Online Agent-Based Aerial Patrol Planning for Wildlife Surveillance

Master of Science Thesis

Karel Dhoore

Delft University of Technology

TU Delft

Online Agent-Based Aerial Patrol Planning for Wildlife Surveillance

Master of Science Thesis

by

Karel Dhoore

to obtain the degree of Master of Science
at the Delft University of Technology,
to be defended on October 12th, 2020

Student number: 4220587

Date:	October 12, 2020		
Project duration:	March 2019 – October 2020		
Thesis committee:	Asst. prof. dr. B.F. Santos	Delft University of Technology	Chair
	Asst. prof. dr. O.A. Sharpanskykh	Delft University of Technology	Supervisor
	Asst. prof. dr. D.M. Pool	Delft University of Technology	External member

An electronic version of this thesis is available at repository.tudelft.nl.

Cover image courtesy of the "Thermal Wildlife Drones" YouTube channel

TU Delft

Preface

One and a half years ago, I was approached by a friend of mine. He co-founded a startup that aimed to protect wildlife and farm animals in Africa by means of drone surveillance. The team envisioned a highly autonomous system, where a UAV would fly over many square kilometres of land looking for animals and poachers and report on its observations. However, finding poachers is a difficult task. In order to spend less time looking for poachers and more time finding them, they wanted to know the optimal route for the drone. As a result, I chose to research this problem in the context of my master's thesis. Working on this assignment was an interesting and strenuous experience. I learned a lot about writing code, I've experienced the main aspects of doing academic research, and I got to know the world of wildlife protection and "AI for Good". I worked with the motivation to make the outcome of this research useful for further research in the field of wildlife surveillance, with the hope that the knowledge can be applied in practice in the not-to-distant future. I am thankful to have had the opportunity to support this cause, and I am proud to have achieved the results presented in this report.

This assignment is the conclusion of three years of learning, experiencing, and researching in the field of air transport and operations. It was a very exciting period that I will never forget, and where I have learned more than I could think of. It is also a period where I realised that there is infinitely more knowledge to be gained. It motivates me to never stop learning in the future and to continue to be curious and critical of what life brings.

Finally, I have a lot of people to thank for bringing me where I am now. First of all, I am grateful for all the time, support, and understanding from friends and family. I am especially thankful to my parents, who made it possible for me to do what I like at my own pace.
I would like to thank Alexei, my teacher and supervisor, for the ever-motivating atmosphere, his knowledge and understanding of what it took to work on this graduation assignment. His dedication to quality in research and education is truly admirable.
Last but not least, I am thankful to Jamie and the rest of the team at Eyeplane for coming to me with this interesting research problem and supporting me throughout this project.

Karel Dhoore
Delft, October 2020

Contents

I

Paper

Online Agent-Based Aerial Patrol Planning for Wildlife Surveillance

K. Dhoore

Supervisor: dr. O.A. Sharpanskykh
Delft University of Technology, Faculty of Aerospace Engineering, Department of Control & Operations
Kluyverweg 1, 2629HS Delft, The Netherlands

***Abstract*—Wildlife conservation efforts are constrained by a limited amount of resources available for surveillance activities. UAVs are used increasingly to assist rangers in patrol tasks. Effectively patrolling wildlife parks requires detailed knowledge of the environment and its threats, which is not always available. Previous work in Green Security Games (GSGs) that aims to develop defensive strategies to deter adversaries relies on historical poaching data to train machine learning models. Recent advancements in the field have led to the development of an online learning framework that does not require prior data. However, the defensive strategies resulting from this approach are focused on foot patrols by rangers, which do not have the same mobility as UAVs, or do not take into account spatio-temporal constraints associated with patrolling in a real-world situation at all. To address the desire of using UAVs for wildlife surveillance, this paper proposes MEOMAPP, a model that extends on the online learning approach by incorporating a patrol planning algorithm more suitable for aerial patrol. It also includes an evaluative algorithm that considers a human expert next to the online learning expert and balances the application of their strategies based on the observed performance of each expert. By simulating MEOMAPP in a realistic environment, the research demonstrates that the model is suitable to determine aerial surveillance strategies for wildlife conservation.**

***Index Terms*—Green Security Games; Game Theory; Online Learning; Adversarial Bandits; Agent-Based Modelling; Aerial Surveillance; Wildlife Conservation**

I. Introduction

Poaching is still a major problem in large parts of the world. It threatens efforts in wildlife conservation, which negatively impacts biodiversity and possibly results in damaged ecosystems [1]. There is also a large economic cost in the form of reduced income from wildlife tourism and trophy hunters. Currently, the cost of measures to keep animals protected and safe from poachers is not economically viable [2]. Simultaneously, protecting wildlife is not always without risks either. In Africa alone, 349 rangers have died on duty since 2012, although it is thought these figures are substantially higher due to lack of reporting [3].

These high costs are a driver for cost-effective and innovative measures in the wildlife protection domain. Notably, the use of artificial intelligence (AI) has shown potential for detecting animals and poachers with object and image recognition [4, 5], and it can also assist in determining optimal patrol routes based on historical poaching data. Moreover, the deployment of drones is increasingly popular for the conservation of protected areas in general. Their capability to perform surveillance in a relatively low-cost risk-free manner on a high spatio-temporal resolution with a diverse range of sensors makes them a desirable addition to the tools already in place [6, 7]. In the future, it will be possible to develop truly autonomous surveillance systems by coupling current autopilot capabilities of UAVs with AI-driven image recognition tools and surveillance strategies.

A frequently used framework that focuses on developing solutions for the surveillance planning problem is the formulation of a Green Security Game (GSG) [8], a type of Stackelberg Security Game (SSG). In this format, the interactions between patrollers (defenders) and poachers (attackers)are modelled as a repeated single-shot game. The attacker carries out one or multiple attacks, while simultaneously the defender defends according to a specific strategy. The payoff for the defender depends on where the attacker attacked at that round. Multiple repetitions of this single-shot game, which we consider a single round in an infinite game, allow the defender and the attacker to learn and subsequently adapt their strategies. The resulting defender strategy can be used to define where surveillance should take place on the terrain.

A key characteristic of most AI methods is their dependence on large amounts of data to train their internal models. Next to common problems associated with data sets, like imperfections, incompleteness, and data bias [9, 10], an often overlooked fact is that data is not always available in the first place. Especially when developing models to predict optimal patrol routes for wildlife surveillance, there is no guarantee that specific information is available or will be available in the future. This information is about where which animals are at a certain time, where and how many poachers are active on the terrain, and how many attacks have taken place historically. The absence of this knowledge makes it difficult to determine patrol routes for computers and humans alike.

The majority of previously proposed models require historic attack data and/or a complete attacker model with various defining features, such as a specific behaviour model and full knowledge of the attacker's payoff structure [11–18]. However, it is even recognised that usually the attacker's payoffs are unknown to the defender [19, 20]. Moreover, the defender might not even know its own payoffs due to too much

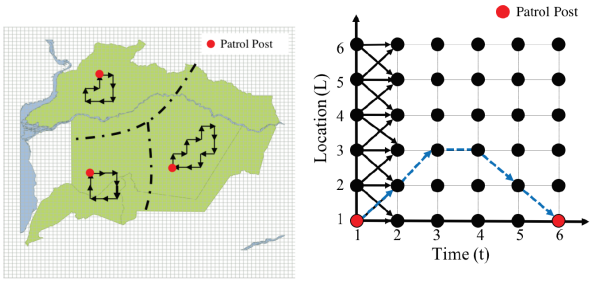

Figure 1. Patrol planning with a time-unrolled graph as by the MINION model [23].

uncertainty in nature. For example, if the payoffs are based on the amount and type of animals at a certain location, they can be random and/or variable, making it difficult to estimate the value of the payoffs.

Recent research by Xu et al. [21] tries to tackle the problems posed by uncertainty by proposing an online learning algorithm to develop a surveillance strategy without prior knowledge. The model, called the Follow the Perturbed Leader with Uniform Exploration (FPL-UE), is an adaptation of the method proposed by Neu and Bartók [22]. It makes no assumptions about adversary behaviour nor defender payoffs while still guaranteeing an efficient theoretical performance. Specifically, it assumes an arbitrary attacker and puts no assumptions on their behaviour or payoff structure. It then chooses between *exploring* (learning which strategy works best) and *exploiting* (maximising utility with gathered knowledge).

the FPL-UE algorithm was further developed by Gholami et al. [23] to take into account spatio-temporal planning constraints proper to patrol rangers, like limited walking time or distance and limited selection of accessible targets. This made the algorithm more applicable for surveillance by foot patrol in the real world. It calculated a feasible patrol route by selecting a starting point at a patrol post and solving an equally distributed time-unrolled graph of adjoining accessible targets on a grid constraint by a specific time horizon (see Figure 1). However, this method is not practical for determining a surveillance flight performed by a drone, since a drone is not bound by flying between adjoining targets. Additionally, Gholami et al. [23] introduces an expert-selection method to evaluate the online learning expert (based on FPL-UE) and a decision tree-based machine learning expert during the game and selecting the best performing expert. However, this approach has its limitations. First of all, since no data is available to train the machine learning expert, a static probability map substitutes as a simulation of the decision tree algorithm's results. Secondly, the expert-selection method proposed by Gholami et al. [23] already starts evaluating both experts at the start of the game. This means that the online learning expert is evaluated before it had the chance to learn.

We propose a novel approach to the patrol planning problem where we take advantage of patrolling with a drone. Continuing on the work by Xu et al. [21], we adopt the same combinatorial adversarial online learning problem formulation to determine a preliminary set of targets for a defender strategy. We formulate the flight path planning as an Orienteering Problem (OP) constrained by the practical limitations of the drone. The solution to the OP results in the final strategy. Inspired by Gholami et al. [23], the model, aptly named the Multi-Expert Online Model for Aerial Patrol Planning (MEOMAPP), also incorporates an expert-selection algorithm that allows evaluating its performance with a second expert. Contrary to Gholami et al. [23], the experts are not evaluated right away, but only after the online expert has had a chance to learn.

In this paper, we present an agent-based model developed for evaluating MEOMAPP's performance. The model represents a simplified wildlife surveillance system, composed of a domain to be surveilled (the *environment*), the drone that performs the surveillance flights (the *defender* agent), and one or multiple poachers (the *attacker* agents). The defender behaves according to the strategies determined by MEOMAPP. We selected two common attacker models to take it up against MEOMAPP: (i) a simple stochastic model with predefined attack probabilities per cell, and (ii) a Quantal Response model [13], which is a state of the art adaptive attacker model with bounded rationality. For a second expert, we assumed a realistic practical scenario where a person familiar with the domain to be surveilled gives every target an attackability score that is used for a probability-based mixed defender strategy.

The question we want to address with this research is whether an FPL-UE algorithm in a multi-expert learning model with a planning method suitable for drones is a viable application for determining wildlife surveillance strategies. To answer this question, we test MEOMAPP using an agent-based model on a real-life wildlife surveillance case. For this case, we develop defender strategies for the Aloegrove Safari Park in Namibia against simulated attackers and with a human expert as a competing strategy method. The test results are inspected for convergence of performance over time and performance variations of MEOMAPP following changes of operational and environmental parameters of the model. The suitability of the model is valid if the convergence behaviour is similar to previous research.

The paper is organised as follows: section II covers additional related work, section III provides a detailed account of the problem formulation, section IV lays out the agent-based model including the specifications of MEOMAPP, section V describes the numerical evaluation by means of the real-world case study. The results are discussed in section VI and we draw a conclusion in section VII. Finally, we present recommendations for future research in section VIII.

II. Related Work

In this section, we address further how our research compares to prior literature regarding GSGs and online learning methods for wildlife surveillance, matheuristics for path planning, and the agent-based modelling and simulation paradigm.

A. *Adversary Modelling in Green Security Games*

It is understood that assuming a perfectly rational, value-maximising adversary is not ideal for addressing human adversaries [24]. Subsequently, two competing approaches have emerged to address human bounded rationality in SSGs and subsequently GSGs. One approach departed from the idea that attackers behave according to specific parametric models of human decision-making of which the parameters could be learned by fitting them to historical data, and subsequently deriving a defender strategy based on the probability where an attacker would attack next. These models include the BRQR algorithm [11] and MATCH [12], based on a Quantal Response (QR) model for adversary behaviour [13]. Based on a new attacker model with an added Subjective Utility function to the QR model (SUQR) [14], algorithms SU-BRQR [14], PAWS [15, 16] and SHARP [17] were developed, followed up by CAPTURE [18]. These models use parameter estimation methods like Maximum Likelihood Estimation (MLE) or Estimation Maximisation (EM) to determine the adversary model's parameters. The underlying data for the estimations comes from real-world experiments or simulations with actual human players.

The other approach is to intentionally avoid adversary modelling and instead focusing directly on reward maximised route optimization based on the individual targets. These methods usually make use of data-driven machine learning techniques. Models like INTERCEPT [25] and others by Gurumurthy et al. [26] and Gholami et al. [27, 28] are based on decision trees that use the target's environmental characteristics and historical attack data to predict the attackability of individual targets. APE, the algorithm by Park et al. [29] uses a variety of classification algorithms in combination with live GPS data of animals and patrol rangers alike to determine real-time dynamic patrol strategies. A black box optimization with neural networks has also been presented by Gurumurthy et al. [26]. These methods all require extensive (historic) data sets, preliminary data manipulation, and extensive knowledge of the terrain.

The recent models proposed by Xu et al. [21] and Gholami et al. [23] also avoid adversary modelling. However, instead of looking at target characteristics for attackability determination, they define a game-theoretic behaviour model for the defender that does not require prior data. The defender does learn an optimal defensive strategy during the game though, regardless of the behaviour the attacker exhibits. This research continues on this specific online approach while avoiding adversary modelling.

B. *Matheuristic Path Planning*

When the question is asked "what is the optimal route for a vehicle given a specific set of constraints?" the resulting problem is always a variant of the Vehicle Routing Problem (VRP) [30]. The case of a single vehicle maximising its reward over a closed route (i.e. returning to the starting point) is known as the Orienteering Problem (OP). The OP, which is NP-hard, is a well-studied problem, and many exact and (meta)heuristic methods have been proposed to solve it [31]. The problem is formulated as an integral problem where a path has to be found on a graph of nodes connected by arcs. All nodes have a certain reward that is collected when the node is visited, and the arcs induce a certain cost when they are part of the route. In the aerial surveillance case, the graph is considered complete, meaning all nodes are interconnected.

Even though the aerial wildlife surveillance problem is presented as a GSG, the planning aspect is considered an OP that has to be solved repeatedly on a graph with rewards that change every round. However, since GSGs assume a defender with limited resources (i.e., it cannot defend all targets in a single round), the possible solution space for the OP is limited. This makes the combination of the defender strategy algorithm and the path planning algorithm in MEOMAPP a method considered a *matheuristic* for solving routing problems [32]. More specifically, it can be classified as a two-phase decomposition approach [33], where the first phase is considered selecting a subset of the nodes in the graph and the second phase is solving the OP on the reduced graph.

This research does not focus on improving solution methods for an OP but it was considered noteworthy that this work is on the intersection between GSGs and operations research.

C. *Agent-Based Modelling and Simulation*

In the majority of referenced literature in this paper, the wildlife surveillance problem is represented as a Multi-Agent System (MAS), wherein GSGs provide a framework to model the agent's interactions. The presence of autonomous actors in the system that interact with each other makes Agent-Based Modelling and Simulation (ABMS) the most appropriate technique to implement and study this model. The ABMS technique enables us to model a natural representation of a system, provide flexibility to modify the system model, and examine emergent outcomes resulting from interactions between autonomous, individual entities with dynamic, adaptive behaviours and heterogeneous characteristics [34]. This is a suitable modelling framework for wildlife conservation in general since models can be specified realistically and dynamically, including changes in environmental conditions and animal movements [35]. This enables them to study externalities related to natural resource management [36].

III. Problem Formulation

In this section, we describe the conceptual formulation of the practical problem of aerial wildlife surveillance. The formulation is similar to the problem formulation by Xu et al. [21] and Gholami et al. [23], as this research aims to extend their proposed solution model.

A. *Game Setup*

The components of the gamified system are the wildlife area that is to be surveilled (the "targets"), the drone that performs the surveillance flights (the "defender"), and the poachers the drone aims to observe (the "attackers"). The entire area is discretized by square grid cells, resulting in set $[N]$ consisting

of N targets. The diagonal of the grid cells is assumed to be the width of the drone camera's Field of View (FoV) in order to always entirely cover a target when choosing it for the route. We assume the drone's height to be constant, ensuring a constant FoV.

The surveillance planning problem is regarded as an infinitely repeated security game between an attacker and a defender. Each round t, m attackers each choose a target to attack. Simultaneously, the defender chooses a surveillance flight to cover k targets specified according to a strategy $v_t \in \{0,1\}^N$. Vector v_t is a binary vector denoting waypoints of the surveillance path by entry $i = 1$ if target i is selected as a waypoint for the flight path. The targets that are observed resulting from this strategy are indicated by binary vector c_t with $|c_t| \geq k$. Targets that are only partially observed due to the nature of the flight path have a chance of being included in c_t equal to the ratio of grid cell area covered by the defender. Similarly to v_t, the attacker strategy is denoted as a_t, where entry $i = 1$ if target i is attacked by the attacker. The path the attacker takes is not taken into account. Also, it is assumed that the attacker remains at the same location for the entire duration of the round. Given that target i is attacked, the defender gets utility U_i^c if target i is covered by the defender, and U_i^u if i is not covered by the defender. It is assumed that covering a target is better than not covering it, which we formalise by stating $U_i^c > U_i^u$.

B. *Information Access and Player Behaviour*

In wildlife surveillance, it is usually unknown to the defender and the attacker what the specific value of a target is as it depends on unknown and/or variable environmental factors and actor-specific preferences. Also, the other players' behaviour is difficult to predict completely, as players can have different knowledge or behave irrationally. Given this information gap, the approach taken for this and previous models is to assume that the defender is unaware of prior information regarding payoffs and attacker behaviour. It can only observe utilities of targets when they are observed. Also, there is no behaviour model of the attacker required for the defender to learn a strategy, since it will adapt to any kind of attacker behaviour. Furthermore, it is required for the attacker that he can only observe the defender when being observed by the defender himself in the current round. That way he is not able to evade the defender during the same round. Finally, we assume a perfect observation from the defender, meaning once the defender covers 100% of an attacked target, the attacker is observed.

C. *Utility and Game Objective*

Given the attacker strategy a_t and the defender observations c_t in round t, the defender's utility at round t is defined as

$$u\left(c_t, a_t\right) = \sum_{i \in N} c_{t,i} a_{t,i} U_i^c + \sum_{i \in N} \left(1 - c_{t,i}\right) a_{t,i} U_i^u \quad (1)$$

where the first term denotes the utility of the covered targets and the second term the utility of the uncovered targets. Both terms are dependent on a_t. The equation can be rewritten as

$$u\left(c_t, a_t\right) = c_t r_t\left(a_t\right) + C\left(a_t\right) \quad (2)$$

with $r_{t,i} = a_{t,i}\left[U_i^c - U_i^u\right]$ and $C\left(a_t\right) = \sum_{i \in [n]} a_{t,i} U_i^u$. This notation helps understanding that the defender's utility at round t is dependent on the attacker's moves during that round. The objective of the security game is to minimise the overall total regret R_T of the defender, defined by

$$\begin{aligned} R_T &= \max_{c \in \mathcal{C}} \sum_{t=1}^{T} u\left(c, a_t\right) - \mathbb{E}\left[\sum_{t=1}^{T} u\left(c_t, a_t\right)\right] \\ &= \max_{c \in \mathcal{C}} \sum_{t=1}^{T} r_t c - \mathbb{E}\left[\sum_{t=1}^{T} r_t \cdot c_t\right] \end{aligned} \quad (3)$$

The first term is the utility of the optimal strategy for round t in hindsight, with $\mathcal{C}$ being the set of all possible observation vectors c. The second term is the expected value of the defender's utility. This notation is consistent with previous online learning theory literature. As noted in Xu et al. [21], the underlying notion of this regret formulation is that it is typically impossible to learn the optimal (adaptive) defender strategy v_t. The reason for this is that the attacker can choose a_t independent from previous actions or even adversarially to the defender. Therefore, the optimal strategy at round t can be independent from history. Without complete knowledge of the attacker behaviour or the environment, there is no way to predict the optimal strategy v_t and it is thus impossible to learn the optimal adaptive strategy.

However, with access to previous observations, it is possible to learn the best strategy in hindsight. The idea behind this problem formulation is that after more and more rounds, the performance of the best strategy in the next round will be affected less and less by a_t, no matter how adversarial (i.e. only caring about minimising the defender's utility) the attacker plays.

IV. Agent-Based Model

The agent-based model forms the framework for the different methods used to solve the formulated problem. These methods and their parameters are represented by the characteristics of the environment and the agents' inputs, internal states, and cognitive models. The representations of the models in Xu et al. [21] and Gholami et al. [23] serve as the baseline of the model formulation. A diagram providing an overview of the agent-based model is given in Figure 2. The following subsections present the properties of the environment and all agents in the model as well as the agents' interactions. Verification of the model is discussed in the last subsection.

A. *Environment Specification*

The environment is defined by a space and time wherein the agents are situated. In this model, one time step equals one round of the GSG, where the attacker attacks one target and the defender performs one surveillance flight.

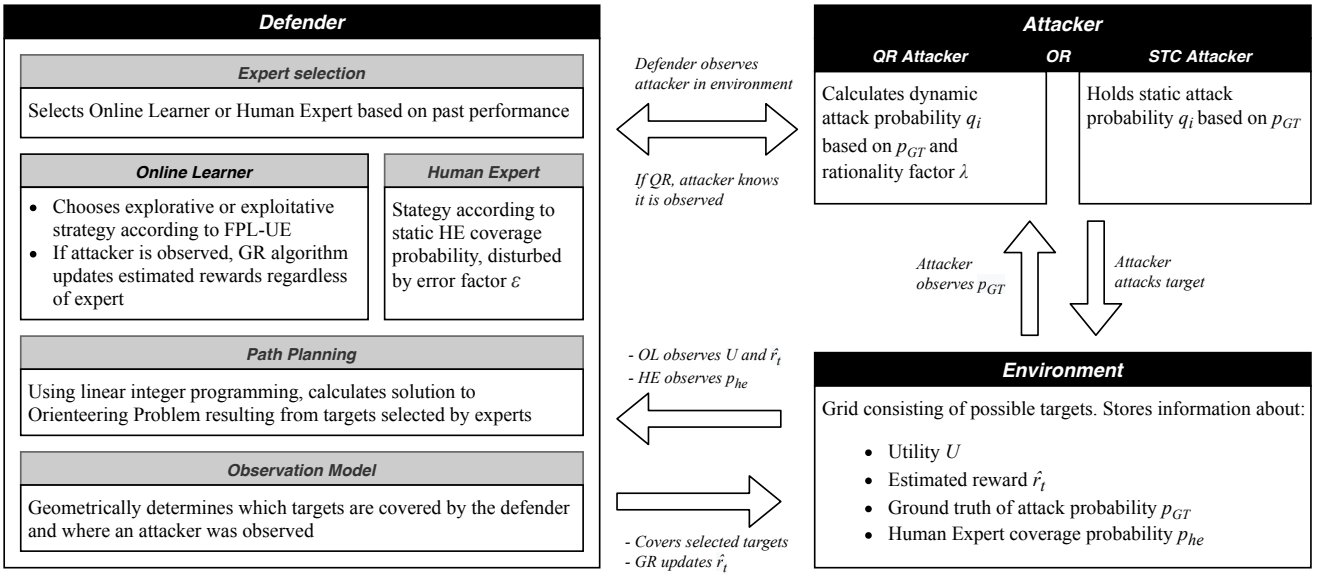

Figure 2. Diagram of the components of the Agent-Based Model and their interactions that occur during one round.

