Sensors **2015**, *15*, 1365-1388; doi:10.3390/s150101365

OPEN ACCESS
sensors
ISSN 1424-8220
www.mdpi.com/journal/sensors

Article

Cooperative Surveillance and Pursuit Using Unmanned Aerial Vehicles and Unattended Ground Sensors

Jonathan Las Fargeas *, Pierre Kabamba and Anouck Girard

Department of Aerospace Engineering, University of Michigan, Ann Arbor, MI 48105, USA;
E-Mails: daninman@umich.edu (P.K.); anouck@umich.edu (A.G.)

* Author to whom correspondence should be addressed; E-Mail: jfargeas@umich.edu;
Tel.: +1-734-763-1305; Fax: +1-734-763-0578.

Academic Editor: Felipe Gonzalez Toro

Received: 17 October 2014 / Accepted: 4 January 2015 / Published: 13 January 2015

Abstract: This paper considers the problem of path planning for a team of unmanned aerial vehicles performing surveillance near a friendly base. The unmanned aerial vehicles do not possess sensors with automated target recognition capability and, thus, rely on communicating with unattended ground sensors placed on roads to detect and image potential intruders. The problem is motivated by persistent intelligence, surveillance, reconnaissance and base defense missions. The problem is formulated and shown to be intractable. A heuristic algorithm to coordinate the unmanned aerial vehicles during surveillance and pursuit is presented. Revisit deadlines are used to schedule the vehicles' paths nominally. The algorithm uses detections from the sensors to predict intruders' locations and selects the vehicles' paths by minimizing a linear combination of missed deadlines and the probability of not intercepting intruders. An analysis of the algorithm's completeness and complexity is then provided. The effectiveness of the heuristic is illustrated through simulations in a variety of scenarios.

Keywords: sensor networks; target tracking; path planning for multiple UAVs

1. Introduction

A team of small unmanned aerial vehicles (UAVs) is tasked with patrolling a network of roads near a friendly base. Ground intruders (e.g., trucks) use the road network to reach the base and do not know of the presence of the UAVs. The UAVs patrol the roads to detect and take pictures of any intruders present on the roads.

However, small UAVs possess limited onboard processing resources, and the current detection capability of small aircraft using electro-optical or infrared sensors is not sufficient to ascertain whether an intruder is present or not [1]; thus, the UAVs in this problem are assumed to not possess automated target recognition capabilities. Instead, intruder detection is performed by unattended ground sensors (UGSs) placed on the roads. These sensors measure a given property (e.g., weight of a vehicle driving by), perform classification on the measurement to decide whether it corresponds to an intruder or not and register the time of the detection if the measurement was classified as an intruder. The use of UGSs in conjunction with UAVs enables the pursuit of an intruder along the road network, which is otherwise not possible solely using UAVs.

To maximize the coverage of the road network, the UGSs are placed far apart. The UGSs possess short-range communication devices, but have limited long-range communication capabilities; they require a line of sight to a dedicated communication device with a permanent power source. Placing communication devices to meet these requirements can be difficult depending on the terrain and conditions (e.g., contested environment). However, this problem can be circumvented by using mobile UAVs instead of immobile communication devices. The UAVs do not require the difficult placement of communication devices and power sources in the area, such that the line of sight is maintained between devices; instead, they act as mobile communication devices by querying UGSs directly below using a short-range communication device. Thus, it is assumed that the UGSs cannot communicate with one another, but only with UAVs directly overhead. In contrast, the UAVs are capable of communicating between one another and a central authority (e.g., the base) via a low bandwidth, long-range communication link. The low bandwidth link allows for the transmission of small amounts of information, such as UGS detections or waypoints, but prohibits the transmission of large datasets, such as images. The link is assumed to cover the entire operating area. The short-range and long-range communication devices are assumed to transmit in the electromagnetic spectrum, and as such, communication times are small compared to the other time constants involved in the problem.

This work is motivated by base defense scenarios within the Talisman Saber biennial U.S./Australian military exercise [2], where UAVs are tasked with obtaining intelligence (e.g., location and imagery) about intruders. In these base defense scenarios, the UAVs have limited onboard processing capabilities and, thus, cannot autonomously detect intruders; the UAVs thus rely on UGSs for intruder detection, pursuit and interception [3].

