Oper. Res.*, 196(2):509–515, 2009.

[Ogata, 1981] Yosihiko Ogata. On lewis' simulation method for point processes. *IEEE Trans. Inform. Theor.*, 27(1):23–30, 1981.

[Ogata, 1998] Yosihiko Ogata. Space-time point-process models for earthquake occurrences. *Ann. Inst. Stat. Math.*, 50(2):379–402, 1998.

[Pillac *et al.*, 2013] Victor Pillac, Michel Gendreau, Christelle Guret, and Andrs L. Medaglia. A review of dynamic vehicle routing problems. *Europ. J. Oper. Res.*, 225(1):1–11, 2013.

[Robin and Lacroix, 2016] Cyril Robin and Simon Lacroix. Multi-robot target detection and tracking: Taxonomy and survey. *Auton. Robots*, 40(4):729–760, 2016.

[Samothrakis *et al.*, 2011] S. Samothrakis, D. Robles, and S. Lucas. Fast approximate max-n Monte Carlo tree search for Ms Pac-Man. *IEEE Trans. Comput. Intell. AI in Games*, 3(2):142–154, June 2011.

[Silver and Veness, 2010] David Silver and Joel Veness. Monte-Carlo planning in large POMDPs. In *Advances in neural information processing systems*, pages 2164–2172, 2010.

[Silver *et al.*, 2016] David Silver, Aja Huang, Chris J Maddison, Arthur Guez, Laurent Sifre, George Van Den Driessche, Julian Schrittwieser, Ioannis Antonoglou, Veda Panneershelvam, Marc Lanctot, et al. Mastering the game of go with deep neural networks and tree search. *Nature*, 529(7587):484–489, 2016.

[Smith and Rus, 2010] Stephen L. Smith and Daniela Rus. Multi-robot monitoring in dynamic environments with guaranteed currency of observations. In *Proc. of IEEE CDC*, pages 514–521, 2010.

[Smith *et al.*, 2011] Ryan N Smith, Mac Schwager, Stephen L Smith, Burton H Jones, Daniela Rus, and Gaurav S Sukhatme. Persistent ocean monitoring with underwater gliders: Adapting sampling resolution. *J. Field Robot.*, 28(5):714–741, 2011.

[Stump and Michael, 2011] Ethan Stump and Nathan Michael. Multi-robot persistent surveillance planning as a vehicle routing problem. In *Proc. of IEEE CASE*, pages 569–575, 2011.

[Tulabandhula *et al.*, 2011] Theja Tulabandhula, Cynthia Rudin, and Patrick Jaillet. The machine learning and traveling repairman problem. In *Alg. Decis. Theor.*, pages 262–276. Springer, 2011.

[Yu *et al.*, 2014] J. Yu, M. Schwager, and D. Rus. Correlated orienteering problem and its application to informative path planning for persistent monitoring tasks. In *Proc. of IEEE/RSJ IROS*, pages 342–349, 2014.

[Yu *et al.*, 2015] Jingjin Yu, Sertac Karaman, and Daniela Rus. Persistent monitoring of events with stochastic arrivals at multiple stations. *IEEE Trans. Robot.*, 31(3):521–535, 2015.