As stated in the previous section, the environment space is discretized by a rectangular grid. The grid cells that result from this representation are all potential targets for an attacker to attack and the defender to defend. The targets i are modelled as patches in the ABM, meaning they are stationary agents. We define $[N]$ to be the set of all targets i. One target is assigned to be the $base$, which means it is the location where the defender will start and end its patrol strategy. When the model is initialised, every target is assigned four numerical characteristics:

1) Utility: Every cell is assigned a utility U_i^c for when it is covered and a utility U_i^u for when it is uncovered. We assume that the utilities are unknown for the defender and that $U_i^c > U_i^u$. For normalisation, we define the values of U_i^c and U_i^u to be within $[-0.5, 0.5]$. This way the maximum regret of the defender per round of the game is at most 1 for each attacker.

2) Estimated reward: The estimated reward $\widehat{r}_{t,i}$ for cell i at round t is initialised as $\widehat{r}_{1,i} = 0$ and indicates the estimated reward of covering that cell in the following step of the game. As explained further down, the estimated reward of a target is incrementally updated by the drone's Online Learner algorithm at every step (see Figure 2), and thus the order of magnitude of the estimations increases gradually throughout the game. It is therefore only compared to the estimated reward of other targets during that same round.

3) Attackability score: The downside of an online learning algorithm without historical data is that it takes time for a model to learn and perform as desired. However, more information about the area to be surveilled is sometimes available in the form of human expertise based on knowledge of the area and the people that live there [26]. We make use of this information in the form of an attackability score. The attackability score is an integer between 0 and 3 assigned to each cell by an expert who is familiar with the environment. It is a positive ratio scale determined with the expert that indicates the likelihood of attacker presence on a specific cell. We convert this score to an attack probability by modifying the standard normalised exponential function (left in Equation 4) to a version that takes into account the variation of discretization (right).

$$\sigma(\mathbf{z})_i = \frac{e^{\beta z_i}}{\sum_{j=1}^{K} e^{\beta z_j}} \Longrightarrow p(\mathbf{a})_{GT,i} = \frac{\sqrt{N}^{a_i}}{\sum_{j\in[N]} \sqrt{N}^{a_j}} \quad (4)$$

R. D. Luce [37] first used the normalised exponential function in decision theory for relative preferences in his Choice Axiom. It has seen multiple applications in psychology [38] and game theory [39, 40] relating to human choices and utility representation [41]. The general idea behind it is that humans have an initial intuition to map numbers onto space logarithmically. We propose setting $\beta = \log\left(\sqrt{N}\right)$, which is equivalent to using N (total number of targets) as base of the exponentiation instead of e. This concentrates the probability distribution more around the positions of the largest input values. Taking a base that varies according to the number of cells also prohibits the higher probabilities to dilute when the grid is discretized further. We chose to select $\sqrt{N}$ instead of N since humans' tendency to logarithmically project numerals is based on a linear distribution. Therefore we linearize the increase of N, which exhibits quadratic growth when we linearly increase the dimensions of the grid. The resulting probability distribution p_{GT} is the ground truth for the behaviour models for attackers, further explained in Section IV-C, and for the defence strategies of the Human

Expert (HE), explained hereafter.

4) Human Expert Coverage Probability: The MEOMAPP algorithm incorporates a self-evaluation procedure for the defender where it compares the performance of an online learning expert with the performance of a human expert (HE). To simulate the performance of a human expert defence strategy, we propose the following. The human expert strategy consists of a set S_t of n targets selected based on the probability distribution p_{he}. This distribution is used in the defender's path planning algorithm to calculate a patrol route. To represent the different levels of inaccuracy of the human expert, we approximate a probability distribution p_{he} with a certain mean absolute error (MAE) as follows:

$$p_{he,i}(\varepsilon) = p_{GT,i} \pm \epsilon \quad , \quad \epsilon \sim \mathcal{N}(\varepsilon, \varepsilon/4) \tag{5}$$

with a resulting MAE of the probability distribution approximately equal to ε. Error factor ϵ is drawn from a normal distribution $\mathcal{N}(\varepsilon, \varepsilon/4)$ with mean ε and variance $\varepsilon/4$. The $+$ or $-$ is as such that $p_{he,i} \in [0, 1]$, with a random choice if both options are viable.

B. Defender Specification

The defender is a reactive although complex agent. It is characterised by the range R or the total distance it can travel in a single round. The defender always departs and returns to the $base$ target, Furthermore, it incorporates four cognitive models: *1)* an online learning game-theoretic expert (OL) that calculates the reaction to the attacker's moves, *2)* a mathematical integer linear program to calculate a spatially optimised patrol route within its planning constraints, *3)* an observation model, and *4)* an expert selection algorithm that chooses between a human expert and the OL expert as the preferred choice for next round's strategy.
A theoretical argument for the defender's parameter values is presented at the end of this subsection.

1) Online Learner: The online learning algorithm presented in Algorithm 1 is capable of generating a defender strategy without any prior knowledge. It is based on the FPL-UE algorithm proposed by Xu et al. [21], wherein a "Follow the Perturbed Leader" and a "Uniform Exploration" element can be distinguished.
The FPL element evaluates the perturbed estimated reward $\widetilde{r}_{t,i}$ for each target i at round t. Let $\widehat{r}_t$ be the vector of the pure estimate rewards at round t, and let $z_t = (z_{t,1}, \ldots, z_{t,n})$ be a random noise vector such that each $z_{t,i} \sim exp(\eta)$ is independently drawn from the exponential distribution $exp(\eta)$. At each round, the algorithm then chooses n targets, from all targets $[N]$, with the highest perturbed estimated reward $\widetilde{r}_{t,i} = \widehat{r}_{t,i} + z_{t,i}$, as formulated by Equation 6. This set is called S_t.

$$S_t = \underset{S \subset [N]}{\arg\max} \left\{ \sum\nolimits_{i \in S} \widetilde{r}_{t,i} \;\middle|\; |S| = n \right\} \tag{6}$$

In this case, the noise vector $z_{t,i}$ represents the uncertainty of the reward estimation and thereof dependent target selection. It also results in unique estimated reward values for every target, which prohibit that the MILP solver selects identical sequences of targets (especially in the first rounds when most targets have $\widehat{r}_{t,i} = 0$). The FPL element is the *exploitative* element of the defender strategy.

The UE element, which is the *explorative* element of the defender strategy, also selects n targets to form S_t but does it randomly and uniformly. For the randomly selected targets in S_t, a noise vector z_t is drawn independently from $exp(\eta)$ as well for the same reasons as for the exploitative strategy. Since these targets were selected regardless of the value of their estimated reward, the perturbed estimated reward is set to equal the noise factor: $\widetilde{r}_{t,i} = z_{t,i}$. In every round, taking a random explorative step happens with probability γ, resulting in a complementary probability $(1-\gamma)$ to pursue an exploitative strategy. The goal of the exploitative element is to maximise the total utility over time, whereas the goal of the explorative element is to learn which strategy is the best against a particular attacker.

Up to now, the online learner follows the FPL-UE algorithm to determine a set of nodes S_t at every round by either following an exploitative or an explorative strategy. The set S_t is used by the MILP to calculate the flight path strategy v_t (as described in section IV-B2), where the targets in S_t are potential waypoints for the flight path. Once flight path strategy v_t is determined and applied at round t, we define set O_t as the targets where the defender observed an attack when executing the strategy during round t.

Knowing this, we can now update the estimated reward values for the next round $\widehat{r}_{t+1,i}$ as follows:

$$\widehat{r}_{t+1,i} = \widehat{r}_{t,i} + \tfrac{r_{t,i}}{p_{t,i}} \mathbb{I}(t,i) \quad \forall i \in O_t \tag{7}$$

where $p_{t,i}$ is the probability that target i was observed by the defender within round t and $\mathbb{I}(t,i)$ is an indicator function indicating if a target was observed by the defender, with $\mathbb{I}(t,i) = 1$ if target i was observed and $\mathbb{I}(t,i) = 0$ otherwise. In online literature, the term $\frac{r_{t,i}}{p_{t,i}}\mathbb{I}(t,i)$ is preferred over directly using $r_{t,i}\mathbb{I}(t,i)$, since it is an unbiased estimator of $r_{t,i}$: $\mathbb{E}\left[\frac{r_{t,i}}{p_{t,i}}\mathbb{I}(t,i)\right] = r_{t,i}$. Note that this corresponds to updating the estimated reward for targets that were attacked *and* defended, and keeping the estimated reward for the other targets the same in the next round.

To efficiently estimate $p_{t,i}$, which is unknown and hard compute exactly, Neu and Bartók [22] proposed a method to calculate the value of $1/p_{t,i}$ called Geometric Re-sampling (GR), presented in Algorithm 2. The method works by simulating defender strategies until the targets that were attacked and defended in round t are defended again by the simulated round. The number of simulations required until target i is defended for the first time follows a geometric distribution with mean $1/p_{t,i}$. The probability of observation $p_{t,i}$ is thus estimated by the number of simulations it requires to defend target i. However, theoretically, the number of

simulations can be infinitely large, so the GR algorithm truncates the number of simulations with a finite quantity M.

Algorithm 1 Online Learner

Parameters: $\gamma \in [0,1], n \in \mathbb{N}, \eta \in \mathbb{R}^+, M \in \mathbb{Z}^+$

for t = 1, ..., T **do**
  Sample $\alpha \in 0, 1$ such that $\alpha = 0$ with prob. γ
  if $\alpha = 0$ **then**
    Let $S_t \subset [N]$ be a set of n randomly selected targets
    Draw $\widetilde{r}_{t,i} \sim exp(\eta)$ independently for all $i \in S_t$
  else
    Draw $z_{t,i} \sim exp(\eta)$ independently for all $i \in [N]$
    Set $\widetilde{r}_{t,i} \leftarrow \widehat{r}_{t,i} + z_{t,i}$
    Let $S_t \subset [N]$ be the set of n targets with $max(\widetilde{r}_{t,i})$
  end if
  Let v_t be $\mathcal{P}(S_t)$
  Adversary picks $r_t \in [0,1]^n$ and defender plays v_t
  Defender observes O_t
  Run GR$(\eta, M, \widehat{r}, t)$: estimate $\frac{1}{p_{t,i}}$ as $K(t,i)$
  Update $\widehat{r}_i \leftarrow \widehat{r}_i + K(t,i) r_{t,i}$;
end for

Algorithm 2 Geometric Resampling

Input: $\eta \in \mathbb{R}^+, M \in \mathbb{Z}^+, \widehat{r} \in \mathbb{R}^n, t \in \mathbb{N}$
Output: $K(t) := \{K(t,1), \ldots, K(t,n)\} \in \mathbb{Z}^n$

Initialize $\forall i \in [N] : K(t,i) = 0; k = 1$
for $s = 1, 2, \ldots, M$ **do**
  Repeat lines 2 - 13 in alg. 1 once to produce $\widetilde{O}$ as a simulation of O_t with a_t.
  for all $i \in O_t$ **do**
    if $s < M$ **and** $i \in \widetilde{O}$ **and** $K(t,i) = 0$ **then**
      Set $K(t,i) = s$;
    else if $s = M$ **and** $K(t,i) = 0$ **then**
      Set $K(t,i) = M$;
    end if
  end for
  if K(t, i) > 0 for all $i \in O_t$ **then** break
end for

2) Path Planning: The patrolling strategy v_t is calculated as a solution to a symmetric Orienteering Problem, formulated by the mathematical program $\mathcal{P}(S)$ in Equation 8. It is an integer linear programming problem applied to targets i in S_t, which represent the nodes of a network where the perturbed estimated rewards $\widetilde{r}_{t,i}$ represent the profits collected if a node is visited. The network's edges are defined as the arcs $a_{i,j}$ between targets i and j, with a length of $d_{i,j}$. Equation 8.a limits the total distance travelled by the defender to its range R. To every target the defender goes, it also has to leave from, which is constrained by Equation 8.b. A target can only be visited once, meaning only two arcs can connect to it. This is constrained by Equation 8.c. Constraint 8.d ensures that no subtours (i.e., tours that are not part of the tour that includes the base) are included in the solution. If an arc $a_{i,j}$ is selected, the difference between u_i and u_j is exactly -1 if arc $a_{i,j}$ is selected in the strategy, where u_i and u_j are the orders at which the targets are visited. Subscript $base$ indicates the target where the defender starts and ends its round. that target's inclusion in the path is ensured by constraints 8.e and 8.f. These are not strictly speaking necessary when Equation 8.d is applied (where the $base$ is the only target left that can "close" the loop), but they increase the computational performance.

$$v_t = \arg\max_{v \in \mathcal{V}} \sum_{i,j \in S_t} a_{i,j} \widetilde{r}_i \tag{8}$$

subject to

$$\sum_{i,j \in S_t} a_{i,j} d_{i,j} \leq R \qquad i \neq j \quad \text{(a)}$$

$$\sum_{i \in S_t} a_{i,j} = a_{j,i} \qquad \forall j \in S_t; i \neq j \quad \text{(b)}$$

$$a_{i,j} + a_{j,i} \leq 1 \qquad \forall i,j \in S_t; i \neq j \quad \text{(c)}$$

$$u_i - u_j \leq |S_t| (1 - a_{i,j}) - 1 \qquad \forall i,j \in S_t; i \neq j \neq base \quad \text{(d)}$$

$$\sum_{i \in S_t} a_{i,base} = 1 \qquad i \neq base \quad \text{(e)}$$

$$\sum_{i \in S_t} a_{base,i} = 1 \qquad i \neq base \quad \text{(f)}$$

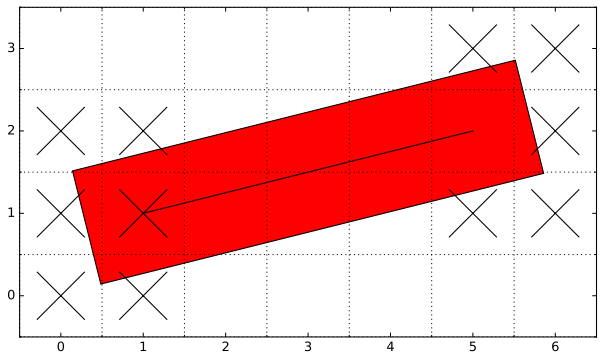

Figure 3. Coverage between target $(1,1)$ and $(5,2)$.

3) Observation Model: After v_t has been calculated by the path planning algorithm, it is necessary to determine which targets are covered and whether an attacker has been observed. Based on the assumption that the defender's view is as wide as the diagonal of a cell, we draw a rectangle of width $w = \sqrt{2l^2}$, where l is the width of a target, and length $d_{i,j} + w$, where d_{ij} is the distance between targets i and j, so that arc $a_{i,j}$ coincides with the longest centerline of the rectangle. Afterwards, for every target that has an overlap with this rectangle, the fraction $frac_i$ of the observed area of the target over the total area of the target is calculated. The exemptions to this are the starting point i and the targets right next to i and j that are not in the line of the path, in order to prevent them to be counted twice. As an example, a schematic of coverage between target $(1,1)$ and $(5,2)$ is shown in Figure 3, where $frac_i > 0$ for targets $(2,0), (2,1), (2,2), (3,1), (3,2), (4,1), (4,2), (4,3)$ and $(5,2)$.

This is done for all arcs $a_{i,j}$ that are part of the solution calculated by the path planning algorithm in Equation 8, resulting in a set C_t consisting of targets i that have been (partially) covered by the defender at round t.
The probability that an attacker is observed by the defender on target i, if an attacker was on that target, is equal to the fraction of the area observed. In other words, target i is considered to be in set O_t (targets where an attack has been observed at round t) following the next equation:

$$P(i \in O_t | i \in C_t \cap A_t) = frac_i \tag{9}$$

where A_t is the set of targets i that are attacked.

4) Expert Selection: The following expert selection algorithm is proposed to enable MEOMAPP to select the best performing expert to decide on a strategy. Formulation wise, it is wrapped around the core algorithms 1 and 2 as presented in algorithm 3. First, a constant θ is defined which is compared to the value of variable γ. Only after the value of γ drops below threshold θ (line 4) we consider the online learning algorithm to have learned enough to compare it to the HE. We chose γ as a threshold measure because it depends on k and m as well, which allows us to maintain the same value for θ throughout different game settings.
To compare both the experts' performance, cumulative reward values r_{ol} and R_{he} are initialised as 0 and updated after every round t. The performance is then evaluated by comparing the average reward over the number of rounds where that expert has been selected, n_{ol} and n_{he}. If the human expert is chosen to be the best expert, S_t is determined by p_{he} as presented in Section IV-A3. If the online learning expert is selected, S_t is determined by Algorithm 1. The patrol strategy v_t is then determined by $\mathcal{P}(S_t)$ resulting in observed attacked targets O_t and the collected rewards are updated accordingly for each expert. Note that until γ reaches the threshold θ the performance of the human expert, which has not been evaluated by then, is synchronised with the performance of the online learning expert (lines 20 - 22).
The idea behind this expert selection algorithm is that given the online learner expert has learned sufficiently, the algorithm will be able to decide whether it is more successful to follow the human expert or the online expert. In this case, it will differentiate between the human expert, whose performance results from the potentially imperfect interpretation of the environment and/or historical data, and the online learner without any prior knowledge.

Defender's parameter values: From the theoretical properties of the FPL-UE algorithm stated by Xu et al. [21], we know that the total regret R_T (i.e., the difference between the performance of FPL-UE and that of the best fixed patrol path in hindsight) is proven to be upper bounded as:

$$R_T \leq \gamma m T + 2Tke^{-M\frac{\gamma}{[N]}} + \frac{k(\log N + 1)}{\eta} + \eta m T \min(m, k) \tag{10}$$

Algorithm 3 The MEOMAPP Algorithm

Parameters: $\gamma \in [0,1], n \in \mathbb{N}, \eta \in \mathbb{R}^+, M \in \mathbb{Z}^+, \theta \in \mathbb{R}^+$
1: Initialise $\widehat{r} = 0, r_{ol} = 0, r_{he} = 0, n_{ol} = 0, n_{he} = 0$
2: Pick a value θ as a threshold for γ for which the Online Learner is considered good enough;
3: **for** $t = 1, \ldots, T$ **do**
4: **if** $\gamma \geq \theta$ **or** $\frac{r_{ol}}{n_{ol}} > \frac{r_{he}}{n_{he}}$ **then**
5: Let S_t be computed by lines 2 - 10 in alg. 1;
6: $n_{ol} \leftarrow n_{ol} + 1$
7: $f = 0$
8: **else**
9: Let S_t be determined by the HE where $\widetilde{r}_{t,i} = p_{he,i}$;
10: $n_{he} \leftarrow n_{he} + 1$
11: $f = 1$
12: **end if**
13: Let v_t be $\mathcal{P}(S_t)$;
14: Adversary picks $r_t \in [0,1]^n$ and defender plays v_t;
15: Defender observes attackers at O_t;
16: **for** $i \in O_t$ **do**
17: $r_{he} \leftarrow r_{he} + f r_i$
18: $r_{ol} \leftarrow r_{ol} + (1-f) r_i$
19: **end for**
20: **if** $\gamma \geq \theta$ **then**
21: $n_{he} \leftarrow n_{he} + 1$
22: $r_{he} \leftarrow r_{ol}$
23: **end if**
24: Run GR$(\eta, M, \widehat{r}, t)$: estimate $\frac{1}{p_{t,i}}$ as $K(t,i)$;
25: Update $\widehat{r}_i \leftarrow \widehat{r}_i + K(t,i) r_{t,i}$;
26: **end for**

where upper bound $\mathcal{O}(\sqrt{kmT \min\{m,k\} \log N})$ can be obtained by taking $\eta = \sqrt{\frac{k(\log N+1)}{mT \min\{m,k\}}}$, $\gamma = \frac{\sqrt{k}}{\sqrt{mT}}$ and $M = N\sqrt{\frac{mT}{k}} \log(Tk)$. This means that the values of η, γ and M depend on the total number of targets (N), the number of attackers (m), the number of intentionally protected targets (k), and the number of rounds T that have passed, the contribution of the latter resulting in a gradual decline of the values of η and γ over time. This can be interpreted as a decline in uncertainty because of the decline in noise z_t and the lower probability of engaging in an explorative strategy respectively.
The number of intentionally protected targets k is the number of targets that are selected as waypoints in the flight path $|v_t|$. This value is not known beforehand, but since the first round is explorative regardless, any number larger than 0 can be chosen as an initial value. After the first round, k is calculated as the average value of $|v_t|$ over time.
In theory, the value of M, the maximum number of simulations in the GR algorithm, is very high relative to T and can result in extremely long running times. For example, for a 500-step simulation with $N = 100, m = 5$ and $k = 15$, the GR algorithm runs up to $M = 11,519$ times in the worst-case scenario. However, as Neu and Bartók [22] theorise a lower expected number of samples in practice, we limit the maximum amount of GR simulations to 100. During the experiments, this number of samples was never reached.

C. Attacker Specification

To simulate attacker behaviour we chose two commonly seen behaviour models: Stochastic behaviour (STC) and Quantal Response behaviour (QR) [14].

The STC attacker model chooses to attack target i based on a stationary attack probability q_i per target i. We assume this probability to be equal to the ground truth attack probability $p_{GT,i}$ of target i presented as the basis for the human expert in Section IV-A3, as this is the best information available.

$$q_i = p_{GT,i} \tag{11}$$

The QR attacker model simulates non-stationary attacker behaviour by observing and responding to the defender strategy. The probability that an attacker will attack target i at round t is given by:

$$q_i = \frac{e^{\lambda U_i^a}}{\sum_{j\in[N]} e^{\lambda U_j^a}} = \frac{e^{\lambda(x_i P_i^a + (1-x_i)R_i^a)}}{\sum_{j\in[N]} e^{\lambda(x_j P_j^a + (1-x_j)R_j^a)}} \tag{12}$$

Parameter $\lambda \in [0, \infty]$ represents the rationality level of the attacker. A lower value for λ indicates lower rationality (resulting in a more uniform q_i) and a higher value indicates higher rationality (resulting in a reward maximising q_i). Parameter $U_i^a = x_i P_i^a + (1 - x_i)\, R_i^a$ is the attacker's expected utility for target i. It depends on the likelihood the target is defended x_i, and on the reward R_i^a and penalty P_i^a for the attacker associated with the target. R_i^a is obtained by normalising the GT probability p_{GT} to a value between $[0, 10]$ in accordance with QR experiment settings in previous literature [11]. We assume penalty $P_i^a = -10$ for all targets since getting caught by the defender is the worst that can happen on any target.

The likelihood x_i that target is defended is not the same as probability p_i estimated by the GR algorithm, but a likelihood calculated by the attacker based on how often it is caught on target i. Previous literature does not explain how x_i is calculated. We propose to initialise $x_i = 0$ and to recalculate it at every round t as:

$$x_i = \frac{\sum_{t=1}^{T} a_{t,i}\mathbb{I}(t,i)}{T} \qquad \forall i \in [N] \tag{13}$$

where we note that this only holds because we make the additional assumptions: (i) The attacker is not initially aware of the fact that the area is under surveillance. (ii) The attacker does not know where or when the defender has been if the attacker was not observed by the defender. (iii) If an attacker was observed in any previous round, all attackers know about this in the next round.

For both models, after the probability q_i of the attacker choosing a target i in round t is determined, one target is picked per attacker based on that likelihood. Furthermore, it is important to note that the attacker is modelled to visit only one target in a single round, meaning no route to and from the target are simulated.

The fact that p_{GT} is used as a foundation to calculate q_i for both attacker models as well as the coverage preference p_{he} of the HE means that the accuracy ε of the HE relates to how good the human expert is in assessing the attacker behaviour.