Figure 1 shows a visualization of such a scenario with two UAVs and a single intruder attempting to reach the base. The UAV on the bottom of the figure is communicating with a UGS directly below. Through the communication, the UAV learns of a recent detection from the UGS. The detection is shared with the central authority and used to predict possible future locations of the intruder. The central authority then selects destinations for the UAVs where they are likely to image the intruder.

A mission designer is assumed to have set the UGSs locations and patrol parameters before the mission begins; optimal UGSs placement and patrol parameter selection for this base defense scenario is treated in [4].

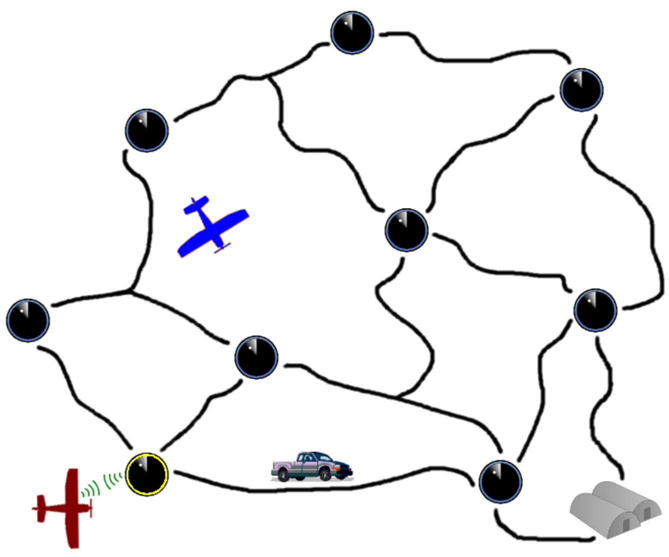

Figure 1. Cooperative surveillance and pursuit scenario.

The UAVs are forced to revisit UGSs to maintain up-to-date and accurate information on the intruder; this is enforced through the use of revisit deadlines, *i.e.*, the maximum time that can elapse between two consecutive visits to a UGS. Forced revisits also help mitigate the impact of false alarms by UGSs. The UAVs cannot detect a target autonomously; hence, they loiter above a UGS and capture an image when the UGS below detects an intruder passing.

The objective is to generate paths for the UAVs (*i.e.*, select waypoints in real time) that satisfy the revisit constraints of UGSs and capture images of intruders before the latter reach the base (*i.e.*, maximize the likelihood of a UAV and intruder being at the same location). A heuristic is provided to compute such paths, and its completeness and complexity are assessed. While the heuristic directs the paths taken by the UAVs, it also indirectly interacts with the UGSs used for monitoring the roads. The heuristic selects when information from the UGSs is obtained, and the heuristic uses the information acquired from the UGSs to predict possible intruder locations in the future. In addition, the capture of an image of the intruder is achieved when a UGS that a UAV is loitering (as directed by the heuristic) detects an intruder.

1.1. Literature Review

The UAVs' task consists of patrolling the road network and pursuing the intruders. A broad amount of research exists on patrolling problems and pursuit-evasion games; as such, literature pertaining to these topics is reviewed.

The problem of persistently monitoring a given area with one or more vehicles has been studied extensively. The problem of finding cycles, such that all points in the patrol area are covered by the vehicle's sensor footprint, is treated in [5]. In [6,7], the authors investigate a patrolling problem for multiple UAVs and maximize the minimum frequency of visitation for different partitions in the area. In [8,9], the authors find paths with a minimized uniform frequency of visitation for all partitions. In [10], the authors investigate the problem of finding paths for mobile agents that minimize the accumulation of an uncertainty metric in the mission area. In [11,12], the authors investigate heuristics, such that the time between visits to the same object of interest is minimized. The problem of patrolling multiple targets cooperatively is also treated in [13], where the authors provide algorithms to compute trajectories for vehicles that minimize the weighted refresh time of the targets being visited. The dual objective of satisfying patrolling constraints (revisit deadlines) and intercepting the intruder separates the problem studied in the current paper from many patrolling problems investigated in the literature.