D. Validation and Verification

As for agent-based models in general, the bottom-up nature of the building process of MEOMAPP led to validation being applied during the model construction itself [42]. To assure the validity of the model as a whole, every model component was validated individually, including its output of information to other components. The goal of this research is to evaluate the application of an online learning defender strategy for GSG in a simulated environment to estimate its performance in a realistic scenario. The representation of the environment, the attackers, and the defender in the agent-based model has been performed with the support of wildlife surveillance experts whose contribution ensured further validity of the model. It is important to note that a higher validity could be obtained for more specific cases depending on the information available. The ultimate test to validate the model would be to perform experiments in the real environment.

Verification of the model was performed at different levels. At the code level, compiler errors were resolved within Spyder, the integrated development environment (IDE) chosen for this research. At the unit level, error-oriented testing [43] has been performed by plausibility tests [44] on parameter values and on results of intermediate computation mechanisms. Attention has also been paid to avoid issues arising from floating-point arithmetic within the computations. At the system level, conceptual verification was performed by observing whether the results matched expectations regarding convergence.

V. Numerical Evaluation

This section describes how we implemented the proposed model, the real-world case we simulated, the simulation setup, the derivation of the model's parameter values for this specific case, and the results from the simulations.

A. Algorithm Implementation

The simulation model was written in Python using Mesa, a Python-specific agent-based modelling framework [45]. The code is written on compliance with the PEP 8 style guide for Python code [46]. It is available in the Delft University of Technology Gitlab repository. For the ILP component, the open-source module PuLP [47] was used to generate the problem file. To solve the ILP problem, the Gurobi™[48] solver was called using an academic licence. It is however interchangeable with open-source solvers readily available in the PuLP toolkit. All simulations that are discussed later have been run using a machine with a 1.2 GHz Intel® Core™ i7-3610QM CPU with 7 available cores and 12GB RAM. The runtime for the presented experiments can be found in Table III.

B. Case Study

To examine the performance of MEOMAPP, we selected a case together with industry experts to simulate wildlife surveillance in a real-world setting. The domain to surveil is the Aloegrove Safari Park in Namibia and is approximately 10 by 10 km in size (see Figure 4). The attackability values

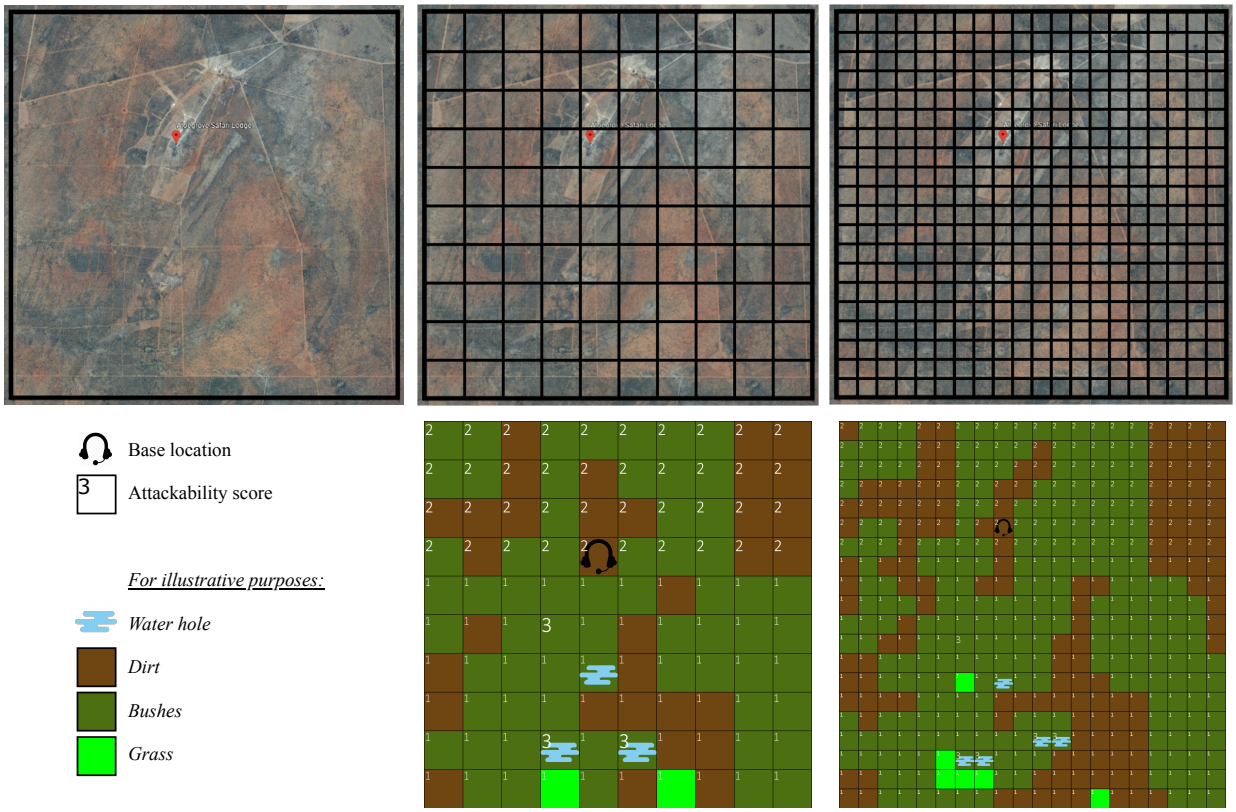

Figure 4. Schematic of Aloegrove Safari Park and its implementation into MEOMAPP *(image from Google Earth - CNES/Airbus Maxar Technologies).*

have been assigned by an expert familiar with the domain. The expert based the attackability values on his interpretation of the accessibility of the terrain, location of fences or barrier, and the location of animal enclosures or water holes. The colour scheme on the map is purely for illustrative purposes. Contrary to simulations in previous research where the values for utilities U_i^c and U_i^u are randomly selected, we chose to define $U_i^c = 0.5$ and $U_i^u = -0.5$. The reasoning behind this is that we consider the utility to be a result primarily of whether a poacher has been observed or not. This is regardless of the damage the poacher has or could have done. Also, defining the utility values specifically eliminates a random factor from the regret equation, which makes the results less prone to random variations.

The drone range is 25 km and we chose to have the algorithm select 20 possible waypoints for a surveillance flight path at every round ($n = 20$). It is not important to determine when or how long a surveillance flight takes place, as long as we assume that the attacker(s) remain(s) at the attacked location(s) for the duration of the surveillance flight. For a grid discretization of 10 by 10 cells we assume a field of view of the drone of approximately 1.4 km wide. For a grid discretization of 20 by 20 cells, the field of view is considered 0.7 km wide, which in practice means that the drone flies lower with a lower resolution camera and therefore has a reduced field of view.

C. *Key Performance Indicators*

To evaluate MEOMAPP's performance, we look at the following three indicators. These are the average regret over time, the average distribution of employed strategies, and the average distribution of observed attacks.

As in previous research, the overall performance is measured by the average regret over time R_T/T. Since we chose to limit the values of U_i^c and U_i^u to 0.5 and -0.5 respectively, the regret value directly relates to the number of targets attacked. Since every attacker attacks one target every round, the regret can be at most m. and at least 0. Experiment results expressed as regret from previous research [21, 23] cannot be used to directly compare MEOMAPP's regret values for two reasons. First of all, information about their specific payoff structure and attacker behaviour is incomplete. Without that information, it is not possible to reproduce their experiments. Secondly, we chose a specific utility structure with extreme values, which will produce relatively higher regret values for any experiment setup. Additionally, the grid discretization used by Gholami et al. [23] was only 5x5 cells, which we deem insufficient for real-world approximations.

Furthermore, we look at the distribution of employed strategies: explorative, exploitative, or defined by the human expert. This indicates which expert is superior in which situation, and gives an insight into whether the expert selection algorithm works.

Finally, in order to evaluate the accuracy of MEOMAPP we look at the distribution of observed attacks that took place on intentionally visited targets (the waypoints of routes selected by the HE and the exploitative OL) and on coincidentally visited targets (explorative routes and the targets between waypoints).

D. Simulation Setup

The parameter space can be varied in seven dimensions:

1) Grid size/discretization
2) ε of the HE
3) Attacker type (STC or QR with different λ)
4) Number of attackers m
5) Number of possible waypoints n
6) Expert selection parameter θ
7) Drone range R

Since it is impractical to evaluate all possible combinations, we select a base setting with parameters that remain constant and manipulate the remaining parameters to deduce their impact on MEOMAPP's performance.

The parameters that remain constant throughout all experiments are n and θ. The number of possible waypoints $n = 20$ is an arbitrary choice. The logic behind it is that it should not be too large for 1) computational reasons and 2) a twisty trajectory which is impractical for the drone. It should also not be too small in order to evaluate enough points of interest for the surveillance strategy. After some iterations, we chose $n = 20$ as it satisfied both accounts. Variations in n and the planning algorithm in general, are included as recommendations for future research.

For the baseline model we propose the following: The attacker types are an input that would not be required in a real-world setting. Therefore we decide to test the performance with an STC attacker as an example of stationary behaviour, and with one QR attacker with a specific λ as an example of adaptive non-stationary behaviour. To choose a value for λ, ten games with one QR attacker's λ value ranging from 0.1 to 1.0 with 0.1 increments were simulated without the HE. The 0.1 to 1.0 range was chosen to include λ values found and used in previous research by Nguyen et al. [14] and Gholami et al. [23]. Additionally, one game with an STC attacker was simulated as well. Each simulation lasted 500 steps and the results are presented in Table I. It can be observed that the final average regret is inversely related to the value of λ. This means that the performance of MEOMAPP is better the more rational the attacker is.

Table I
AVERAGE R_T FOR SIMULATIONS WITH $N = 100$, $m = 1$, $T = 500$ FOR VARYING λ OF QR ADVERSARY AGAINST OL EXPERT.

λ	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9	1.0	STC
R_T	0.70	0.66	0.60	0.52	0.48	0.43	0.39	0.35	0.35	0.30	0.50

Also, the OL's performance against the QR attacker with $\lambda = 0.4$ is similar to the performance against the STC attacker. To confirm this similarity, four additional simulations were performed for both attacker types. The results of the additional simulations, shown in Table II, indicate that the performance is comparable. We therefore decided to use a QR attacker with $\lambda = 0.4$ as the basis for the remaining experiments for to reasons. If any changes in the total average regret occur for those experiments, the difference in attacker type has less influence on the result. Furthermore, the resulting average regret value of 0.52 allows observing changes in the average regret value in both directions when the other parameters of the model are evaluated.

Table II
AVERAGE R_T FOR SIMULATIONS WITH $N = 100$, $m = 1$, $T = 500$ FOR QR ATTACKER WITH $\lambda = 0.4$ AND STC ATTACKER AGAINST OL EXPERT.

Attacker	1	2	3	4	5	Average R_T
QR, $\lambda = 0.4$	0,52	0,51	0,54	0,50	0,54	0,52
STC	0,50	0,54	0,53	0,51	0,50	0,52

The expert selection threshold θ is determined by inspecting preliminary results from testing the OL model alone against QR modelled attackers. In a simulation setup with $m = 1$ and $n = 20$, the number of waypoints for a flight path is on average 16, meaning $\gamma = 0.4$ at $t = 100$. The 100 step mark was chosen because the value to which the average regret converges was reached at step 100 already.

We want to examine the effect of changing the remaining five parameters through five experiments. **Experiment 1** establishes the results for the baseline settings that serve as a reference for the remaining experiments. The differences in resulting defender behaviour against the two different attacker types are of interest as well, so the baseline is established for the QR and the STC attacker. **Experiment 2** evaluates the effect of range variations by setting $R = 15$ and $R = 35$. The range is an important feature when choosing a suitable drone in a real-world situation. **Experiment 3** evaluates the difference in performance of having a perfect HE (i.e. with $\varepsilon = 0$). It aims to uncover the performance of the OL against an expert with more precise knowledge of the attacker behaviour. **Experiment 4** evaluates the impact of having more attackers on the domain by simulating the game with 3 and 5 attackers. **Experiment 5** evaluates the effect of discretizing the terrain with a grid that is twice as fine, i.e. 20x20 cells versus 10x10 cells. Given the relationship between the defender's FoV and the grid size, it is important to analyse the effect of a different grid discretization.

Based on the preliminary experiments that have been performed for establishing the base model, we can visually determine that a simulation duration of 500 steps is sufficient for the average regret value to stabilise.

E. Results

In this subsection, the results for every simulation setup discussed above are presented. All simulation settings and the resulting values of the KPI's at $T = 500$ are summarised in

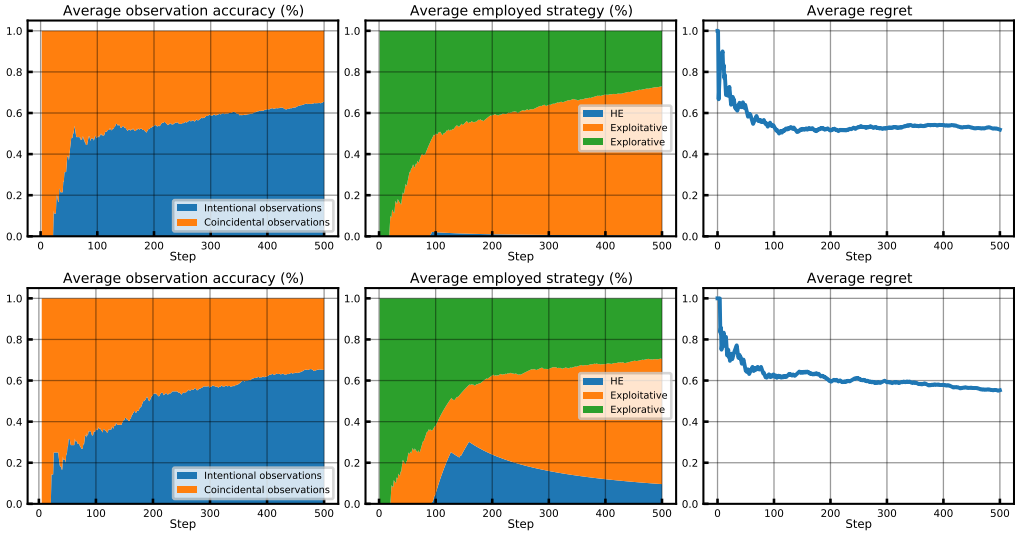

Figure 5. Simulation results from the baseline model with QR adversary (top) and STC adversary (bottom).

Table III
SUMMARY OF THE SIMULATION RESULTS WITH $T = 500$.

Att. type	R	ε	m	N	***Final av.*** R_T	***Final av.*** R_T/m	***Final accuracy***	***Final strategy***	***Runtime (min.)***
	25	**0,3**	**1**	**100**	**0,52**	**0,52**	0,65	OL	189
	15	0,3	1	100	0,63	0,63	0,75	OL	67
	35	0,3	1	100	0,44	0,44	0,48	OL	110
QR	25	**0**	1	100	0,46	0,46	0,71	HE	148
	25	0,3	**3**	100	1,77	0,59	0,58	OL	192
	25	0,3	**5**	100	3,17	0,63	0,54	OL	193
	25	0,3	1	**400**	0,76	0,76	0,52	OL	561
	25	**0,3**	**1**	**100**	**0,55**	**0,55**	**0,65**	OL	120
	15	0,3	1	100	0,72	0,72	0,74	OL	79
	35	0,3	1	100	0,43	0,43	0,63	OL	95
STC	25	**0**	1	100	0,54	0,54	0,69	OL	193
	25	0,3	**3**	100	1,58	0,53	0,71	OL	187
	25	0,3	**5**	100	2,63	0,53	0,69	OL	247
	25	0,3	1	**400**	0,80	0,80	0,23	HE	544

Table III. For every experiment, the values that are discussed are also highlighted in a figure. All other plots can be found in the report accompanying this paper [49].
We emphasise that, even though all these figures are the convergence plots for one randomly generated game instance, the general convergence trends of the KPI's is almost the same across the simulated instances for every setup. However, the initial rounds in the figures may vary among different instances.

Experiment 1 - baseline establishment: The baseline model shows a similar trend in observation accuracy after $t = 200$ for the QR and STC attacker in Figure 5. Before $t = 200$ however, the observation accuracy against the STC attacker was significantly worse. This also translates to a higher average regret in the first stage of the game. As a result, the convergence of the average regret is slower against the STC attacker. For this particular simulation, it can be observed that the HE was briefly superior in the average distribution of defender strategies, but after 59 steps (i.e. 59 steps after step 100) the OL expert was trained enough to outperform it.

Experiment 2 - range variation: Comparing the results in Figure 6 with the results for the QR attacker in Figure 5, we can deduce the following. When the drone range is reduced from 25 to 15 km, we observe a slower convergence of the regret value. Furthermore, from Table III it is clear that the final regret value is higher when the defender has a lower range, meaning decreasing the range results in a decrease in performance. However, we can also observe that the accuracy of the attack observations increases, and from the average coverage per target in Figure 7, we can deduce that the primary reason behind this is the fact that the defender primarily surveilled the three hot spots. The proportion of coincidentally observed attacks is therefore also less, as most of the attacks take place at the hot spots. Secondly, the increase in accuracy could also result from the faster decrease of the value of γ. With lower γ values, the defender performs less explorative strategies in total.
Note that with a range of 15 km, the drone could not reach all cells starting from the base at cell $(4, 6)$, and could barely cover the hot spots. If the hot spots would have been out of its reach, the results could have been worse.

Not surprisingly, when the drone range is increased from 25 to 35 km, we observe a better performance. The average final regret decreases from 0.52 to 0.44. Interestingly, similar to the decrease in range, we can observe an inverse effect on the observation accuracy. Increasing the range results in a decrease in observation accuracy. The reasoning for this behaviour is similar. With a larger range, the drone now covers proportionally more cells, which results in more

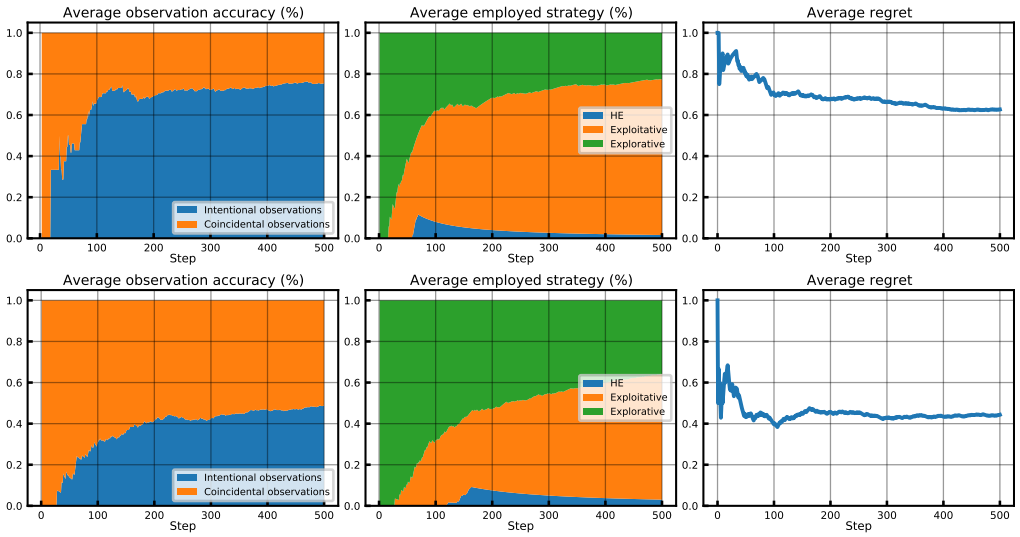

Figure 6. Comparison of the observation accuracy, the employed strategies and the regret for a defender with range $R = 15$ (top) and $R = 35$ (bottom) against a QR adversary.

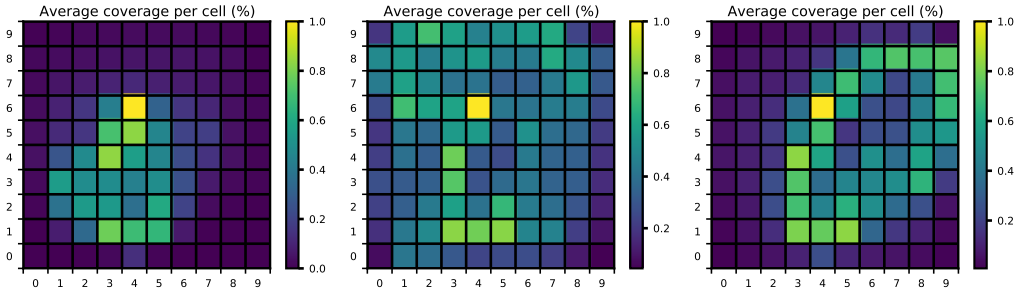

Figure 7. Comparison of the average coverage per cell for a defender with a range of 15, 35, and 25 km LTR.

coincidentally observed attacks. Of course, the value of gamma that is affected by the higher value of k results in more explorative routes, further increasing the coincidentally observed attacks.

Experiment 3 - Human Expert ε variation: Looking at the average employed strategies in this experiment we distinguish the following. As shown in Figure 8, the expert selection algorithm chose the perfect HE with $\varepsilon = 0$ against the QR attacker. This resulted in a lower final average regret than for the baseline model where the HE's error margin is higher ($R_t = 0.46$ versus $R_t = 0.52$ respectfully).

Unexpectedly, the expert selection algorithm chose the OL over the HE when facing an STC attacker. One would think that because of the similar probability distributions q_i and p_{he} of the STC attacker and the HE respectively, the HE would be better suited against the STC attacker. By chance, the HE might have performed poorly in the rounds after the expert selection algorithm was activated and never got a chance to redeem itself, or it is possible that the OL performed exceptionally well. The overall performance is comparable to the base model situation, which makes the latter situation less probable. This indicates that either the proposed expert

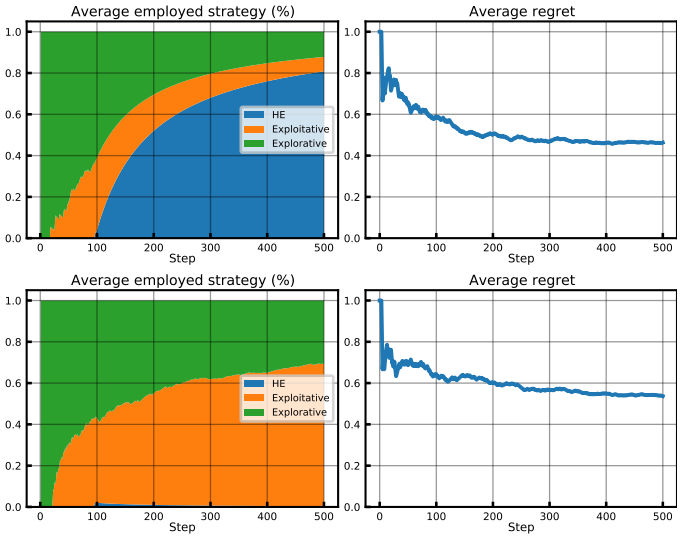

Figure 8. Simulation results from the base model with the HE's $\varepsilon = 0$ against a QR attacker (top) and an STC attacker (bottom).

selection might not function as good as intended, or that the perfect HE's performance against the STC attacker is not as good as expected.

Experiment 4 - Multiple attackers: The results from simulating the game with $m = 3$ and $m = 5$ in Figure 9 yield the following. When adding more attackers to the model, the γ variable reaches the threshold value $\theta = 0.4$ faster than the case with only a single attacker. Therefore we can see that the expert selection algorithm is activated earlier as well. We can also observe that the HE's estimations are better in the early stages of the game. The same holds for the strategy selection against the STC attacker (not represented in a figure), but in that case, the OL takes over earlier than against the QR attacker.

Even though the proportion of explorative strategies is also reduced due to the increased presence of attackers, the overall increase in attacks and observations results nonetheless in a fast learning process for the OL. This manifests itself in the definitive switch from HE to OL, and the drop in average regret that can be observed after that switch.

Note that the average regret is higher due to the higher number of attackers, but normalised by the number of attackers m we can observe that the performance compared with the base models is worse in the case of QR attackers, and better in the case of STC attackers. The reason for this might be that the QR attackers are adaptive and are modelled in as such that the individual attacker has the knowledge of the collective of attackers. This means that, just as the defender, the attackers in this model learn quicker when there are more attackers. This could also explain the difference in observation accuracy between the simulations against the QR attackers and the STC attackers.

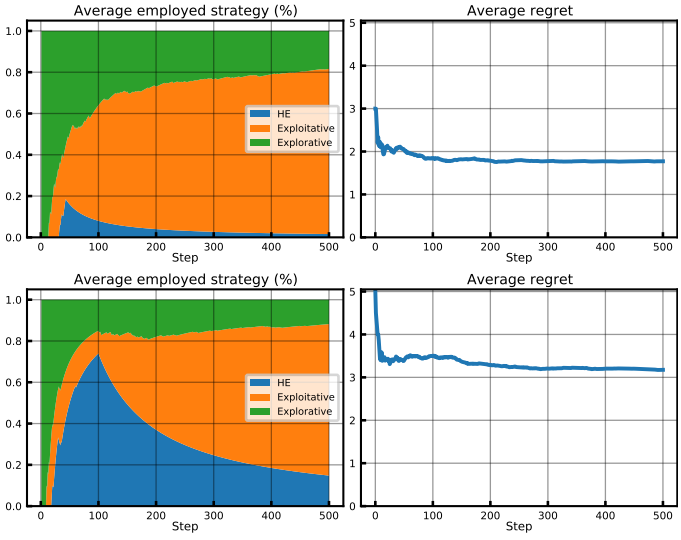

Figure 9. Comparison of the employed strategies and regret for simulations with 3 QR attackers (top) and 5 QR attackers (bottom).

Experiment 5 - Finer grid discretization: The finer discretization of the surveillance area results in the following. First of all, as can be observed in Figure 10, the spatial distribution of the attacks on a 20x20 grid is similar to the spatial distribution on a 10x10 grid. This reveals that the

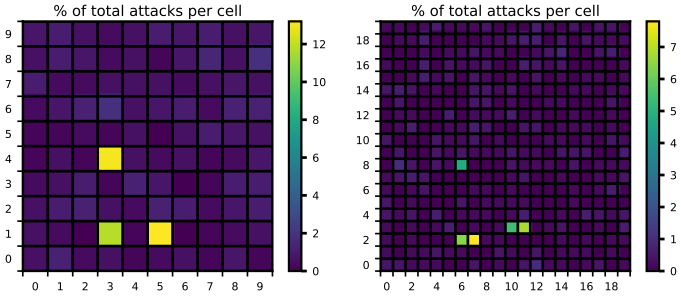

Figure 10. Comparison of attack distribution of a QR attacker with $\lambda = 0.4$ on a 10x10 grid (left) and a 20x20 grid (right).

model's calculation of q_i based on ground truth p_{GT} is not affected by a finer discretization on the same area. In turn, this means that the normalisation of the attackability score $\mathbf{a}$ to p_{GT} is scalable. Secondly, the overall performance of MEOMAPP is worse, with $R_t = 0.76$ and $R_t = 0.80$ against the QR and STC attacker respectively. This is not unexpected as the overall area that is covered by the defender every round is smaller due to the halving of the drone's FoV. Furthermore, comparing the two simulations on a 20x20 grid with each other in Figure 11, it is important to note that the initial performance of both simulations is very different and can be a reason for the difference in performance during the rest of the game. Playing against the STC adversary, MEOMAPP did not find the attacker until round 37. This significantly delayed the start of the learning process and possibly results in the choice to use the HE strategy for the rest of the game. Against the STC attacker, the observation accuracy is relatively low and the average coverage per target is more evenly distributed compared with the other coverage distributions obtained so far. Against the QR attacker, the coverage distribution of the defender is visually traceable to the distribution of estimated rewards per cell. That distribution is not directly relatable to the distribution of attacks per cell, which is more straightforward for the simulation results against the STC attacker. The observation accuracy playing against the QR attacker is significantly higher compared to the STC attacker. The increase in observation accuracy as of step 327 seems to result from the switch to the OL strategies.

VI. Discussion

In this section, we discuss the main findings of this research and the implications of assumptions on its results.

A. Reflection on overall performance

In general, the findings resulting from this study show that MEOMAPP exhibits adaptive behaviour when faced with different attacker types and different game settings. Performance-wise, MEOMAPP's regret converges within 500 steps for all presented game settings. Also, when presented with two experts, each proposing different defender strategies, MEOMAPP can choose the expert that performs best on average at any given time. This is consistent with the work from Xu et al. [21] and Gholami et al. [23], even though we chose to use a different but more realistic and reproducible

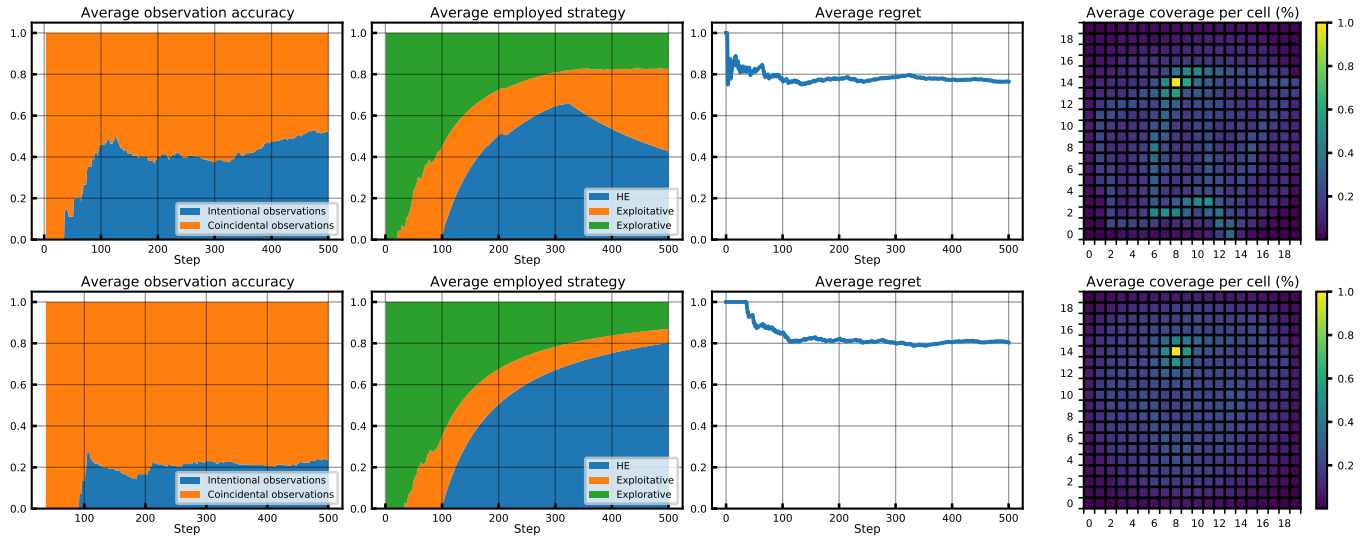

Figure 11. Simulation results from the baseline model on a 20x20 cell grid against a QR attacker (top) and an STC attacker (bottom).

experimental setup. Unfortunately, because of the different setup, it is not possible to directly compare the results, and thus we cannot state with certainty whether MEOMAPP performs better or worse than the stand-alone FPL-UE model [21] or the MINION model [23]. However, the similar trends compared with previous results show that the FPL-UE algorithm can be combined with a route planning algorithm more suitable for UAVs. Therefore, it is expected that MEOMAPP is a more comprehensive solution for aerial surveillance. Additionally, the simulation setup and results from this research can be used as a benchmark for future studies.

B. Reflection on limitations

It is important to put the findings in the perspective of specific limitations resulting from assumptions made for the model and from the specific simulations settings. Even though MEOMAPP is designed for and evaluated by a real-world situation, the simulated attacker behaviour and human expert defender strategies only approximate what a realistic setting would be. For a definitive verdict on MEOMAPP's performance and applicability in real-world wildlife surveillance settings, a field test validating the agent-based model and its simulations is highly recommended. It is important to keep in mind that for a real-world test, the notion of regret cannot be used to evaluate the model. The reason for this is that the regret is calculated using complete information of defender utility, which is not always available. For example: if no attacks are observed by the drones, it is not always possible to know if there was an attack and the drone missed it, or if there was no attack at all.

In this subsection, we elaborate on some important assumptions made for this agent-based model simulation and the limitations they represent.

The definition of the exploration/exploitation variable γ in FPL-UE assumes any adversary behaviour, but only as long as it is constant. The effect of changing adversary behaviour, or the introduction of more or new attackers, in a later stage of the game has not been investigated. In reality, however, this is not an unimaginable scenario.

The model assumes that attackers remain stationary at the attacked target for the whole duration of the defender's surveillance flight, or at least until they are observed. This assumed temporal relation between attacker and defender might heavily influence the real-world performance of MEOMAPP. Also, realistically attackers are present at other locations in the environment before and after attacking a certain target. This is also not reflected in the model. Furthermore, attackers are modelled as independent agents, meaning that attacker cooperation for an attack is not taken into account. Note that QR-based attackers are modelled as having collective knowledge after they were observed, but not as cooperative attackers before an attack.

The observational capabilities of the defender are assumed to be perfect within its modelled observational range. This means that the practical consequences of observing while flying are not taken into account, like bank angles when turning, speed variations, altitude variations, and influences from the weather.

The area that is surveilled is modelled as a two-dimensional, static environment. Changes to the environment and therefore possible variations in attacker and defender payoffs are not taken into account in this model.

It is important to note that even though MEOMAPP was evaluated using a specific real-world scenario, the online learner model does not make any specific assumptions about the park wherein it was simulated. MEOMAPP can be used in any wildlife park whatsoever.

VII. Conclusion

This research investigated if it is possible to apply an FPL-UE algorithm in a multi-expert learning model with a planning method suitable for drones to determine wildlife

surveillance strategies. We proposed MEOMAPP, a Multi-Expert Online Model for Aerial Patrol Planning that compares the performances of a defender strategy by the online learner FPL-UE and a defender strategy by a human expert, and uses the defender strategy to plan a flight path for a surveillance drone. We evaluated MEOMAPP using the agent-based modelling and simulation paradigm, using the real-world case of Aloegrove Safari Park in Namibia as an experimental setup for simulations. We demonstrated that MEOMAPP achieves convergence against two typical attacker models in a variety of simulation settings concerning the environment, the attackers, and the defender. The main contributions of this paper are:

1) The integration of a path planning algorithm for aerial vehicles and a game-theoretic defender strategy algorithm
2) An updated expert selection algorithm that allows the OL to mature before being evaluated
3) The evaluation of the ensemble of algorithms in a reproducible realistic simulation

Despite the agent-based model being a simplification of a real system, MEOMAPP is deemed a suitable algorithm for determining aerial surveillance strategies for wildlife surveillance.

VIII. Recommendations for Future Research

The proposed model and its simulation setup can serve as a foundation for future research in the field of Green Security Games and path planning for wildlife surveillance. The initial assumptions about the wildlife surveillance system made constrained components of the model resulting from this research. These components can be investigated in future research:

A. Attacker model

The following characteristics of the attacker are interesting for future research.

Even though a group of QR attackers could benefit from the collective knowledge, further research into **attacker coordination** could be done.

During the round, attackers are considered stationary. Including **attacker routes** would be a more realistic representation of the system.

If attacker data would be available, it is possible to evaluate MEOMAPP to a more **realistic attacker**. This data can be used to include a more realistic game-theoretic model of attacker behaviour, like SUQR [14], CAPTURE [18] or SHARP [17].

This does not mean that MEOMAPP will require prior knowledge, only that it could be evaluated against realistic players that are modelled using real attacker data.

More information about attackers can also be used for a **different temporal model** than the defender.

B. Defender model

Currently, the defender is a single drone. However, in reality, wildlife surveillance is not done by a drone alone. **Coordination with rangers** and other defensive agents is an interesting research field to elaborate on in the future.

C. Path planning

The path planning algorithm now takes into account the most simple drone model for its constraints. Including constraints related to **flight dynamics** can give insights in the actual flight path.

Even though the online learner's GR algorithm can learn from the coincidental observations, a path planning algorithm that includes estimated rewards of the arcs, as well as rewards of the nodes, could produce more optimised flight paths.

If more defenders enter the game, Multi-Agent Path Finding algorithms could be studied in the context of GSGs.

References

[1] D. Mouillot, D. R. Bellwood, C. Baraloto, J. Chave, R. Galzin, M. Harmelin-Vivien, M. Kulbicki, S. Lavergne, S. Lavorel, N. Mouquet, C. E. T. Paine, J. Renaud, and W. Thuiller, "Rare species support vulnerable functions in high-diversity ecosystems," *PLOS Biology*, vol. 11, no. 5, pp. 1–11, May 2013.

[2] L. O. Smith and C. Gerstetter, "The Costs of Illegal Wildlife Trade: Elephant and Rhino. A study in the framework of the EFFACE research project," Ecologic Institute, Berlin, Tech. Rep. 1, 2015. [Online]. Available: www.efface.eu

[3] Unknown. (2020, July) Recognizing and supporting rangers working against all odds. UNESCO. [Online]. Available: https://whc.unesco.org/en/news/2139

[4] E. Bondi, A. Kapoor, D. Dey, J. Piavis, S. Shah, R. Hannaford, A. Iyer, L. Joppa, and M. Tambe, "Near real-time detection of poachers from drones in airsim," in *Proceedings of the Twenty-Seventh International Joint Conference on Artificial Intelligence, IJCAI-18*. International Joint Conferences on Artificial Intelligence Organization, July 2018, pp. 5814–5816.

[5] M. S. Norouzzadeh, A. Nguyen, M. Kosmala, A. Swanson, M. S. Palmer, C. Packer, and J. Clune, "Automatically identifying, counting, and describing wild animals in camera-trap images with deep learning," *Proceedings of the National Academy of Sciences*, vol. 115, no. 25, pp. E5716–E5725, 2018.

[6] J. Jiménez López and M. Mulero-Pázmány, "Drones for conservation in protected areas: Present and future," *Drones*, vol. 3, no. 1, 2019.

[7] B. Ivoševi, Y.-g. Han, Y. Cho, and O. Kwon, "The use of conservation drones in ecology and wildlife research," *Journal of Ecology and Environment*, no. February, 2015.

[8] F. Fang, P. Stone, and M. Tambe, "When security games go green: Designing defender strategies to prevent poaching and illegal fishing," in *IJCAI International Joint Conference on Artificial Intelligence*, vol. January, 2015, pp. 2589–2595.

[9] R. Longadge and S. Dongre, "Class imbalance problem in data mining review," *CoRR*, vol. abs/1305.1707, 2013.

[10] T. D. Pigott, "A review of methods for missing data," *Educational Research and Evaluation*, vol. 7, no. 4, pp. 353–383, 2001.

[11] R. Yang, C. Kiekintveld, F. Ordonez, M. Tambe, and R. John, "Improving resource allocation strategy against human adversaries in security games," in *Proceedings of the Twenty-Second International Joint Conference on Artificial Intelligence - Volume Volume One*, ser. IJCAI'11. AAAI Press, 2011, p. 458–464.

[12] J. Pita, R. John, R. Maheswaran, M. Tambe, R. Yang, and S. Kraus, "A robust approach to addressing human adversaries in security games," vol. 242, June 2012, pp. 1297–1298.

[13] M. R. D. and T. Palfrey, "Quantal response equilibria for normal form games," *Games and Economic Behavior*, vol. 10, no. 1, pp. 6–38, 1995.

[14] T. H. Nguyen, R. Yang, A. Azaria, S. Kraus, and M. Tambe, "Analyzing the effectiveness of adversary modeling in security games," in *Proceedings of the Twenty-Seventh AAAI Conference on Artificial Intelligence*, ser. AAAI'13. AAAI Press, 2013, p. 718–724.

[15] R. Yang, B. Ford, M. Tambe, and A. Lemieux, "Adaptive resource allocation for wildlife protection against illegal poachers," in *Proceedings of the 13th International Conference on Autonomous Agents and Multiagent Systems (AAMAS)*, 2014, pp. 453–460.

[16] F. Fang, T. H. Nguyen, R. Pickles, W. Y. Lam, G. R. Clements, B. An, A. Singh, B. C. Schwedock, M. Tambe, and A. Lemieux, "Paws — a deployed game-theoretic application to combat poaching," *AI Magazine*, 2017.

[17] D. Kar, F. Fang, F. M. D. Fave, N. Sintov, and M. Tambe, "A Game of Thrones: When Human Behavior Models Compete in Repeated Stackelberg Security Games," in *Proceedings of the 14th International Conference on Autonomous Agents and Multiagent Systems*, 2015, pp. 1381–1390.

[18] T. H. Nguyen, A. Sinha, S. Gholami, A. J. Plumptre, L. N. Joppa, M. Tambe, M. Driciru, F. Wanyama, A. Rwetsiba, R. Critchlow, and C. M. Beale, "CAPTURE: A New Predictive Anti-Poaching Tool for Wildlife Protection," *Proceedings of the 15th International Conference on Autonomous Agents and Multiagent Systems (AAMAS)*, pp. 767–775, 2016.

[19] C. Kiekintveld, J. Marecki, and M. Tambe, "Approximation methods for infinite bayesian stackelberg games: Modeling distributional payoff uncertainty," in *Proc. of 10th Int. Conf. on Autonomous Agents and Multiagent Systems (AAMAS)*, Tumer, Yolum, Sonenberg, and Stone, Eds., Taipei, 2011, pp. 2–6.

[20] A. Blum, N. Haghtalab, and A. D. Procaccia, "Learning optimal commitment to overcome insecurity," in *Advances in Neural Information Processing Systems 27*, Z. Ghahramani, M. Welling, C. Cortes, N. D. Lawrence, and K. Q. Weinberger, Eds. Curran Associates, Inc., 2014, pp. 1826–1834.

[21] H. Xu, L. Tran-Thanh, and N. R. Jennings, "Proceedings of the 15th international conference on au- tonomous agents and multiagent systems," 2016.

[22] G. Neu and G. Bartók, "An efficient algorithm for learning with semi-bandit feedback," *CoRR*, vol. abs/1305.2732, 2013.

[23] S. Gholami, A. Yadav, L. Tran-Thanh, B. Dilkina, and M. Tambe, "Don't Put All Your Strategies in One Basket: Playing Green Security Games with Imperfect Prior Knowledge," in *Proc. of the 18th International Conference on Autonomous Agents and Multiagent Systems*, vol. 9, 2019.

[24] C. Camerer and R. S. Foundation, *Behavioral Game Theory: Experiments in Strategic Interaction*, ser. The Roundtable Series in Behavioral Economics. Princeton University Press, 2003.

[25] D. Kar, B. Ford, S. Gholami, F. Fang, A. Plumptre, M. Tambe, M. Driciru, F. Wanyama, A. Rwetsiba, S. California, and L. Angeles, "Cloudy with a chance of poaching : Adversary behavior modeling and forecasting with real-world poaching data," in *Proceedings of the 16th Conference on Autonomous Agents and MultiAgent Systems*, no. May, 2017, pp. 159–167.

[26] S. Gurumurthy, L. Yu, C. Zhang, Y. Jin, W. Li, H. Zhang, and F. Fang, "Exploiting Data and Human Knowledge for Predicting Wildlife Poaching," in *ACM SIGCAS Conference on Computing and Sustainable Societies 2018*, 2018.

[27] S. Gholami, B. Ford, F. Fang, A. Plumptre, M. Tambe, M. Driciru, F. Wanyama, A. Rwetsiba, M. Nsubaga, and J. Mabonga, "Taking It for a Test Drive: A Hybrid Spatio-Temporal Model for Wildlife Poaching Prediction Evaluated Through a Controlled Field Test," in *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, vol. 10536 LNAI, 2017, pp. 292–304.

[28] S. Gholami, S. Mc Carthy, B. Dilkina, A. Plumptre, M. Tambe, M. Driciru, F. Wanyama, A. Rwetsiba, M. Nsubaga, J. Mabonga, T. Okello, and E. Enyel, "Adversary models account for imperfect crime data: Forecasting and planning against real-world poachers," 2018, pp. 823–831.

[29] N. Park, E. Serra, T. Snitch, and V. S. Subrahmanian, "APE: A Data-Driven, Behavioral Model-Based Anti-Poaching Engine," *IEEE Transactions on Computational Social Systems*, vol. 2, no. 2, pp. 15–37, 2015.

[30] P. Toth and D. Vigo, *The vehicle routing problem*. SIAM, 2002.

[31] A. Gunawan, H. C. Lau, and P. Vansteenwegen, "Orienteering problem: A survey of recent variants, solution approaches and applications," *European Journal of Operational Research*, vol. 255, no. 2, pp. 315 – 332, 2016.

[32] R. Martí, P. M. Pardalos, and M. G. C. Resende, Eds., *Handbook of Heuristics*. Springer, 2018.

[33] M. Speranza and C. Archetti, "A survey on matheuristics for routing problems," *EURO Journal on Computational Optimization*, vol. 2, November 2014.

[34] A. Bazghandi, "Techniques, advantages and problems of agent based modeling for traffic simulation," 2012.
[35] A. McLane, C. Semeniuk, G. Mcdermid, and D. Marceau, "The role of agent-based models in wildlife ecology and management," *Ecological Modelling*, vol. 222, pp. 1544–1556, April 2011.
[36] F. Bousquet, R. Lifran, M. Tidball, S. Thoyer, and M. Antona, "Agent-based modelling, game theory and natural resource management issues," *The Journal of Artificial Societies and Social Simulation*, vol. 4, March 2001.
[37] R. Luce, *Individual choice behavior: a theoretical analysis*. Wiley, 1959.
[38] S. Anderson, J. Goeree, and C. Holt, "The logit equilibrium: A perspective on intuitive behavioral anomalies," *Southern Economic Journal*, vol. 69, pp. 21–47, July 2002.
[39] D. O. Stahl and P. W. Wilson, "Experimental evidence on players' models of other players," *Journal of Economic Behavior & Organization*, vol. 25, no. 3, pp. 309 – 327, 1994.
[40] B. Gao and L. Pavel, "On the properties of the softmax function with application in game theory and reinforcement learning," 2017.
[41] D. McFadden, "Conditional logit analysis of qualitative choice behaviour," in *Frontiers in Econometrics*, P. Zarembka, Ed. New York, NY, USA: Academic Press New York, 1973, pp. 105–142.
[42] F. Klügl and A. L. C. Bazzan, "Agent-based modeling and simulation," *AI Magazine*, vol. 33, no. 3, p. 29, September 2012. [Online]. Available: https://www.aaai.org/ojs/index.php/aimagazine/article/view/2425
[43] L. Morell, "Unit testing and analysis," April 1989.
[44] D. Helbing and S. Balietti, "How to do agent-based simulations in the future: From modeling social mechanisms to emergent phenomena and interactive systems design," *Technical Report 11-06-024*, July 2015.
[45] D. Masad and J. Kazil, "Mesa: An agent-based modeling framework," January 2015, pp. 51–58.
[46] G. van Rossum, B. Warsaw, and N. Coghlan, "Style guide for Python code," PEP 8, 2001. [Online]. Available: https://www.python.org/dev/peps/pep-0008/
[47] S. Mitchell, S. M. Consulting, and I. Dunning, "Pulp: A linear programming toolkit for python," 2011.
[48] L. Gurobi Optimization, "Gurobi optimizer reference manual," 2020. [Online]. Available: http://www.gurobi.com
[49] K. Dhoore, "Online agent-based aerial patrol planning for wildlife surveillance," Master's thesis, Delft University of Technology, Delft, October 2020.