This problem also shares similarities to the traveling salesman problem (TSP), where an agent is tasked with visiting a certain number of locations while minimizing the distance traveled. TSP is a non-deterministic polynomial-time (NP)-complete problem, and thus, heuristics are used to find solutions; [14] contains a survey on existing heuristics for TSP. In [15], the authors study stochastic and dynamic variations of TSP, where the target locations are generated randomly; the authors then provide algorithms to compute paths for the vehicles that optimize criteria, such as the length of the path or the time between the generation of the target and its observation by the vehicle. The problem treated in the current paper differs from vehicle routing problems and TSPs due to the added goal of capturing images of mobile intruders.

The problem considered in this paper has several components in common with pursuit-evasion games, where defenders are to capture an intruder. Pursuit-evasion games often occur on graphs; the defenders win the game if the intruder is caught; otherwise, the intruder wins. The defenders and intruder move in turns and can only travel to adjacent nodes. The pursuit-evasion problem was studied, and conditions on the number of defenders necessary to guarantee capture were derived in [16–18]. Many variations of the problem exist, such as the addition of constraints on the topology of the graph, the velocities of the defenders or intruders and the amount of information held by one team about the graph or the other team; a survey of pursuit-evasion research relevant to mobile robotics is presented in [19]. Security games introduce targets that the defenders must protect from intruder attacks; in addition, the intruder can observe the defenders. In [20,21], the authors study and provide algorithms to solve security games where the intruder is attempting to infiltrate nodes in a graph, and defenders must visit the various nodes in the graph, such that the intruder never has enough time to infiltrate. The current paper differs from the pursuit-evasion literature, because the defenders and intruders do not move in turns.

In [22], the problem of monitoring an area with intruders using UGSs and UAVs is discussed, and approaches to finding paths for the UAVs that maximize the number of interceptions are presented. The related problem of designing a sensor network for surveillance of a moving target has been studied in [23], where the authors provide methods for multi-point surveillance and demonstrate their effectiveness with regards to tracking probability. The problem of estimating a vehicle's position in a graph given its velocity distribution and a previous detection at a known location and time is treated in [24]. The authors use a histogram filter to predict the vehicle's potential locations in a graph. The approach used in this paper coordinates multiple UAVs (using a different method for prediction) to intercept an intruder on a graph in a centralized fashion while handling uncertainty in the target motion.

While the current literature examines many variations of patrolling problems and pursuit problems, no method treats a framework where the mobile agents fully rely on static sensors placed on an arbitrary road network to track and intercept intruders and provides a path planning algorithm for the mobile agents. The current paper concentrates on this subject. Addressing the reliance on static sensors is important, since the proper tracking of a target by a pursuer cannot be guaranteed in a large number of scenarios and, in some cases, may not be possible at all (as was indicated earlier with small UAVs currently being incapable of autonomous target recognition). In addition, not making assumptions about the topology of the network enables the handling of cases where the environment does not permit favorable sensor placement.

1.2. Original Contributions

The original contributions of this work are as follows: the problem of cooperative surveillance and pursuit is formulated and shown to be NP-hard; a heuristic to solve the problem is given; and an analysis of the heuristic's completeness and complexity is provided.

In previous work, the persistent visitation problem was introduced in which a single vehicle persistently visits a set of nodes, each with a revisit deadline [25]. The goal was to find a path for the vehicle such that no revisit deadline is missed. The existence of periodic solutions was proven; the complexity class of the problem was derived; and heuristics that could solve the problem were presented and characterized.

A version of this problem that included fuel constraints and refueling costs was also studied, and an algorithm that found the minimal cost path satisfying the revisit deadlines and fuel constraints was provided [26].

The current paper differs from the authors' previous work, which focused solely on patrolling, in the following ways: multiple vehicles are considered; an adversary is present in the area; sensors that provide information from which decisions need to be made are included; and the vehicles pursue the adversary while patrolling a number of locations with revisit deadlines.

1.3. Paper Outline

The remainder of the paper is as follows. In Section 2, the model for the defenders and intruders is presented, including the kinematics of the mobile agents and the properties of the UGS. The mathematics for the defenders' pursuit of the intruder are also presented in Section 2. The problem is formulated in Section 3. The heuristic that generates the defenders' actions to solve the problem is presented in Section 4. Results from simulations are shown in Section 5 and discussed in Section 6. Conclusions and future work are discussed in Section 7.