II

Literature Review

The literature review has already been graded

Nomenclature

ABM	Agent-Based Model
ABMS	Agent-Based Modelling and Simulation
ABS	Agent-Based Simulation
CPUE	Catch Per Unit Effort
DD	Data Duplication
DT	Decision Tree
DTB	DT bagging ensemble
EM	Expectation Maximisation
FPL-UE	Follow-the-Perturbed-Leader with Uniform Exploration
GPR	Gaussian Process Regression
GSG	Green Security Game
MAS	Multi-Agent System
MILP	Mixed Integer Linear Program
MLE	Maximum Likelihood Estimation
MRF	Markov Random Field
NN	Nearest Neighbour
NPP	Net Primary Productivity
NS	Negative Sampling
PS	Positive Sampling
pSTG	probabilistic Spatio-Temporal Graph
PWL	PieceWise Linear
QENP	Queen Elizabeth National Park
SSG	Stackelberg Security Game
STG	Spatial Temporal Graph
STO	Spatio-Temporal Optimisation
SUQR	Subjective Utility Quantal Response
SVB	SVM bagging ensemble
SVM	Support Vector Machine
TSP	Travelling Salesman Problem
UAV	Unmanned Aerial Vehicle

1 Introduction

This document contains the literature review and subsequent project plan for an optimisation research project about drone surveillance mission planning in wildlife conservation. The following chapter introduces the domains of drone surveillance and wildlife conservation, as well as the motivation for this particular project.
Furthermore, the problem that will be investigated is broken down in sub-problems, which are the backbone of the literature review.

1.1. Motivation for Research

Applications of drone surveillance

Unmanned Aerial Vehicles, also known as ”drones” or ”UAVs” have become more and more popular in the past few decades. While kids and hobbyists have been having fun with small low range and low endurance devices since a long time, the number of professional applications for drones have seen an ever-increasing inclination the latest years. Especially the unique bird’s-eye view that a UAV has is a major characteristic that makes them very interesting as carriers of visual sensors, mainly cameras and infra-red sensors. This effectively removes the need of a human pair of eyes in the sky, with the added advantages that most drones are smaller than human-carrying aircraft and can fly in conditions that are not preferable for humans.
The use of drones for military surveillance is well known and goes back a long time, but the continuous decrease in costs and increase in availability of reliable (semi-)automated flight control systems has made drones available to the public as well, which are used widely in the movie and photography industry. Also, visual inspection of high-voltage cables, windmills, pipelines and other equipment that is hard to reach is increasingly done by UAVs.
Drone surveillance in the non-military domain is coming up as well. UAVs can cover large surfaces quickly and even autonomously, which reduces the number of staff. This can be applied in the security of large industrial sites or events, surveillance of traffic or maritime zones, and also in wildlife conservation.

Wildlife Conservation

Wildlife on Earth is facing the largest threats of human presence ever. More and more forests are destroyed for logging and farming, the ocean's fish reserves are depleting, ice caps are melting, and animals are being poached to extinction. Furthermore, water levels and temperature changes are impacting habitats in such a way that the growth of food and keeping livestock becomes difficult and leads to cattle and land theft, pollution and diversion of water sources, illegal fishing and grazing, etc. These events, whether caused by human activity or not, need to be observed and studied in order to be understood and managed. Multiple organisations are devoted to this, in any domain mentioned above. The majority of these organisations, however, does not have the resources necessary to fulfil their goals at 100%, but their activities are more important than ever and any improvement in their resources and efficiency will help their causes.

The Eyeplane Case

One of the new developments in wildlife conservation is the use of drones to surveil protected natural areas. These areas are usually several square kilometres and are hard to protect. Drones can be useful to locate animals, spot fires and other natural hazards, or to detect poachers and other security breaches. Next to conservation parks, large farms also suffer the same problems. Farmers in sparsely populated areas in Africa have to deal with cattle theft and killing, as well as the presence of natural predators on their domains.
These problems were also identified by Eyeplane, an organisation that aims to protect farms and wildlife parks by means of drone surveillance[1]. Eyeplane is also a partner for this research. The scale of Eyeplane's wildlife surveillance is unprecedented in the domain and consists of large fixed-wing UAVs that can fly autonomously for multiple hours in a range of hundreds of square kilometres. This new application of such type of surveillance is challenging and requires innovative approaches to efficiently manage the operations in order to effectively address the problems for which it aims to be a solution.
One of the tasks involved in drone surveillance is making a flight plan. For wildlife conservation, this means that the flight plan must be so that the amount of useful observations by the drone (ranging from detecting poachers to counting animals) is maximised within the operational limits, whether they depend on the drone's capabilities or on the available resources. The task can be summarised as to know where to fly at what time, and to do that as efficiently as possible.

1.2. Problem Analysis

The focus domain of this research will be the apprehension of poachers over farmland and wildlife parks in Namibia. Practically, the research will be tailored to the operational situation of Eyeplane in order to be able to conduct tests to verify, validate and evaluate the theory and models that will be developed. Adaptation of the theory and models to other situations in the wildlife conservation domain and beyond will be considered as well. The operational situation for this research can be described as follows.
The plots of land that will be protected measure 80 km^2 on average and are distributed in a region with a radius of approximately 100 km around a single base of operations. Surveillance will be done by a fixed-wing UAV with a sufficient range and endurance to operate in the described area. The drone will fly at an altitude of 100 to 200 m going at 80 km/h and is equipped with a camera and an infra-red sensor that can be positioned in different angles.

[1] For more information about Eyeplane, the reader is invited to visit `www.eyeplane.com`

In order to conduct a comprehensive literature review for the research, the problem is broken down by means of sub-questions that need to be answered. The main question that needs to be answered is:

How to plan the flight path for a UAV on a surveillance mission in order to minimise the number of missed observations of attacks given its operational constraints?

A flight path, which is the route followed by an aircraft through the air, is usually composed of at least an origin, a destination, and a route between both. The route between both is defined by selected waypoints. For conventional air travel, these waypoints are navigational beacons that guide aircraft through selected airspace, but for surveillance, this is not the case. Moreover, the origin and destination are often the same. The first subquestion will thus be

1. How to determine when and where to surveil?

For the surveillance problem, this translates to the question where surveillance should be performed and which points are of interest to keep an eye on. This leads to the following questions:

   (a) How should a waypoint be defined?

   (b) How to evaluate the importance of surveillance of every point in the area?

Furthermore, there are multiple ways to fly between the different waypoints. The order in which they appear in the route will determine how efficient the flight path is.

2. How to determine the optimal flight path between interesting points of surveillance?

This question, in turn, leads to the following subquestions:

   (a) What is the most efficient way to calculate the optimal flight path?

   (b) How is the flight path constrained?

Also, the desire to make use of the agent-based modelling paradigm results in a third subquestion:

3. How to represent this system of actors and their environment that need surveillance as an agent-based model?

The first question is addressed in chapter 2 and the second question is addressed in chapter 3. The choice of paradigm for this optimization research will be further elaborated upon in chapter 4. Finally, the results of this literature review are summarised and form the basis of a research plan presented in chapter 9.

2

Security: Where to Surveil

In this chapter, existing literature where similar security problems have been addressed is analysed. The insight gathered by this analysis assists in defining what is required to find a solution for the security problem.

2.1. Related Work

The security problem has been addressed multiple times in the wildlife domain. Currently, it focuses mainly on delivering assistance in determining a patrol route or patrol strategy for foot patrols, in a rare case assisted by a short-range drone [21], to intercept poachers and find traps/snares. However, the main part of the security problem which is deciding where to patrol is applicable to the problem presented in this study as well.
Below, nine models with different takes on the problem are summarised. For each model, the summary contains a description of the structure, the output and the expected quality of the results. It is important to note that the majority of the models (in chronological order: PAWS, CAPTURE, INTERCEPT, DTB+MRF, iWare-E, MINION) was developed based on the results each of them produced when applied to Uganda's Queen Elizabeth National Park (QENP).
All models approach the problem by dividing the area that needs protection in square grid cells and try to define the expectation of an attack ("attackability") on every cell. Differences in the models can be observed in the following aspects:

- Variable and invariable data that characterises every cell
- Application of data pre-processing
  - If applied, what kind of pre-processing
- Method for estimating attackability
- Application of patrol planning
  - If applied, the presence of constraints
  - If applied, the method for patrol planning

Sometimes some of the aspects can be tackled simultaneously within the model, like the estimation of observability of an attack, which can be applied during the data pre-processing, or during attackability estimations.
An overview of model characteristics is given in Table 2.1 and the data that every model assigns to each cell in order to compute attackability and/or patrol routes is summarised in Table 2.2.
The presented models can be assigned to two groups, namely models that apply game theory for attacker behaviour modelling, and models that integrated the attacker behaviour in the attackability. This grouping has been adhered to below.

Models With Game-Theoretic Attacker Behaviour

The following three models use game-theoretic models to determine attacker behaviour. Game theory as a straightforward method to model thy dynamics of the relation between a defender and an attacker, since it allows to study the decision making of both based on their expected utilities, but the outcome might not be the intention of any of the agents [23].

PAWS

First presented in 2014 by Yang et al. [26], PAWS is the first application of security games and learning adversary behaviour in the field of wildlife conservation. The main problem that PAWS aims to solve is the efficient allocation of resources in patrols in order to find traps and snares set by poachers. In order to do so, PAWS consists of a model based on Subjective Utility Quantal Response Model (SUQR) that captures behavioural uncertainty of a population of poachers by learning from identified (where the poacher is known) and anonymous (where the poacher is unknown) data, after which it applies Stackelberg Security Games (SSG) as a framework to adapt patrol strategies to the poachers' behavioural model, which marked the start of so-called Green Security Games (GSG).
The data used by PAWS consisted of observations of animals and suspected illegal activity reported by park rangers in QENP which spans an area of approx 2500 sq. km. The parameters of the poacher data are estimated by Maximum Likelihood Estimation (MLE) by using identified data to correctly distribute the parameters over the anonymous data. The data is discretized locally in squares of 1km x 1km and the time is defined in ”game rounds”, which depend on when patrols take place and how long they take (up to multiple days).
After testing PAWS in QENP development continued and improvements to PAWS have been presented by Fang et al. [6]. Regarding poacher behaviour improvements were made by adding more complex topographic features and handling uncertainties in species distribution. Regarding planning and practical applications of the model, improvements were made to ensure scalability and to adhere to patrol schedules. The last version of PAWS was tested in Malaysian forests with a more complex terrain with an average patrol visiting 22.67 grid cells in 4.67 days.

CAPTURE

CAPTURE is developed as an improvement of PAWS by Nguyen et al. [19]. Differences with PAWS are the incorporation of imperfect observations by rangers, the temporal dependence of poachers' behaviour on their past activities, and further enriching the set of data features that PAWS used.
CAPTURE's behavioural model exists of two layers of which the first calculates the probability that a target i will be attacked at a certain time t and the second layer computes the probability that the attack is observed if the attack has taken place and the target was patrolled consecutively.
Furthermore, instead of an MLE, it uses an Expectation Maximisation (EM) method to estimate the behavioural model's parameters because an imperfect observation probability is available too. It also introduces two new heuristics to reduce the computational costs of using an EM procedure to less

than one-sixth of the original run time. These heuristics are based on reducing complexity by exploring intrinsic properties of this specific game, namely the possibility to compute the attack probability and the observation probability separately. Additionally, it is assumed that neighbouring cells with similar properties will have similar parameters.
The patrols are planned by a game-theoretic model for single-step and multi-step patrolling. The single-step patrol is solved by piecewise linear approximation of the utility function and the multi-step patrol strategy is solved by non-convex solvers. This complexity is due to the temporal dependence and detection uncertainty that complicate the utility functions. CAPTURE has also been tested on historical QENP data and was found to outperform strategies based on SUQR, Maximin and historical real-world strategies. For CAPTURE, the QENP data was further enriched and categorised in types of poaching activity (commercial and non-commercial, animal, fishing, plants, etc.), grouped in one of four seasons, and included net primary productivity (NPP).

MINION
Gholami et al. [10] presents the latest model for patrol route planning based on GSG's. It stands out between the other models presented in this literature review because of its ability to construct an attacker behaviour model without the need of prior (error-prone) data ánd generating a constraint-aware patrol schedule simultaneously.
MINION consists of two patrol planning algorithms that define a strategy. One is an online[1] learning algorithm called MINION-sm ("Sub Module") and the other one is an ML-based algorithm presumably similar to iWare-E by Gholami et al. [9]. MINION-sm is based on the FPL-UE algorithm (follow-the-perturbed-leader with uniform exploration) [25] which frames the GSG as an adversarial combinatorial bandit problem with a trade-off at every round between exploration and exploitation of the domain based on expected payoffs.
MINION itself evaluates the expected payoffs of both algorithms (although perturbed by artificial noise) every round and chooses the best performing one.
To evaluate MINION it was compared with stand-alone versions of its two submodels. All models have been tested with both a stochastic (stationary) and a QR (non-stationary) adversary behaviour model for different mean absolute errors and rationality parameters respectively. For all the stochastic models, the MINION algorithm's results were superior. For the QR models, the MINION-sm algorithm outperforms the others. The ML-based performs worst of the three since it relies only on prior knowledge and does not take into account behavioural change of the attacker during the game.
The training and test data for MINION and its sub-models are from QENP, but the algorithm has not yet been tested in the field.

Models With Integrated Attacker Behaviour

The models below directly calculate the attackability of cells without representing attackers separately. The estimations are based on either regression or classification methods.

APE
Park et al. [21] introduce an algorithm that aims to find a coordinated route for rangers and drones (short-range quadcopter drones) to protect a maximal number of animals in a game park at any given point in time. In order to do so, they predict future locations of the animals as well as locations that poachers would typically target. These predictions are used as input for the algorithm which produces routes for the rangers with additional instruction on where to fly their drones. The routes are updated

[1] Online meaning that it has to make decisions and allocate resources with incomplete knowledge of the future, see Karp [15] for more information.

daily with hourly data, meaning that a prediction is made today for where rhinos will be on each hour tomorrow.
APE was developed as two heuristic algorithms, one of which is local search-based and one that builds on that by leveraging results using a genetic algorithm. The algorithms use Spatio-temporal graph (STG) methods to define routes for the rangers and their drones and probabilistic Spatio-temporal graphs (pSTG) to predict animal movements. Finding the optimal route is a process of spatio-temporal optimisation (STO) The poachers are not modelled as agents and their behaviour is assumed to be included in an attackability prediction model that assigns a probability of attack to a certain place on a certain time of day based on Gaussian process regression (GPR).
The models are based on large and relatively precise sets of data about animal locations (GPS-tracked rhino's) and poacher attacks. Expert knowledge was also heavily introduced in the probability predictions to estimate time-of-day of attacks.

INTERCEPT
In a different take to PAWS and CAPTURE, INTERCEPT was developed by Kar et al. [14] with a focus on future attacks instead of finding past attacks. Furthermore, it improves on CAPTURE's shortcomings by taking a different modelling approach: instead of a complex game-theoretic behavioural model that considers temporal relationships, it applies a democratic ensemble of five different decision trees (DT) to model adversary behaviour. Advantages of this approach are the increased interpretability of the model and the reduction of computational effort. Furthermore, even though DT's do not usually capture spatial correlations, a spatially-aware decision tree algorithm was developed in order to identify hot spots and use them as a variable for future predictions.
Apart from the invariable terrain features variable data as patrol effort and observed illegal human activity are collected as well for every grid cell. A cell is deemed attackable if the cell has ever been attacked before, meaning that the time span of the training set is a factor that influences the result directly, and that – contrary to CAPTURE – INTERCEPT lacks a fine temporal element in the behaviour prediction.
INTERCEPT was evaluated by comparing prediction results of CAPTURE (and some CAPTURE variants presented in the paper) and several machine learning methods with the data set of QENP used previously for PAWS and CAPTURE. INTERCEPT performs significantly better for attackability prediction while simultaneously being a faster and more interpretable method. The model has also been evaluated during a period of one month in a real-world deployment over two areas of approximately 9 sq. km where the reported observations were higher than in historical data. It is important to note that INTERCEPT does not provide detailed patrol planning, but only an attack prediction module. Furthermore, it only determines if a target will be attacked or not, it does not provide a probability of being attacked other then 0% or 100%.

Multiple Decision Tree Model
Gurumurthy et al. [11] presents another model based on decision trees to predict the attackability of grid cells. The model presented in the paper relies on extensive topographic data, historic patrol data, and positive (non-zero poaching activity), negative and unknown poaching activity data. The imbalance in the data is addressed by artificial data augmentation and by eliciting information from domain experts, which give threat level scores to clusters of cells (determined by k-means clustering) in the domain. These aggregated scores are then used as another data point per cell.
The model consists of 1000 DT's which each train on a different version of 10% of the total training data. Their aggregated scores produce an attack probability for every cell. For evaluation different data augmentation techniques have been used for differentiation between models, of which the version that applied data duplication (DD), negative sampling (NS), and positive sampling (PS) proved to be

the best performing one. Data duplication is the practice of duplicating positive examples in order to balance the data set. For negative sampling, a portion of the unlabelled data is classified as negative since most of the unlabelled data will be negative, and positive sampling adds unlabelled points with a possible positive label to the positive data set based on the experts' scores. Applying DD proved crucial for good performance, and NS and PS only had a positive influence when added together.
An early version of the model augmented by DD and NS only was applied for patrol planning in the Huang Ni He National Nature Reserve (75 sq. km) and was proven successful.

Neural Network
In the same paper, Gurumurthy et al. [11] also proposed a model bagging 100 neural networks for attack probability estimation. The networks had three layers, the first one having eight neurons, the second one having four neurons and a single neuron for the final layer giving the probability. The same data and data augmentation techniques were used to train and test the neural network as for the Multiple Decision Tree model. The application of all data augmentation techniques (DD, NS and PS) also yield the best results for the neural network method, although the performance is very poor compared to the DT method.

Decision Tree Bagging + MRF Augmentation
Another example of the use of decision trees to determine poaching threats is presented by Gholami et al. [8]. A novelty for this model, however, is a second layer of data processing with a spatio-temporal model based on Markov Random Fields (MRF) which improves the prediction results relative to INTERCEPT and other well-known machine learning techniques. To estimate the MRF model's parameters Estimation Maximisation (EM) is used (similar to CAPTURE).
For experimentation with the MRF, the data was divided into time steps of one year each, and training sets consisted of three consecutive years of data. Furthermore, different models were compared with each other. A difference was made between a *global* model with one set of local parameters, and a *geo-clustered* model with multiple sets of local parameters throughout the area. Both sets have been applied to an MRF with spatial effects that took into account the effect of neighbouring cells and an MRF without spatial effects. the best performing version was the geo-clustered data set with spatial effects.
It is important to keep in mind that the MRF is only an augmentation of the prediction for targets that are continually monitored, meaning that the targets have a data point for every time step in the time span of the data set. Consequently, for a good performing model of this type, a large amount of data is required, and the computational costs are still significant (see Table 2.1). The time span of the data sets is multiple years, and one time step equals one year.
Next to being developed with the QENP dataset, the DT bagging + MRF hybrid model has been tested extensively in real-life applications. The model was used in QENP over an area of 243 sq. km during eight months, and proved to be successful with a catch per unit effort (CPUE) (observations of illegal activities per 1 km patrolled) of 0.12 compared to the historical 0.04 in QENP.

iWare-E
Gholami et al. [9] present another model to address shortcomings from the model presented in Gholami et al. [8]. Most notably, the model adheres to the temporal discretisation of three months instead of one year. Also the run time for running this model with a finer discretisation almost eight times shorter compared to the hybrid model (also run with a finer discretisation).
The iWare-E model also addresses the issue of imperfect observation and does that by training a set of weak learners with different patrol effort thresholds. Since observations where deemed more reliable the higher the patrol effort was, different patrol effort thresholds for weak learners result in different

predictions. A votes power matrix is constructed based on the weak learner's qualifications and is applied to determine a weighted average of the predictions to construct a final prediction of attackability for every grid cell.
The data set is compiled in a combined matrix form including a predictive feature for every cell at every time step and the corresponding variable and invariable characteristics of each cell.
Furthermore, iWare-E also presents a planning algorithm that first approximates the (black box) ML through the use of piecewise linear (PWL) functions, to subsequently solve a mixed-integer linear program (MILP) that maximises the probability of detecting attacks. Solving it as a GSG was considered too, but for the continuous predictions that iWare-E provides using a GSG would either result in a loss in solution quality for a coarse discretisation or very large run times for a fine discretisation.
iWare-E has been tested with a DT-bagging ensemble (noted as DTB and as a Support Vector Machine (SVM) bagging ensemble (noted as SVB) as weak learners. Both versions of the model were compared against pure SVB, DTB and MRF models, and against the DTB-MRF hybrid model from the previous section. For the aforementioned QENP data set, both iWare-E versions produced better or similar results than the other models, but the version with DTB as weak learners was 7.5x faster than the SVB version.

Overview and Discussion of Model Characteristics

This subsection contains two tables that summarise information about the models presented above. Table 2.1 gives an overview of different model characteristics, and Table 2.2 gives an overview of input data associated with each cell. With these tables and the summaries presented in the previous sections, it is possible to differentiate models based on the different problem characteristics and expected results.

The first question that can be asked is whether enough data is available to calculate attack probabilities. All but one model heavily rely on an initial data set that enables them to estimate correct parameters for the predictions, and even the model that offers the ability not to rely on data is enhanced by the use of it. It can be stated that if no data is available, an online game-theoretic model inspired by MINION-sm would be an optimal choice to determine where to allocate the resources to protect them. Gathering data during deployment is vital to improve future performance, and integration with an ML or probabilistic model to generate predictions would enhance the performance of the overall model.

If data is available, the attackability results by CAPTURE versus the results of INTERCEPT, the DTB + MRF Hybrid model, and iWare-E prove that using relatively simple combinations of ML methods are a faster and more reliable way than game-theoretic methods to model which cells are possible targets and which are not. Depending on the available computational time more or fewer relations can be integrated into the data. It is important to note that there are a lot of different classification and regression techniques and that these consist of multiple parameters and variables that can be modified, meaning that the results can vary heavily depending on the setup (for example, notice the difference in the application of DTB's by Gurumurthy et al. [11] and Gholami et al. [8]). All of the papers which applied an ML technique, however, resulted in using DTB as the preferred method.
Furthermore, the effects of data pre-processing on the final results cannot be underestimated, as shown by Gurumurthy et al. [11].
Next, it is important to evaluate what kind of data to use. Similarities in the state-of-the-art are the registration of distances to settlements, patrol posts, and sources of water, the slope of the terrain, and not surprisingly the registration of animals and previous attacks and presence of hostile actors. Interesting

to see is the estimation of connectivity and the probability of attack during a certain time of day that is applied by APE. It can be suggested that these parameters are specifically interesting to ensure good results on a fine temporal mesh.

This brings us to the topic of expected results. The presented models are all used for patrol by foot, which is a significantly slower observation method then flying a drone. Furthermore, together with the fact that finding snares is probably easier than catching poachers red-handed, this results in the fact that time discrepancy is not required to be more precise than a few days (the time for a single patrol) to a few months (the time for a seasonal patrol strategy) and often is only dependant on updating the input data. APE is an exception, but for that model, the focus was more on real-time protection of a specific number of animals on a relatively small area with the availability of very fine-grained data. It is important to take into account the coarseness of input data to ensure a required level of temporal and spatial discretisation in the results while taking into account that processing more data will always result in larger computational complexity and longer computational times.
As can be seen in Table 2.1, it is hard to compare the performance of all models. There is no standardised metric that analyses the performance of the entire system. This is because of the differences in (presence of) model components, which makes it hard to compare different models with each other. Where possible, the model performance has been expressed relative to other models. It is important to keep in mind that all aspects of a model can heavily influence its performance, therefore it is necessary to select the components best suited for the situation and build a model with those components. After all, the research is about optimisation.

At last, some of the models combine the calculation of the actual patrol route with optimising expected results of the patrol. Both topics can be separated, but the combination proved to be efficient since the patrol itself is an extra constraint that can help to eliminate non-feasible options in the solution space.

Table 2.1: Summary of model characteristics

Model	Performance	Comp. Time[1]	Cell Size	Time Discr.	Data Dependence	Spatio-Temporal relation	Scheduling constraints	Model Components
PAWS	/	/	1km x 1km	every turn of game (3mo)	historical data and expert knowledge	/	Yes	Predict animal and human activity distribution Behavioural model based on SUQR with MLE parameter estimation Patrol strategy based on SSG with iterative learning Build ”street map” and calculate patrol strategy
CAPTURE	Outperforms SUQR (PAWS)	EM param. est.: 13321 sec. incl. patrol planning: up to 6h	1km x 1km	every turn of game; seasons (3mo)	historical data	Temporal	No	Attackability and observability est. by logistic regression Patrol planning (without constraints) based on game theory, solved by PWL approx. and non-convex solvers
MINION	/	/	1km x 1km	every turn of game	historical data (only in ML model)	/	Yes	Patrol planning by online learning FPL-UE based algorithm Attackability est. and patrol planning by ML model Game theoretic model chooses best performing strategy each round
APE	75%-100% of animals survive (66% without APE)	80-100 min for QENP-like data set size	400m x 400m	hour (animal model); daily strategy	historical data and expert knowledge	Spatio-temporal	Yes	Attackability by GPR and expert data Animal movement model as pSTG Optimise ranger routes as STG’s by local-search and gen. algorithms
INTERCEPT	outperforms CAPTURE	faster than CAPTURE	1km x 1km	none	historical data and expert knowledge	Spatial	n.a.	Data pre-processing by eliminating contradictory data Attackability by DTB of 5 DT’s with spatial feature
Decision Tree	expected equivalent to capture	relatively fast	1km x 1km	none	historical data and expert knowledge	Spatial	n.a.	Data augmentation by DD, NS and PS Attackability estimation by DTB
Neural Network	poor	relatively fast	1km x 1km	none	historical data and expert knowledge	Spatial	n.a.	Data augmentation by DD, NS and PS Attackability estimation by NN
DTB + MRF Hybrid	CPUE up to 0.12 (vs 0.04 av.)	DTB: 71 sec MRF: 31115 sec Hybrid: 10348 sec	1km x 1km	1y (good performance) 3mo (bad performance)	historical data and expert knowledge	Spatio-temporal	n.a.	Attackability by DTB augmented by MRF
iWare-E	equivalent to DTB+MRF Hy.	attackability est.: 175 sec	1km x 1km	3mo period	historical data and expert knowledge	Spatial	Yes	Data pre-processing by significany weighting of data points Attackability by DTB with uncertainty thresholds Planning constraints as max. network flow in time unrolled graph Patrol planning by maximising total detected attacks as MILP

[1] All exact times are given for complete execution of QENP data set on a machine with 2.6GHz and 8GB RAM.

Table 2.2: Summary of data attributed to every cell for every model presented.

Data features		**PAWS**	**CAPTURE**	**MINION (ML alg.)**	**APE**	**INTERCEPT**	**DT/NN**	**DT+MRF**	**iWare-E**
Invariable	Distance to		villages	villages	houses/buildings	villages	village	(villages?)	villages
		patrol post	patrol post	patrol post	patrol post	patrol post	patrol post	(patrol post?)	(patrol post?)
		water	water	water	water	water	river stream marsh	(water?)	(water?)
		road	road	road	road	road	village road provincial road national road highway	(road?)	road
			(border?)	border		(border?)	border	border	border
					# of cells reachable by poacher in 20 min				
	Environment	land type	habitat	(habitat?)		habitat	habitat	habitat	(habitat?)
		elevation			elevation		elevation		
			slope	slope	slope (max. elev. - min. elev. per cell)	slope	slope	slope	slope
			animal density	animal density		animal density		animal density	animal density
			season			(season?)			
			NPP			NPP			
					expert est. av. ToD of attack		expert threat score of 1 to 10		
Variable	Presence	patrol effort	patrol effort	patrol effort		patrol effort	patrol effort	patrol effort	patrol effort
		human activity	(human activity?)	(human activity?)		(human activity?)		(human activity?)	(human activity?)
		crime	crime (per type)	crime (per type)	crime (per type)	crime (type?)	crime	crime (per type)	crime (per type)
		animal	animal		animal GPS data	animal	animal		

2.2. Solution Techniques for Drone Surveillance

The following section makes a decision on which solution components would be best suited for the drone surveillance security problem based on its differences with foot patrol route planning.

The Effect of Differences Between Foot Patrols and Drone Surveillance

The differences in the problems of the presented models and the drone surveillance problem are obvious. Where foot patrols travel at slow speed on the terrain itself, the drone will fly at an altitude ranging between 150m and 200m at a minimum speed of 80 km/h. Furthermore, the area's that are considered for the models above are large conservation park measuring up to 2,500 square km with little internal borders. The patrols, however, take place only over a few square km every time and are on the ground. The environment for drone surveillance in the case with Eyeplane will consist of multiple, possibly dispersed, enclosed farms or private game parks of eight by ten square km on average within a region approximated by a circle with a radius of 50 to 100 km. This will result in differences in the following parts of the solver:

Temporal and Spatial Discretisation

The speed at which drone surveillance can be done will require an input that has a higher temporal discretisation, but hopefully, its observations will also result in a higher level of discretisation than what has been available for the ranger patrol models.
Spatial discretisation will depend on the problem size that can be handled within a certain computational time, but possibly also by the dimensions of the observation frame of the drone and the ability to narrow down attacker locations (eg. due to their presence on roads, next to terrain borders, near hiding spots, etc.)
Furthermore, it is important that the difference in speed with human beings will also result in a difference in rhythm of actions, in other words meaning that a drone can patrol every hour, but an attacker might not be able to attack every hour.

Data

The ability to quickly scan the area to surveil allows to regularly generate precise data about the location of animals relatively fast. Also, changes in the environment are noticed rapidly, for example regarding the location of water.
It is important to realise that the areas are used commercially and can be very well documented and/or will see relatively more human activity than large conservation areas.
The drone will also allow performing surveillance flights during the night, which might make it interesting to add data regarding nighttime to the data set, e.g. moon phase [12].
Furthermore, the role of rangers, being the defenders in all presented models, will also have to be evaluated to take into account their effect when surveilling attackers by drone.

Observabillity

Naturally, the method of observation is different for a human being than for a drone. The advantage of a large overview, infra-red sensors and the ability to move faster than an attacker also knows downsides like not being able to see through vegetation and other obstacle and being further away from the target. It will not be possible to spot hidden snares or other small signs of human presence from high up in the sky, meaning that actually seeing attackers is the only way to gather data about the attacker.
Next to observability *by* the drone, there is also the case of observability *of* the drone. This phe-

nomenon is already introduced by Bondi [4] and will have to be investigated further.

Patrol Planning
Although the actual route between targets chosen to surveil is the main subject of the next chapter, it is important to realise that the rapid rate of information gathering by a drone might be of use to adapt flight route in real-time, for which the addition of real-time variable parameters is required.

Preferred Solution Model Components for Drone Surveillance

Given the expected differences presented above, the different aspects of a drone surveillance model will be characterised as follows.

Data
For invariable data, similar types of data will be collected. This data is largely available as open-source data on Google Earth and can be collected on-site.
Regarding variable data, it is expected that if available, initial poaching data for the development of this model is not sufficient nor uniformly available for the total area to surveil. However, the fast expected rate of patrol might result in rapid acquisition variable data that can subsequently be used.

Data (pre-)processing
Given that no poaching data is available initially, pre-processing the data to balance the positive and negative data is not necessary. However, when data will be collected, initial class imbalances will exist and will have to be taken into account.
The practice of spatial clustering of data that Gholami et al. [8] apply will be suitable considering that the area that needs protection is composed of different smaller areas with their own distinct characteristics (economic activity, accessibility, animal population, etc.).

Attacker Behaviour Model
The lack of initial poaching data narrows down the choice of approach that can be taken to solve the problem. Consequently, the first step to model attacker behaviour will probably be most successful when assuming no data is available and to develop a model inspired by MINION-sm. This means that the problem will have to be formulated as a GSG adapted to the differences identified above.
An issue with this approach is that there is no comparable data to evaluate MINION's results or run time.
As observations of animals and poachers will occur over time, an algorithm that takes the new data categories into account needs to be available as well. Given the expected fine-grained temporal discretisation of the data, it will be an option to estimate a real-time location of animals and to take that data into account for a parallel probabilistic attackability model to enhance the results obtained by the game-theoretic behaviour model.

2.3. Conclusion

Previously developed models that aimed at estimating place and/or time of attacks on wildlife conservation zones to assist in patrol planning are diverse in pre-processing data, methods of modelling attacker behaviour, and planning the actual patrol route. Their similarities lie in the fact that all discretise the are in a similar way, have been developed for slow, long-term patrols or even patrol strategies.
The main part of every model is the calculation of the probability of attack. This is either based on a

game-theoretic model consisting of attackers and defenders, or by a probabilistic calculation that integrates attacker behaviour merely as previous attacks. When little data is available, the game-theoretic approach is deemed more successful. When large quantities of data are available, classic regression or classification techniques provide a satisfactory result in a significantly shorter time.

To apply the presented techniques to the use of drone surveillance, notable differences need to be taken into account regarding temporal and spatial discretisation, available types of data, relative speed differences between attackers and defenders, observability and patrol planning. Adaptation of the presented techniques is necessary and will result in an approach that combines game theory and classification/regression techniques, and provides fast adaptation to new input data.

3

Path Planning: How to Fly

A flight path of an aircraft is the trajectory that it follows through space as a function of time. Conventional air travel which has as a purpose to transport goods or people usually has a flight path originating in a certain point A with as destination a certain point B. However, the actual path between those points rarely is a straight line connecting both, but it is influenced by wind or air currents, dedicated (no-)fly zones and other air traffic. This chapter will elaborate on the problem formulation of the flight path for drone surveillance and how to solve that problem.

3.1. Flight Path for Drone Surveillance

Contrary to conventional air travel, the goal of surveillance is not transportation from A to B, but rather observation over several points, routes, or even zones, and the origin and destination of the aircraft are usually the same. Based on the outcomes of the security problem in chapter 2 it can be stated that the flight path will have to cover a certain number of high-risk grid cells in the region. It is yet not possible to determine how exactly these points will be distributed, and there is also no possibility to say how they will not be distributed. The range of possible relative locations is large, especially since conditions can vary constantly and because the expected trade-off between exploration and exploitation of the terrain does not allow for a confident intuitive estimate of a flight path. The most suiting general schematic approximation of the flight path problem would therefore be the Travelling Salesman Problem formulation.

Travelling Salesman Problem

The Travelling Salesman Problem (TSP) figuratively addresses the following question: ”What is the shortest possible route in a set of cities that visits each city once and returns to the starting city?”. This problem is NP-hard and has been studied extensively as it is a good benchmark for optimization methods in a variety of applications. In general, the problem is modelled as a graph where the ”cities” are the vertices, the ”paths” are the edges, and the distance is the edge’s weight (both sets of vocabulary are used). The objective is to minimise the combined weights of all edges on the route.
The TSP can be modelled as a symmetric or an asymmetric graph. A symmetric graph has the same weight connecting A to B as connecting B to A, but an asymmetric graph might give a different weight to travelling in the opposite direction. TSP’s can also be modelled with or without triangle inequality.

A graph consisting of vertices A, B and C all connected to each other satisfies triangle inequality if the direct route from A to B has lower or equal weight than the route from A to B through C.
There are different variations to the TSP that add specific requirements to the problem. The bottleneck TSP tries to find the route with the minimal weight of the weightiest edge and the maximum scatter TSP maximises the minimum edge weight. The generalised TSP, or ”travelling politician problem” is formulated as ”states” with multiple ”cities” and one city of every state has to be visited. The partial TSP solves a route for a certain amount of cities in the total set. In the prize-collecting TSP, every vertex is also associated with a penalty, which becomes an added variable in the minimisation of the total weight. Similar but more advanced is the travelling purchaser problem, where a purchaser has to buy a set of products in several cities while minimising the total cost. Not all products are for sale in all cities, and the cost per product can vary in every city.

Depending on the additional variables that will have to be taken into account for the drone surveillance in Eyeplane’s case or for generalisation of the problem, the TSP-based problem formulation might vary. For example, taking into account the wind and its changes over time might change the weights between vertices in a way that the graphical representation of the problem becomes an asymmetrical graph that does not satisfy the triangle inequality. Operational requirements or customer’s requests might result in the addition of extra weights or costs to edges or vertices.
The choice of TSP formulation will also influence the choice of solution methods since not all solution methods apply to all problem formulations.

3.2. Solution Methods for the Travelling Salesman Problem

In the long time that the TSP has been studied, multiple solution methods have been presented to solve this NP-hard problem. The three main categories of solutions methods are exact solution methods, approximation methods and heuristic methods. Depending on the specific TSP for which they have to find a solution their relative performance will be different. The performance of the solution methods is usually measured in time complexity and solution approximation. Time complexity is a function of the inputs (often denoted with the Big O method) and the solution approximation is the fraction of the achieved solution over the optimal solution.
Furthermore, humans appear to be relatively good at producing solutions for the TSP as well.

Exact Solution Methods

Exact solution methods for the TSP are able to achieve the optimal result. The most straightforward one is to calculate all possible routes and checking which one is the cheapest, also known as *brute-force search*. This method requires the calculation of the factorial of the number of cities, which makes it impractical for even a small number of cities (e.g. a graph with 10 cities and paths between all of them would have $10! = 10 \cdot 9 \cdot 8 \cdot 7 \cdot 6 \cdot 5 \cdot 4 \cdot 3 \cdot 2 \cdot 1 = 3,628,800$ possible solutions).
Various linear programming-based algorithms and dynamic programming algorithms can solve the problem in less computational time. The current best program to solve the TSP is the *Concorde TSP Solver* which is based on branch-and-cut methods, combining branch-and-bound methods with cutting-plane methods which systematically reduce the number of solution candidates [1].

Approximation Methods

The algorithms above might produce an exact solution but still require large computational times. In practice, a solution that approximates the optimal solution but does so in a significantly shorter time is

preferable. For smaller problem sizes, it might even reach an optimal solution. Approximation methods trade off computational time with solution precision, but still require a mathematical proof that guarantees the performance of the worst-case result [24].
A simple approximation algorithm is the nearest neighbour (NN) algorithm. Being a greedy algorithm, it chooses the locally most optimal solution every step of the way, meaning it goes to the nearest unvisited city at every move. This quickly generates a short route, but can also generate uniquely bad results, especially when used on asymmetric graphs [2]. A similar version is the nearest fragment algorithm, that connects groups of unvisited cities and solves those groups internally. This reduces the risk of costly connections when the number of remaining unvisited cities decreases.
A very well-performing approximation algorithm is Christofides' algorithm, that is based on Euler's Theorem that states that in order to have a path over a graph that visits every edge exactly once, it is required that all vertices have an even number of edges. Knowing this, an optimal Eulerian graph can be made by finding a minimum spanning tree and matching the odd vertices. Finding the Eulerian graph with the available edges is a route for the TSP as described by Christofides. This method guarantees a lower bound of 1.5 the optimal solution.

Heuristic Methods

Heuristic methods also try to approximate the result in a faster way, but unlike approximation methods, there is no mathematical proof for the quality of the lowest result and thus no proof of whether the solution might succeed at all. However, they are capable of finding very precise results for graphs with a lot of vertices.
In general, V-opt heuristics are considered to be the best performing heuristics for the TSP. Its main principle is based on making a random tour and iteratively removing a variable number of V edges and reconnecting the segments to make a shorter tour. A variation of V-opt is K-opt, where the number of replaced edges is constant.
Match Twice and Stitch by Kahng and Reda [13] is another heuristic that first makes two sequential matchings creating two cycles and subsequently stitches both cycles to create one tour.
Insertion techniques rely on creating a tour with a subset of the available nodes and looking to include the nodes afterwards. When considering the exclusion of nodes as well, these methods can be used for optimal route planning with maximal time or distance constraints.
Many heuristics that are developed for combinatorial optimisation in general also can be applied successfully to the TSP, like the tabu search methods. Also, methods inspired by natural phenomena are applied to the TSP like genetic and evolutionary algorithms, simulated annealing algorithms, ant colony optimisation, swarm intelligence methods and artificial neural networks.
Heuristic methods are also often used in combination with approximation methods. For example, one of the best performing V-opt algorithms uses genetic mutation to escape possible local optimums where the algorithm converges to.

Human Performance

When presented with a graph of 10 to 20 cities, humans tend to outperform several heuristics and can often achieve the optimal solution. Even for graphs up to 120 cities, the performance was on average 11% above the optimal solution [17]. The performance can vary greatly from individual to individual, and the graph geometry or distribution of the cities is suspected to play a large role as well. Nevertheless, it can be interesting to make use of an initial solution given by humans and try to improve them using the methods above.

3.3. Solution Method Selection

Choosing which solution methods are best suitable for the flight path problem will greatly depend on the constraints that will be formulated for that problem and on the results produced by the security problem.
The constraints, that will result from technical details, drone specifications, customer requests, the weather, etc. will determine what kind of graph is most suitable to represent the situation as a whole. The results from the security problem will greatly influence the graph geometry [5].
Not all methods produce satisfactory results within adequate time. It is known that certain graph distributions can make the NN algorithm produce the worst results for example, and fast convergence of genetic algorithms is not always certain.
The uncertainty in specific problem formulation and the large set of solution methods with many variations in performance regarding the problem formulation requires to better study the problem first. In order to do that efficiently, the results from the security problem will be awaited first. Together with a more in-depth case study with Eyeplane, the flight path problem and preferred solution method will be studied for the testing phase.
Lastly, it is also important of evaluating the effect of the flight path on the performance of the drone itself and the execution of the calculated security strategy.

4

Agent-Based Modelling & Simulation

4.1. Introduction to Agent-Based Modelling and Simulation

As the name suggests, agent-based modelling (ABM) is a technique for modelling a system by defining characteristics of agents in that environment. These characteristics can be a goal or an interest, physical attributes, and movements, but also behaviour versus other agents or the environment. These characteristics might change over time or change conditionally based on different inputs from the environment or other agents. The environment can be defined as an abstract space, but it can also be given characteristics to the extent that it can replicate real-life complexity.

The core idea of agent-based modelling and simulation (ABMS) is to create a bottom-up multi-agent system (MAS) in order to investigate complex phenomena by simulation of the system. These phenomena consist of the emergence of global behaviour or unforeseen patterns in the system resulting from the actions and interactions between agents and/or the environment. This approach makes an agent-based simulation (ABS) particularly suitable to analyse complex adaptive systems and it has been applied in a wide range of subjects in the fields of engineering, computer science and mathematics, but also psychology, biology, economics and environmental sciences [20]. In line with Klügl and Bazzan [16], *ABMS* will be used to refer to the paradigm in general, *ABM* for the task of modelling and *ABS* for the execution of the simulations.

4.2. Advantages and Limitations of Agent-Based Modelling and Simulation

The main advantage of ABMS is the ability to model and study a system whose global system dynamics are not represented by a set of equations. This ability comes with the following traits [3]:

- ABMS can capture emergent phenomena
- ABMS provides a natural representation of a system
- ABMS is flexible

This entails that ABMS has some practical advantages. The ”natural” origin of the construction of the model makes it easy to construct and easy to translate back into practice since it is possible to eas-

ily identify which changes to specific components result in changes to the system's performance, or components could easily be added or removed. This also offers direct and specific experimentation ("what-if" experiments) to evaluate the system. Furthermore, the availability of information at the individual level and at the system level offer more validation options. The model output can be compared with the real system behaviour, and the definition of the agents and environment can be compared to local observations on the actual behaviour of the components in the system that are to be represented [22].

For simpler systems, however, or for systems where assumptions allow the system to be generalised, ABM might be more time-consuming than an equation-based model
Also, for natural representation of large systems with a lot of different agents, a large parameter space results in extensive work to model, simulate and validate the system. This is especially true when not all information about agents is available, or when the individual behaviour of agents is not rational or straightforward. Huge numbers of agents will also result in simulations that take a lot of time.
Furthermore, not every system is suitable to be modelled this way. If it is not clear how to identify different components of a system as agents, ABMS is not an optimal choice to analyse the system. Also, for systems with well-defined input-output relation or non-autonomous behaviour, or when an explicit model or a formal analysis of the model without simulation are required, more suitable modelling methods are available [7].

4.3. Multi-Agent System Representation of the Mission Planning Problem

The choice for modelling the drone surveillance mission planning problem as a multi-agent system is straightforward when considering the different components of the problem.
However, when using ABMS some challenges will arise. These are described below.

System Components

Environment

The area where the surveillance takes place will be modelled as the environment. The discretisation of the area will be done by means of a square grid, which result in the grid cells that have been discussed in chapter 2. In ABMS lexicon, these grid cells are called *objects* that together constitute the environment and where agents can act upon.
All objects can be assigned variable and invariable characteristics, just like presented in Table 2.2.

Agents

The actors in the surveillance system are classified into three different categories:

- Assets: anything within the area that needs to be protected e.g. cattle, wildlife, civilians, etc.
- Attackers: anything that trespasses the area and can cause harm to the assets e.g. poachers, predators, enemy soldiers, etc.
- Defenders: the ones tasked with the protection of the assets e.g. rangers, police officers, soldiers, etc. and/or UAVs.

If the formulation of models presented in chapter 2 is closely followed, drones will also be modelled as a type of defender. This is most probable, since the practical implementation of the drone surveillance

will not account for control over defenders, and the observational roles between both are similar in essence. Their existential characteristics, however, are very different, for which the choice might still be made to model them as a completely different agent category.
All actor types have different goals and characteristics and are fairly autonomous, which makes ABMS a suitable way to represent them.
Furthermore, Parunak et al. [22] concluded that ABMS is most appropriate for ”domains characterised by a high degree of localisation and distribution and dominated by discrete decisions”, which is exactly what the system of actors in this system represents.
In general, the distinction can be made between two types of agents, reactive and proactive agents. Reactive agents solely react to changes in their environment, whereas proactive agents are also able to process those changes with an internal model. They can also be described as having a goal and a plan to achieve that goal. In relation with the drone surveillance problem, the assets will be sufficiently represented by a reactive agent (or even be reduced to a characteristic of the object corresponding to their location), whereas attackers and defenders are definitely proactive agents, enabling them among others to interact in the ways defined by the GSG.

Inter-agent and agent-environment relations
The behaviour between the agents and the agents and the environment is only known at the local level, and the global dynamics of the problem are not easily identifiable on first sight. ABMS will reveal what global and individual behaviour can be expected given the information that is available about current individual characteristics and relationships.
The relationship types in the drone surveillance problem are as follows:

- **Environment - Agents**
  Agents can observe the environment. The exact information available to the agent and how it will be processed will depend on the agent itself.
- **Assets - Attackers, Defenders**
  The assets will be observable by the other agents and are able to observe each other. Whether the assets will be able to observe the other agents and what their reaction will be is to be determined.
- **Attackers - Defenders**
  The direct relationship between attackers and defenders is that they will be able to observe each other. The negotiation by means of game theory is based on a combination of these observations, observation of the environment and assets, and the subsequent predictions made based on these observations.