2. Modeling

In this section the model for the defenders (UAVs and UGSs) is presented, followed by a description of the intruder model.

2.1. Defenders

There are n UGSs placed in a planar area along a road network with Cartesian coordinates (ξ_i, ζ_i), $1 \leq i \leq n$. The UGSs and the roads that the intruder can use to travel between them are modeled as a graph $G(N, E)$, where $e_{i,j} \in E$, if there exists a road connecting UGSs i and j. Let $d_{i,j}$ be the length of road $e_{i,j}$. The road network and UGSs' placement is assumed to remain the same over the duration of the mission. The UGSs can measure a continuous property in their proximity and make a classification decision based on the measurement; this classification results in a detection if the measurement is above a certain threshold, e.g., in [22], the UGSs use a small Doppler radar to detect intruders nearby. The UGSs are not equipped with perfect sensors or classifiers and, thus, may emit false alarms.

In addition, m UAVs are patrolling the area each with constant velocity v, finite fuel capacity F and fuel consumption rate $\dot{f}_c$. There is a single base that is capable of refueling the UAVs located at (ξ_{n+1}, ζ_{n+1}). The UGS are assumed to be distant from one another; thus, the UAVs' limited turn radius is not taken into account, and the UAVs are modeled as point masses moving in straight lines [27]. If straight line travel cannot be assumed, then curvature must be accounted for; this can be done using the results from [28], where the authors describe methods to convert paths on a graph with straight line edges to paths on a graph with edges that satisfy the turning constraints of the UAVs.

The UAVs are equipped with a long-range communication device, which enables communication with a central authority, and a short-range communication device, which enables the UAVs to query the status of a UGS directly below (in [22], short-range communication is performed via WiFi radios). The UAVs, UGSs and central authority are assumed to possess synchronized clocks; this can be achieved by calibration before the mission and periodic UGS clock synchronization during UAV visits (it can also be accomplished by equipping each UAV and each UGS with a clock synchronization device, such as a GPS).

Once a UAV obtains information from a UGS, it is immediately relayed to the central authority. The central authority decides which nodes the UAVs are to visit next once their respective destinations are reached. There is no benefit from allowing multiple UAVs to visit the same UGSs simultaneously, since a detection is shared immediately and only a single UAV is required to capture an image; hence, a UGS can only be visited by a single UAV at a time.

2.1.1. Revisit Deadlines

Each UGS has a revisit deadline, $r_i > 0, 1 \leq i \leq n$, set by the mission designer, which under ideal conditions, is the maximum time that can elapse between two visits to the UGS by the defenders. The revisit deadlines are used as a method to keep the defenders' knowledge of intrusions up-to-date. The revisit deadlines are also used to indicate the relative importance of the various UGSs, such that the UAVs can prioritize their actions accordingly.

2.1.2. State Space Model

UGS queries and intruder interceptions only occur above UGSs, and as such, the arrival of a UAV at a UGS is the event that advances the system for the defenders. Thus, a number of states evolve in discrete time corresponding to the arrival of a UAV at a UGS, while other states (such as the amount of fuel onboard a UAV) evolve in continuous time.

The increments (or steps) in discrete time are delimited by the arrivals of UAVs at UGSs. $\tau(k) \in \mathbb{R}$ is the total time elapsed since the beginning of the mission upon completion of step k. $\Omega(k)$ is the set of upcoming UAV arrival times at step k. The current destination of UAV j at step k is indicated by $p_j(k) \in 1, 2, ..., (n+1), 1 \leq j \leq m$. The Cartesian coordinates of UAV j at step k are $(\xi_j{}'(k), \zeta_j{}'(k)) \in \mathbb{R}^2, 1 \leq j \leq m$. Let $y(k)$ denote the discrete time states at step k, *i.e.*, $y(k) = [\, \tau(k)\ \Omega(k)\ p_j(k)\ (\xi_j(k), \zeta_j(k))\,]$.