Challenges when using ABMS

Solving the drone surveillance problem in ABMS will also present some challenges and disadvantages. For one, the entire system is rather complex in reality, and determining to what level this complexity (e.g. rationality) should be modelled is a topic that will emerge while modelling.
Furthermore, ABM’s are not easily backed up mathematically, since it might be hard or even impossible to generalise the emergent behaviour that results from the system. This will make it different to validate and for other parties to understand with relation to conventional stochastic models.

III

Supporting Material

5

Model Elaborations

In this chapter, more information about the agent-based model is presented. The aspects that are elaborated upon are a more detailed account of all assumptions, in section 5.1, and a more specific description of the observation model can be found in section 5.2.

For illustrative purposes, Figure 5.1 presents a diagram showing the specific relations between different parts and variables of the model.

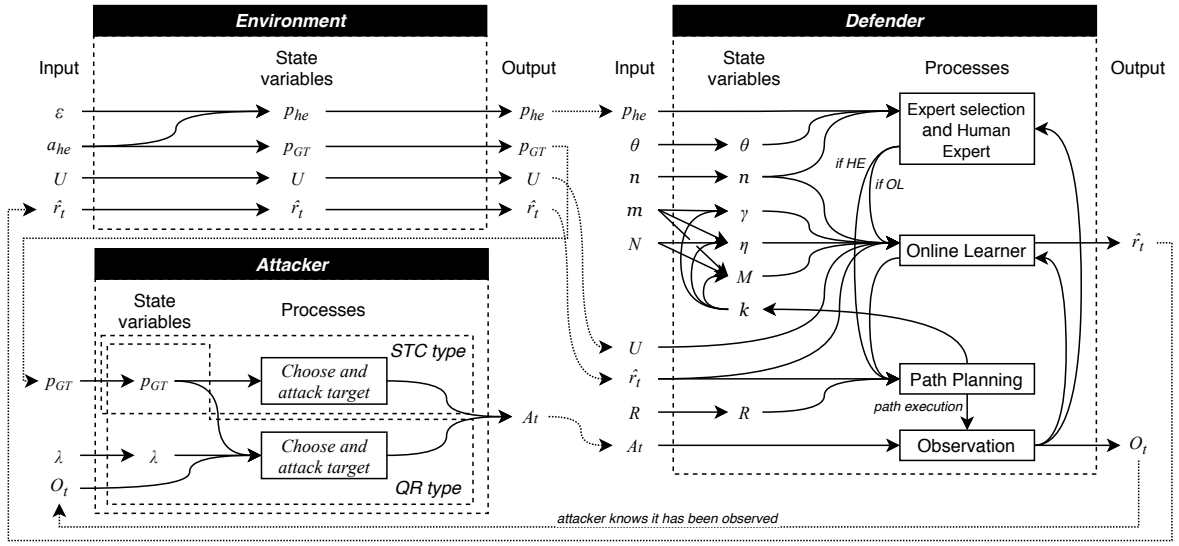

Figure 5.1: Diagram of the components of the Agent-Based Model and their interactions that occur during one round.

5.1. Assumptions

In this section, all specific assumptions regarding the agent-based model are listed. They are categorised according to the model component they apply to, either the environment, the defender, or the attacker.

Environment

1. The environment is considered two-dimensional, meaning the effect of elevation variation on the terrain is not taken into account.
2. The environment is static throughout the duration of the game, e.g. the impact of moving animals, changing vegetation, or amount of sunlight is not considered.
3. The discretization of the environment depends on the field of view of the drone.

Defender - drone

1. The drone flies at a constant speed.
2. The drone flies at a constant altitude.
3. The drone flies straight from waypoint to waypoint on its route.
4. The drone has a constant field of view.
5. The observational capacity of the drone is perfect. In other words, if the defender covers an entire target where an attacker is present, the chance the attacker is observed is 100%. If the target is only partially covered, the chance the attacker is observed is relative to the fraction of the covered area of the target.
6. The duration of the patrol flight is the duration of the round.
7. The nature of the game implies that when an attacker attacks, the defender always defends simultaneously.

Attacker - poacher

1. In a single round, one attacker attacks one target.
2. The attacker is located at the target it attacks for the entire duration of the round.
3. Attackers cannot attack the same target during the same round.
4. Attackers do not coordinate their attacks during the same round.
5. If multiple attackers are of the Quantal Response type, they are aware of how many times they have been observed by the defender at a certain location previously in the game.

5.2. Details on observation algorithm

This section explains step by step how the observation model calculates which targets are covered in a certain route by the defender and what the fraction of coverage is per cell.
In short, as explained in the paper, the algorithm draws a rectangle of width $w = \sqrt{2l^2}$, where l is the width of a target, and length $d_{i,j} + w$, where d_{ij} is the distance between targets i and j, so that arc $a_{i,j}$ coincides with the longest centerline of the rectangle. Afterwards, for every target that has an overlap with this rectangle, the fraction $frac_i$ of the observed area of the target over the total area of the target is calculated. The exemptions to this are the starting point i and the targets right next to i and j that are not in the line of the path, in order to prevent them to be counted twice.
As an example, a schematic of coverage between target $(1, 1)$ and $(5, 2)$ is shown in Figure 5.2, where

$frac_i > 0$ for targets $(2,0), (2,1), (2,2), (3,1), (3,2), (4,1), (4,2), (4,3)$ and $(5,2)$. this example will be used throughout this section. The explanation is broken down by describing the input, determining where the rectangle is situated, and calculating what the observed fraction of the covered targets is.

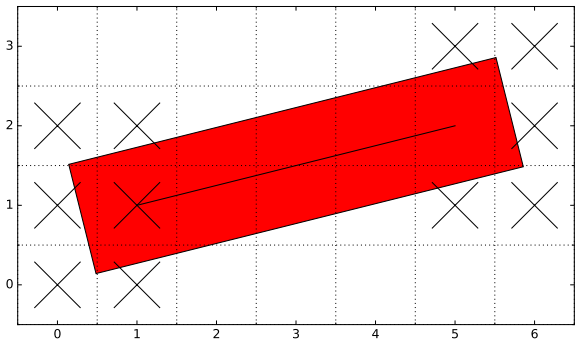

Figure 5.2: Coverage between target $(1,1)$ and $(5,2)$.

Input

The input to the observation algorithm are the coordinates of two consecutive waypoints i and j determined by the path planning algorithm. These coordinates are the centerpoints of the cells that make up the grid that represents the environment. For calculating the observed fractions of cells for the entire route, the observation algorithm is run for every consecutive pair of cells in the flight path determined by the path planning algorithm.

Drawing the rectangle

To draw the rectangle, the location of the corners of the rectangle has to be determined. To do that, we first choose the cell with the lowest x coordinate as the starting point $start$ and the other cell as endpoint end and determined the slope α of the arc between those points in that direction relative to the x axis. In the example, point $(1,1)$ is the starting point and point $(5,2)$ is the endpoint.
With this information, we can calculate the location of the four corners c_1, c_2, c_3 and c_4 of the rectangle. As an example, the equation for the (x,y) coordinates of corner c_1 is given in Equation 5.1

$$c_1 = \left(x_{start} - w/2 * \sqrt{2} * \sin\left(\alpha + \frac{3}{4}\pi\right) \quad , \quad y_{start} + w/2 * \sqrt{2} * \cos\left(\alpha + \frac{3}{4}\pi\right) \right) \tag{5.1}$$

In Figure 5.2, c_1 is the corner in cell $(0,0)$. We call c_2, c_3 and c_4 the corners in cells $(6,1)$, $(6,3)$, and $(0,2)$ respectively.
Now the line segments between the corners can be defined. They represent the boundaries of the drone's field of view. they are defined as follows:

Table 5.1: Starting- and endpoints for the sides of the rectangle defining the field of view of the drone between two waypoints.

Line	From	To	In example figure
l_1	c_1	c_2	'bottom'
l_2	c_2	c_3	'right'
l_3	c_3	c_4	'top'
l_4	c_4	c_1	'left'

Determining cell coverage

The first step is to identify all cells that are covered. This is done by calculating which cells' centerpoints are within distance w of the centerline between i and j.

Secondly, from the cells in the initial selection, cells that we choose not to take into account are identified and removed from the selection. These cells are the corner cells, other cells at the extremities of the rectangle, and the starting points and endpoints themselves. They are marked with black crosses in Figure 5.2. Note that in the figure, the endpoint is not discarded for illustrative purposes, showing that the waypoints are not discarded in the final observation calculation. These are not considered covered, since subsequent observation calculations between the last waypoint in this calculation and the first following waypoint on the route will possibly have overlapping rectangles. We chose to make the conservative choice of not counting them at all rather than twice. The coverage of the waypoints themselves is added separately later on.
The corner cells are already identified in the previous subsection. The other cells at the extremities are identified by looking at the direct neighbours of the corner cells that are also in the initial cell selection. We assume the coverage of these discarded cells to be 0%.

Lastly, we look at the covered areas of the remaining cells in the selection, e.g. all cells that are not crossed out and (partially) covered by the red rectangle in Figure 5.2. It can be observed that the rectangle boundary always divides the cell either in (a) a triangle and an irregular pentagon, or (b) two trapezoids. By determining the points on the cell boundaries where the rectangle and cell boundary intersect, it is possible to determine if it is divided like (a) or like (b) and to calculate the areas of the two parts. When the points are on parallel cell boundaries, it is situation (b), and it is situation (a) otherwise. To determine which part of the cell is covered and which part is not, the distance from the centerpoint of the cell to the centerline of the rectangle can be used. If the distance is $> w/2$ the smallest of the two parts of the cell is covered (see cell $(2, 0)$ in Figure 5.2). If the distance is $< w/2$ the largest part of the two parts of the cell is covered (see cell $(2, 1)$ in Figure 5.2). If the distance is exactly equal to w, half of the cell area is covered.

After calculating the covered area of every cell, this area is divided by the total area of the cell to arrive at the fraction of coverage of every cell. This fraction now represents the chance that an attacker is observable if he has attacked that cell during the current round.

6

FPL-UE parameter values

This chapter discusses the proof presented for the upper regret bound of the FPL-UE algorithm by Xu et al. [25]. More specifically, the validity of this proof for our research is argued, since we use it to determine the values of the model parameters η, which defines the noise factor z, γ, which controls the balance between exploration and exploitation, and M, which is the number of simulations by the Geometric Resampling (GR) estimation. These parameters are directly implemented as part of the FPL-UE algorithm that is incorporated in the online learner. The mathematical proof for the FPL-UE algorithm is described in the paper by Xu et al. [25]. The reader is advised to consult the proof before reading this section.

First, the following lemma stated by Xu et al. [25] is discussed. It was proved by Neu and Bartók [18] and recaptures the bias of estimations from the GR method.

Lemma 1:

$$\mathbb{E}\left(\hat{r}_{t,i} \mid \mathcal{F}_{t-1}\right) = \left(1 - \left(1 - p_{t,i}\right)^{M}\right) r_{t,i} \tag{6.1}$$

Let F_{t-1} denote the history information of the game by time t (exclusive). Furthermore, as stated in this research, $\hat{r}_{t,i}$ is the estimated reward of target i at time t, $p_{t,i}$ is the probability that target i was chosen during round that round, and $r_{t,i}$ is the actual reward associated with target i at time t.

The main difference between MEOMAPP's OL and the FPL-UE algorithm is the addition of the path planning algorithm. The path planning algorithm tries to find an optimal path using the targets selected by the FPL-UE algorithm in the OL. However, not all targets put forward by the FPL-UE algorithm are included in the path by the path planning algorithm. This results in a different $p_{t,i}$ by the online learner in this research compared to the $p_{t,i}$ that would be produced without the path planner. However, we argue that lemma 1 still holds for the OL in MEOMAPP. The estimation of $p_{t,i}$ associated with those targets is still performed by the GR algorithm. This means that $p_{t,i}$ is still geometrically distributed, and thus the proofs presented by Neu and Bartók [18] still holds. The same notion of coverage probability is used throughout the rest of the proof by Xu et al. [25], meaning it is valid in that aspect.
Where the proof differs is in the notion of k, the number of protected cells by the defender. Xu considers this a constant in the experiments, where k can differ in MEOMAPP depending on how many waypoints

the path planning algorithm includes in its path. To achieve a smoother variation of η and γ we choose to average the number of selected waypoints as the game progresses to arrive at a gradually less variating substitute value of k. Considering the similar basis of estimating $p_{t,i}$ and the approximation of k, the proposed equations for η, γ and M can be used to determine their values in MEOMAPP. Those equations are:

$$\eta = \sqrt{\frac{k(\log N + 1)}{mT \min\{m, k\}}} \tag{6.2}$$

$$\gamma = \frac{\sqrt{k}}{\sqrt{mT}} \tag{6.3}$$

$$M = N \sqrt{\frac{mT}{k}} \log(Tk) \tag{6.4}$$

7

Plausibility checks

This chapter treats the plausibility checks that have been performed on the agent-based model used to simulate and evaluate MEOMAPP. Table 7.1 summarises specifically the principal tests that have been performed on each model component.
In general, the following types of plausibility tests have been performed:

- *Limiting case tests:* setting parameter values to boundary values (or approximating boundary values in case of infinity). The solutions to these limiting cases are usually straightforward and easy to check.
- *Magnitude checks:* usually it is possible to determine an expected range for a certain solution. In a magnitude check, solutions are checked to be within this range.
- *Constant of motion check:* verifying if expected constant values remain constant throughout the calculations, e.g. probabilities must add up to 100% at any point in time, or the number of attacked cells must correspond to the number of attackers.
- *Visualisation:* check visually on plotted data if the numerically computed results and corresponding behaviour are correct.

Table 7.1: Principal plausibility checks performed on agent-based model for simulating MEOMAPP.

#	Model component	Performed test
1	**Environment**	
1.1	Calculation of ground truth probability for attackers and human expert, based on attackability score	Probabilities of all cells add up to 100%
1.2	Human expert coverage probabilities disturbed by error factor	Probability of individual cell is within [0,1]
1.3		MAE of total probability vector compared to the ground truth is approximately equal to the error factor.
2	**Defender**	

2.1	GR algorithm	Number of samples does not reach self-imposed artificial threshold of 100 samples
2.2		Value of estimated reward gets updated with an integer multiple of twice its utility
2.3	Noise factor	Noise values have exponential distribution
2.4	Path planning algorithm	Length of the route was under specified range
2.5		Setting range to 0 results in only the base being considered for the ”route”
2.6		Manual recalculation of the total estimated reward values of a random route
2.7		Operation on simplified network with extreme estimated reward values produced foreseen results
2.8		Waypoints were connected to exactly 2 arcs (determined visually)
2.9		No subtours present in calculated route (determined visually)
2.10		No crossing lines in route (determined visually)
2.11	Observation model	Manual recalculation of expected rectangle corner locations in trivial cases
2.12		Manual recalculation of intersections between cell and rectangle in trivial cases
2.13		Manual recalculation of covered cell areas in trivial and non-trivial cases
2.14		Numerically calculated covered cell areas correspond with with plotted rectangle (determined visually)
2.15		Particular intersection cases tested and accounted for: rectangle gradient on 0, 90 and 270 degrees (sin or cos values equal 0) and 45 and 315 degrees (rectangle intersects with cell corners)
2.16		Discarded cells are selected correctly (determined visually)
2.17		Evaluate if output of observation model corresponds with the input from the path planning algorithm
2.18	Expert selection algorithm	Expert selection starts at correct step (determined visually)
2.19		Expert reward changes correspond to respective changes in expert selection
3	**Attackers**	
3.1	Likelihood that a target is defended	Outcome corresponds to ration of observations of the attackers by the defender over total rounds of the game.
3.2	QR attacker attack probability	Probabilities of all cells add up to 100%
3.3	No attackers on the same target in the same round	Total attacks at the end of the game is equal to the amount of attackers time the amount of rounds

4	**General**	
4.1	Average regret	The average regret value is never higher than the number of attackers
4.2		The average regret value over time shows the same conversion trend as results in previous research (determined visually)
4.3	Exploration / exploitation	The choice between exploring and exploiting follows the change in γ (determined visually)

8

Overview of experiment results

This chapter contains the plots of all experiments discussed in the research paper.

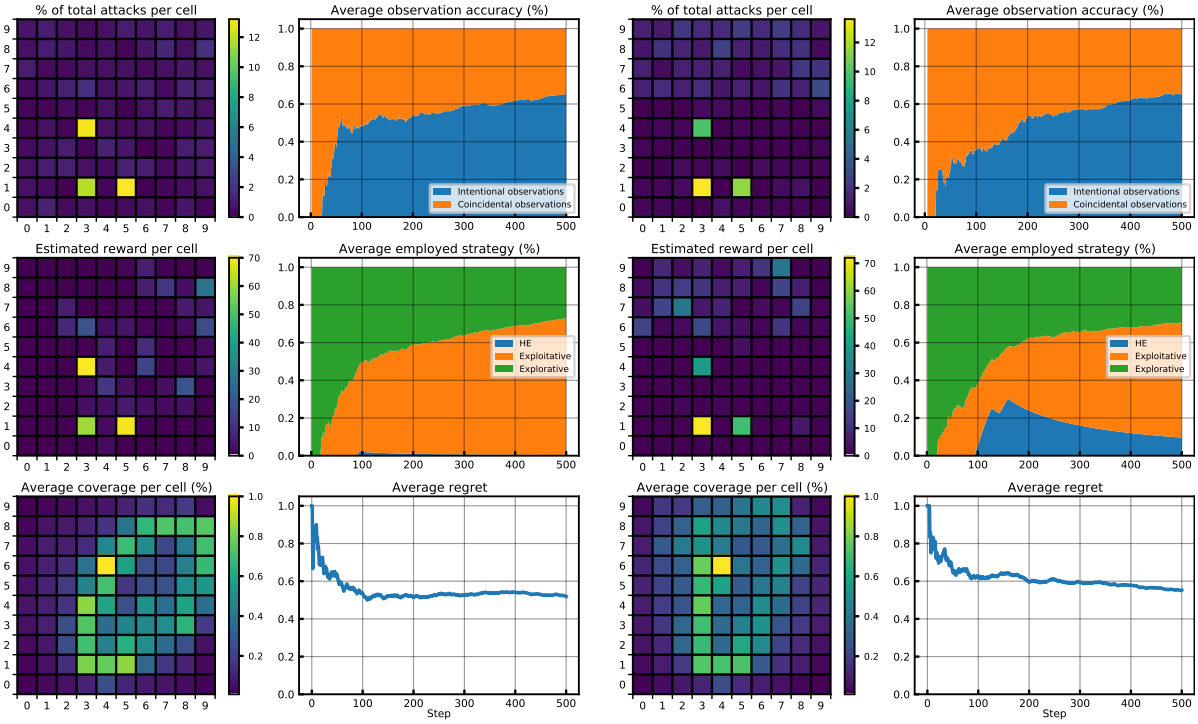

Figure 8.1: Simulation results from the baseline model with QR adversary on the left and STC adversary on the right.

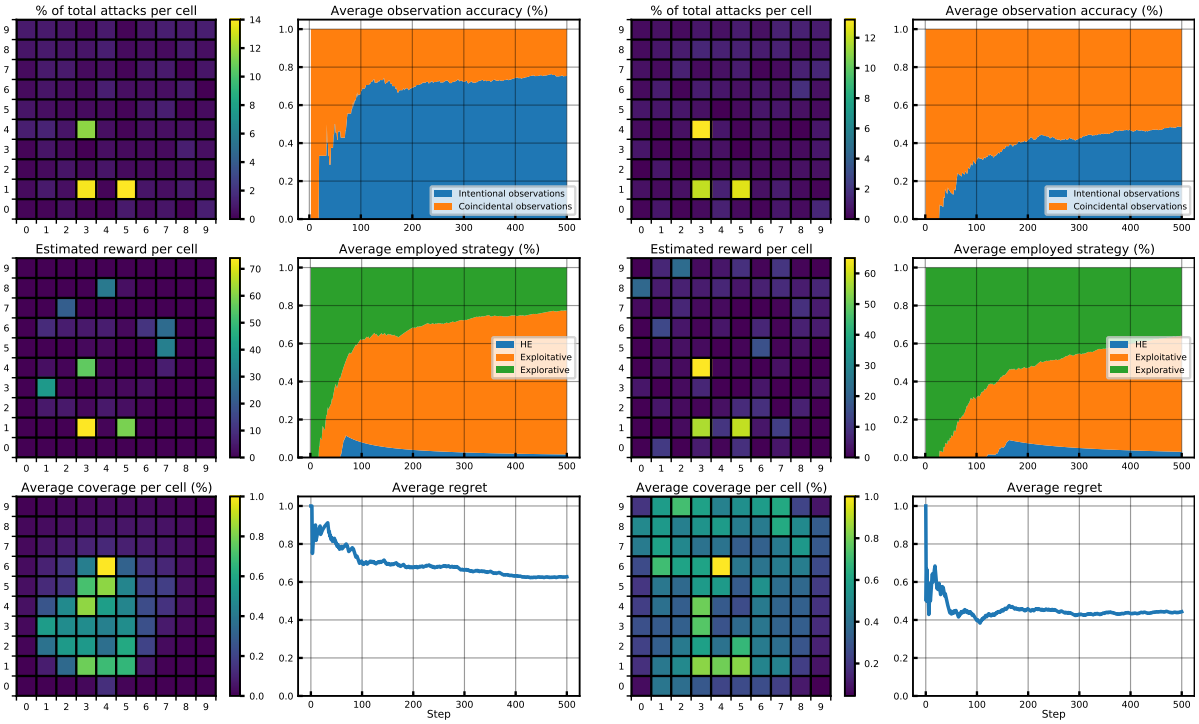

Figure 8.2: Simulation results of the baseline model against the QR adversary with $R = 15$ on the left and $R = 35$ on the right.

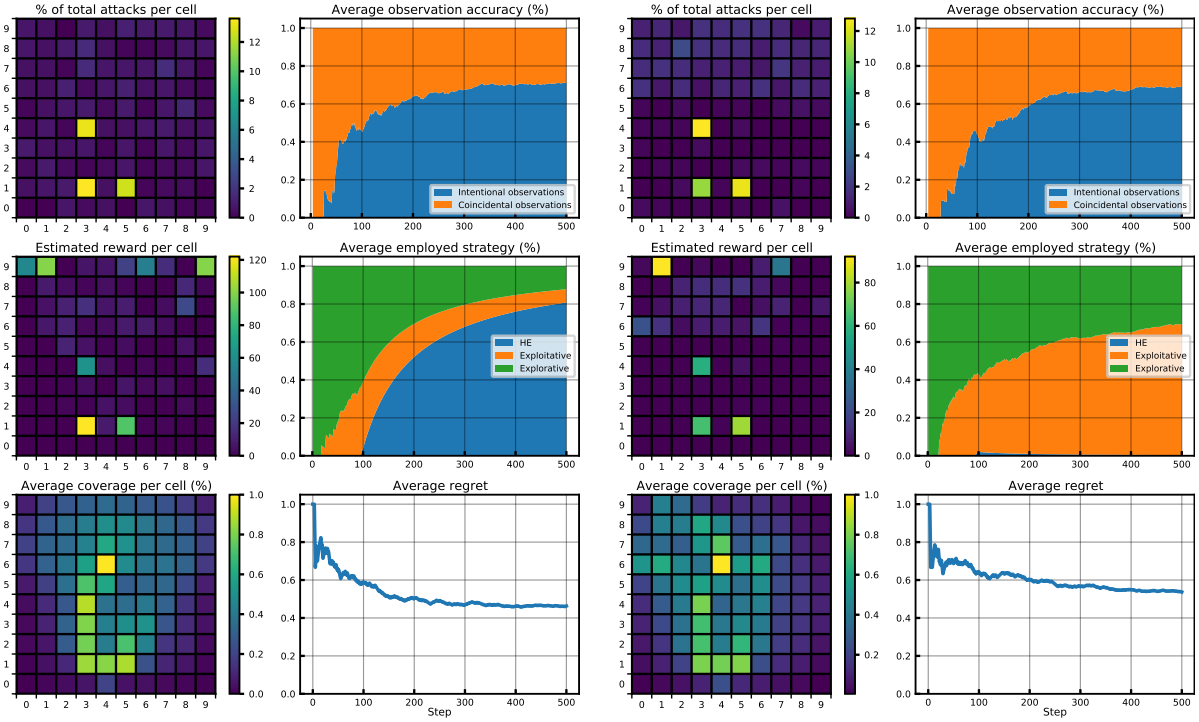

Figure 8.3: Simulation results from the base model with the HE's $\varepsilon = 0$ against a QR attacker on the left and an STC attacker on the right.

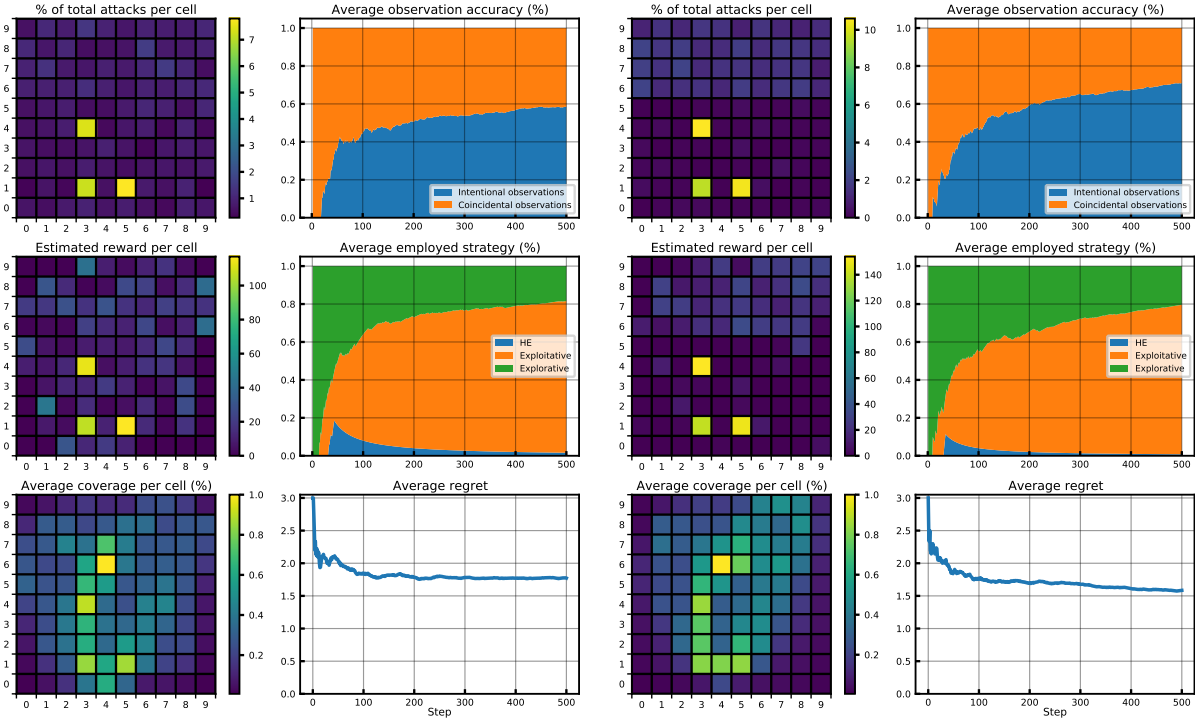

Figure 8.4: Simulation results from the baseline model with the $m = 3$ against a QR attacker on the left and an STC attacker on the right.

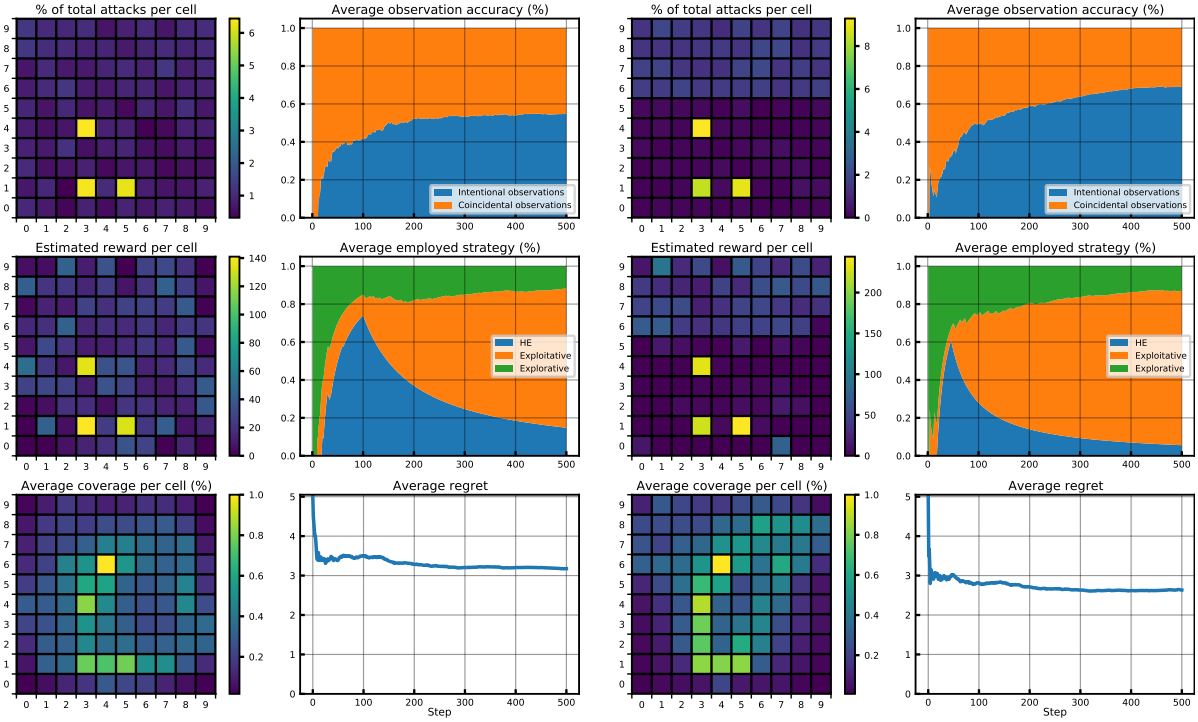

Figure 8.5: Simulation results from the baseline model with the $m = 5$ against a QR attacker on the left and an STC attacker on the right.

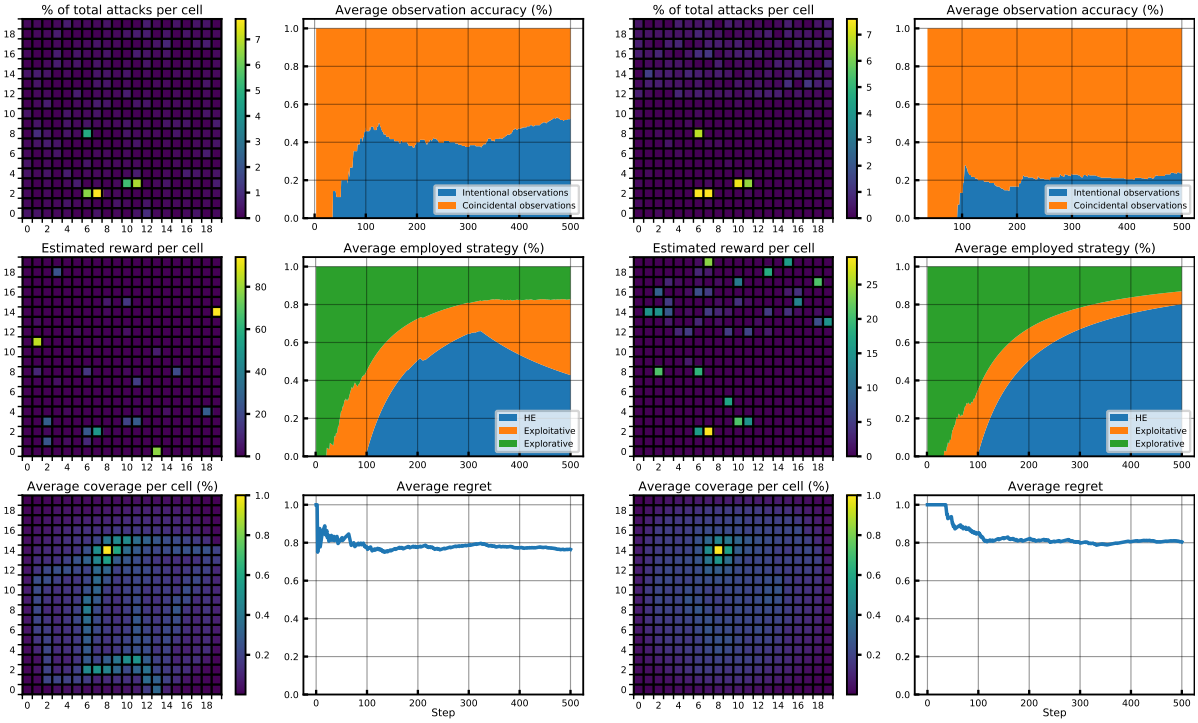

Figure 8.6: Simulation results from the baseline model on a 20x20 cell grid against a QR attacker on the left and an STC attacker on the right.

9

Original Research Plan

The original research plan was a deliverable in the initial phase of the graduation assignment and followed on the Literature Review in Part II. It is important to note that the research plan changed significantly during the research due to practical changes in the initial data availability and testing opportunities. The changes to the research plan have not been documented following the standards for the initial deliverable, therefore only the original research plan is included in this report.

9.1. Summary of the Literature Review

This literature review gives an insight on how to provide an answer for the main question of a drone surveillance mission planning research. The main research question reads

How to plan the flight path for a UAV on a surveillance mission in order to minimise the number of missed observations of attacks given its operational constraints?

From this question two sub-questions were derived, which formed the backbone of this literature review:

1. How to determine when and where to surveil?
    (a) How should a waypoint be defined?
    (b) How to evaluate the importance of surveillance of every point in the area?
2. How to determine the optimal flight path between interesting points of surveillance?
    (a) What is the most efficient way to calculate the optimal flight path?
    (b) How is the flight path constrained?

Question one was treated in chapter 2. The situation has been approached as a security problem in wildlife conservation. To provide more insight into the domain, related works about patrol planning for wildlife parks have been studied, along with the algorithms they propose to address this type of problem. After investigating the differences and similarities between foot patrol routes presented in the literature and this specific drone surveillance problem, it has been concluded that the area that has to be protected will be discretised in square cells forming a grid. The size of the cells is still to

be determined based on the required run time and computational power. These cells are the potential waypoints, which answers question 1.(a). Furthermore, it has also been decided that establishing which cell to surveil and which cells not to surveil is dependent on the probability they will be attacked and on the goal of exploring the area in general. Therefore, the relationship between attacker and defender will be formulated as a game in the class of Green Security Games (GSG) and the game will be solved by an algorithm based on game-theoretic utility, where possible augmented by a machine learning classification algorithm dependent on collected data. This answers question 1.(b).

Question two was treated in chapter 3. The chosen problem formulation for the flight path problem is the Travelling Salesman Problem (TSP). While investigating different formulations of the TSP and their solution methods, it has been observed that there is a difference in the relative performance of the solution methods depending on the formulation of the TSP. However, the exact TSP formulation for the flight path problem for drone surveillance cannot be determined yet. For one, this is due to the current lack of understanding of the local situation regarding operational constraints and stakeholder requirements. Secondly, the graphical representation of the TSP is dependent on the results of the security problem, which are not known yet either. Therefore, question 2 has only been addressed in a general manner, and a specific solution method will be presented during the research itself.

Furthermore, the choice to do this research based on the agent-based modelling paradigm (ABMS) is explained in chapter 4. ABMS is the paradigm of choice since it allows to comprehensibly representing a complex system like the one that is the case for this research. Furthermore, it is a paradigm that is capable of producing (un)expected emergent behaviour.
The representation of the drone surveillance system can be summarised as follows. The environment is represented cell by cell as objects with their respective characteristics. It and can be observed by the assets (animals, people, etc.), attackers (poachers, predators, etc.), and defenders (rangers, drones, etc.) who are modelled as different agents. The different agents have their respective characteristics and can interact with each other. Assets are expected to be reactive agents, and attackers and defenders will be proactive agents capable of observing each other and advanced processing of those observations and observations from the environment.

9.2. Knowledge Gap

Looking at the two subquestions, it is safe to say that the flight path problem can be solved optimally with existing knowledge. The TSP is a well-researched problem for decades and its multiple variants have been solved by a multitude of exact methods, approximation methods and heuristic methods.
The main gap is in the formulation of a GSG for drone surveillance in wildlife conservation. The effect of its differences in speed, range and observational capacity is unknown and will be an area of interest for new insights.
Furthermore, it can be said that the development of GSG's for models with no historical data is at a very early stage. It will be interesting to study the effect of the player variation on the performance of such a game-theoretic model.

9.3. Research Questions

The objective of this Master Thesis Project is to develop a tool that aids in planning surveillance missions for UAVs to protect wildlife. More specifically, the tool is to be tested with and used by Eyeplane

in order to optimise their surveillance missions over farms and wildlife parks in Namibia. The system and its components are to be represented by an agent-based model.

The main question this research aims to answer is

How to plan the flight path for a UAV on a surveillance mission in order to minimise the number of missed observations of attacks given its operational constraints?

The following questions were the basis that structured the literature review and so the preliminary research.

1. *How to determine when and where to surveil?*
2. *How to determine the optimal flight path between interesting points of surveillance?*

With the insight provided by the literature review, some follow-up questions where derived and presented below.

1. *How to determine when and where to surveil?*
   (a) What is the most suitable variable and invariable data to characterise targets?
   (b) What is the best method to optimise historic attack data for classification?
   (c) What if the effect of the faster gameplay on the Green Security Game?
   (d) How will the GSG be affected by possible observation *of* the drone?
2. *How to determine the optimal flight path between interesting points of surveillance?*
   (a) What is the influence of the flight path on the observation capacities of the UAV?

The implementation of this problem as an agent-based model also results in follow-up questions:

3. *How to represent this system of actors and their environment that need surveillance as an agent-based model?*
   (a) What is the effect of the model complexity (or adherence to reality) on the model performance and the outcomes of the strategy in real-world situations?

9.4. Project Plan

The project plan is derived from the problem structure and adjusted to the available time and knowledge. It consists of two phases of development and testing to allow for a gradual improvement of the model with two opportunities for tests and validation.
The following sections elaborate on items in the Gantt chart as far as possible.

Literature Study

This report marks the end of the literature study.

Model Development Phase 1

The initial development of the model consists of multiple simultaneous tasks, of which the most important is the development of the game-theoretic model representing the attacker-defender behaviour and resulting in attack probabilities for the grid cells. At the same time, the acquisition of the right Python programming language skills in Python to be able to model the system as an agent-based model. After the theoretical definitions, the actual modelling and subsequent verification can start.
Furthermore, it is important to start gathering as much data as possible right away to define assumptions and constraints that will influence the problem space for the models and algorithms.
Given the results from the security problem, an analysis of the path planning solution methods can be done.
It is important to have a result towards the end of development phase 1 in order to prepare implementation instructions for field tests with Eyeplane from August onward.
The goal is to deliver a working iterative model that can provide a new mission plan strategy after every flight.

Validation Phase 1

The validation phase will start with Eyeplane implementing the new surveillance mission strategies. Eyeplane will also be gathering data from their operations before the testing period, and the goal is that they continue doing so. These test results can then be analysed, after which changes to the model parameters can be done and testing can be continued with the adapted model.
At the same time, the report and presentation for the midterm evaluation can be prepared.

Development Phase 2

The goal for development phase 2 is to investigate the use of real-time observation data for real-time flight plan adjustments. Given the familiarity with the problem, the task of modelling and the developed algorithms, and the availability of data, a little less time is reserved for this development phase.
The goal is to deliver a workable model to Eyeplane for the subsequent testing and validation phase.

Validation Phase 2

Similar to validation phase 1, Eyeplane will implement and test the new mission plans, after which the newly acquired data will be used to adjust parameters and validate the model.

Final Phase

The final phase is important to look back on the achieved results and synthesise them thoroughly. Conclusions and further recommendations are the foremost contribution to the general public and serve as a starting point for the academic community.
This phase comprises finalising the research, its report, and possibly publishable material.

Deliverables

Official deliveries for this project consist of

- The literature review report providing the direction and steps of the research
- A mid-term presentation, summarising the followed approach, presenting the first results, and explaining the steps that still need to be taken

- A thesis work in the form of a paper to a scientific journal, adhering to the standards of the scientific community

Furthermore, regular (weekly or bi-weekly) progress meetings will take place with thesis supervisors, where performed steps and future steps can be discussed.

Considering the cooperation with Eyeplane, it is also required to deliver guidelines for implementation of the models and/or their strategies. Furthermore, the publication of the final paper will be with Eyeplane's consultation.

Bibliography Part II and III

[1] David L. Applegate, Robert E. Bixby, Vašek Chvatál, and William J. Cook. *The Traveling Salesman Problem: A Computational Study*. Princeton University Press, 2006.

[2] Jørgen Bang-Jensen, Gregory Gutin, and Anders Yeo. When the greedy algorithm fails. *Discrete Optimization*, 1(2):121 – 127, 2004.

[3] Ali Bazghandi. Techniques , Advantages and Problems of Agent Based Modeling for Traffic Simulation. *International Journal of Computer Science Issues*, 9(1):115–119, 2012.

[4] Elizabeth Bondi. Using Game Theory in Real Time in the Real World : A Conservation Case Study. In *Proc. of the 18th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2019)*, 2019.

[5] Vladimir G. Deĭneko, Michael Hoffmann, Yoshio Okamoto, and Gerhard J. Woeginger. The traveling salesman problem with few inner points. In *COCOON 2004: Computing and Combinatorics*, volume 3106, pages 268–277, Berlin, heidelberg, 2004. Springer.

[6] Fei Fang, Thanh H. Nguyen, Rob Pickles, Wai Y. Lam, Gopalasamy R. Clements, Bo An, Amandeep Singh, Brian C. Schwedock, Milind Tambe, and Andrew Lemieux. Paws — a deployed game-theoretic application to combat poaching. *AI Magazine*, 2017.

[7] Klügl Franziska, Christoph Oechslein, Frank Puppe, and Anna Dornhaus. Multi-Agent Modelling in Comparison to Standard Modelling. *Simulation*, 2002:105–110, 2002.

[8] Shahrzad Gholami, Benjamin Ford, Fei Fang, Andrew Plumptre, Milind Tambe, Margaret Driciru, Fred Wanyama, Aggrey Rwetsiba, Mustapha Nsubaga, and Joshua Mabonga. Taking It for a Test Drive: A Hybrid Spatio-Temporal Model for Wildlife Poaching Prediction Evaluated Through a Controlled Field Test. In *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, volume 10536 LNAI, pages 292–304, 2017.

[9] Shahrzad Gholami, Sara Mc Carthy, Bistra Dilkina, Andrew Plumptre, Milind Tambe, Margaret Driciru, Fred Wanyama, Aggrey Rwetsiba, Mustapha Nsubaga, Joshua Mabonga, Tom Okello, and Eric Enyel. Adversary models account for imperfect crime data: Forecasting and planning against real-world poachers. *Proceedings of the 17th international conference on autonomous agents and multiagent systems*, pages 823–831, 2018.

[10] Shahrzad Gholami, Amulya Yadav, Long Tran-Thanh, Bistra Dilkina, and Milind Tambe. Don't put all your strategies in one basket: Playing green security games with imperfect prior knowledge. In *Proc. of the 18th International Conference on Autonomous Agents and Multiagent Systems*, volume 9, 2019.

[11] Swaminathan Gurumurthy, Lantao Yu, Chenyan Zhang, Yongchao Jin, Weiping Li, Haidong Zhang, and Fei Fang. Exploiting Data and Human Knowledge for Predicting Wildlife Poaching. In *ACM SIGCAS Conference on Computing and Sustainable Societies 2018*, 2018.

[12] Timothy C. Haas and Sam M. Ferreira. Optimal patrol routes: interdicting and pursuing rhino poachers. *Police Practice and Research*, 19(1):61–82, 2018.

[13] Andrew B. Kahng and Sherief Reda. Match twice and stitch: A new TSP tour construction heuristic. *Operations Research Letters*, 32(6):499–509, 2004.

[14] Debarun Kar, Benjamin Ford, Shahrzad Gholami, Fei Fang, Andrew Plumptre, Milind Tambe, Margaret Driciru, Fred Wanyama, and Aggrey Rwetsiba. Cloudy with a Chance of Poaching : Adversary Behavior Modeling and Forecasting with Real-World Poaching Data. In *Proceedings of the 16th Conference on Autonomous Agents and MultiAgent Systems*, number May, pages 159–167, 2017.

[15] Richard M. Karp. On-line Algorithms Versus Off-Line Algorithms: How Much is it Worth to Know the Future? In *IFIP 12th World Computer Congress on Algorithms, Software, Architecture - Information Processing*, pages 416–429, Madrid, 1992.

[16] Franziska Klügl and Ana L C Bazzan. Agent-Based Modeling and Simulation. *AI Magazine*, 33 (3):29–40, 2012.

[17] James N. Macgregor and Yun Chu. Human Performance on the Traveling Salesman and Related Problems : A Review. 3(2):1–29, 2011.

[18] Gergely Neu and Gábor Bartók. An efficient algorithm for learning with semi-bandit feedback. *CoRR*, abs/1305.2732, 2013.

[19] Thanh H Nguyen, Arunesh Sinha, Shahrzad Gholami, Andrew J. Plumptre, Lucas N. Joppa, Milind Tambe, Margaret Driciru, Fred Wanyama, Aggrey Rwetsiba, Rob Critchlow, and Colin M Beale. Capture: A new predictive anti-poaching tool for wildlife protection. *Proceedings of the 15th International Conference on Autonomous Agents and Multiagent Systems (AAMAS)*, pages 767–775, 2016.

[20] Muaz Niazi and Amir Hussain. Agent-based computing from multi-agent systems to agent-based models: A visual survey. *Scientometrics*, 89(2):479–499, 2011.

[21] Noseong Park, Edoardo Serra, Tom Snitch, and V. S. Subrahmanian. APE: A Data-Driven, Behavioral Model-Based Anti-Poaching Engine. *IEEE Transactions on Computational Social Systems*, 2(2):15–37, 2015.

[22] H Van Dyke Parunak, Robert Savit, and Rick L Riolo. Agent-Based Modeling vs . Equation-Based Modeling : A Case Study and Users ' Guide. *Lecture Notes in Computer Science*, 1534:1–16, 1998.

[23] Don Ross. Game theory. In Edward N. Zalta, editor, *The Stanford Encyclopedia of Philosophy*. Metaphysics Research Lab, Stanford University, spring 2019 edition, 2019.

[24] D.P. Williamson and D.B. Shmoys. *The Design of Approximation Algorithms*. Cambridge University Press, 2011.

[25] Haifeng Xu, Long Tran-Thanh, and Nicholas R Jennings. Playing Repeated Security Games with No Prior Knowledge. In *Proceedings of the 15th International Conference on Au- tonomous Agents and Multiagent Systems*, 2016.

[26] Rong Yang, Benjamin Ford, Milind Tambe, and Andrew Lemieux. Adaptive Resource Allocation for Wildlife Protection against Illegal Poachers. In *Proceedings of the 13th International Conference on Autonomous Agents and Multiagent Systems (AAMAS)*, pages 453–460, 2014.