The continuous time states are linked to the discrete time states by applying impulses in their dynamics if certain conditions are met when a step is completed. The amount of fuel UAV j is carrying at time t is indicated by $f_j(t) \in \mathbb{R}, 1 \leq j \leq m$. Let $x_i(t) \in \mathbb{R}, 1 \leq i \leq n$ be the slack time of UGS i at time t, which indicates how much time remains before a visit to UGS i is overdue. Let the input $u_j(k)$ be the destination of UAV j at step k.

2.1.3. Initial Conditions

The initial time is set to zero. The initial set of arrival times is initialized to the empty set. Without loss of generality, the UAVs are assumed to start at the base with full fuel capacity, and the slack time for each UGS is initialized to its respective revisit deadline, *i.e.*,

$$\begin{aligned}
\tau(0) &= 0, \\
\Omega(0) &= \emptyset, \\
p_1(0) &= n+1, \\
(\xi_1{}'(0), \zeta_1{}'(0)) &= (\xi_{n+1}, \zeta_{n+1}), \\
p_2(0) &= n+1, \\
(\xi_2{}'(0), \zeta_2{}'(0)) &= (\xi_{n+1}, \zeta_{n+1}), \\
&\vdots \\
p_m(0) &= n+1, \\
(\xi_m{}'(0), \zeta_m{}'(0)) &= (\xi_{n+1}, \zeta_{n+1}), \\
f_1(0) &= F, \\
f_2(0) &= F, \\
&\vdots \\
f_m(0) &= F, \\
x_1(0) &= r_1, \\
x_2(0) &= r_2, \\
&\vdots \\
x_n(0) &= r_n
\end{aligned} \tag{1}$$

2.1.4. Dynamics

Let $q_j(i,k)$ be the Euclidean distance between UAV j and UGS i at step k. When a UAV arrives at its destination, its next destination is set by the input, and the travel time to that destination is added to the set of arrival times:

$$\begin{aligned} &\text{if } q_j(p_j(k),k)=0, \begin{cases} p_j(k+1) & := u_j(k) \\ \Omega(k) & := \{\frac{q_j(u_j(k),k)}{v}\} \cup \Omega(k) \end{cases} \\ &\text{else}, \qquad\qquad\quad p_j(k+1) \;\; := p_j(k) \end{aligned} \tag{2}$$

The time of the next step is set by the minimum UAV arrival time in $\Omega(k)$; that time is then removed from $\Omega(k)$:

$$\tau(k+1) := minimum(\Omega(k)) \tag{3}$$

$$\Omega(k+1) := \Omega(k) \setminus \{\tau(k+1)\} \tag{4}$$

The amount of fuel onboard a UAV decreases as dictated by the fuel consumption rate and is reset to full capacity whenever a visit to the base occurs:

$$\begin{aligned} \dot{f}_j(t) = -\dot{f}_c + \sum_{l=1}^{k-1} \dot{f}_c \times (1 - \delta_{p_j(l)p_j(l+1)} \times \delta_{(n+1)p_j(l)}) \times H(t-\tau(l)) \times H(\tau(l+1)-t) \\ + \sum_{l=1}^{k} \delta_{(n+1)p_j(l)} \times (F - f_j(\tau(l))) \times \delta(t-\tau(l)),\ t \leq \tau(k)\ ,\ 1 \leq j \leq m \end{aligned} \tag{5}$$

where δ_{ij} is the discrete Kronecker delta function, $H(t-\tau(l))$ is the Heaviside step function, and $\delta(t-\tau(j))$ is the continuous Dirac delta function. The slack time of a given UGS decreases linearly in time and is reset to its revisit deadline whenever it is visited by a UAV:

$$\begin{aligned} \dot{x}_i(t) = -1 + \sum_{l=1}^{k-1} \sum_{j=1}^{m} (1 - \delta_{p_j(l)p_j(l+1)} \times \delta_{ip_j(l)}) \times H(t-\tau(l)) \times H(\tau(l+1)-t) \\ + \sum_{l=1}^{k} \sum_{j=1}^{m} \delta_{ip_j(l)} \times (r_i - x_i(\tau(l))) \times \delta(t-\tau(l)),\ t \leq \tau(k)\ ,\ 1 \leq j \leq m \end{aligned} \tag{6}$$

2.2. Intruder

A single intruder is traveling on the road network at a time; but there may be multiple intruders over the course of the mission. The intruders move inside their adversary's territory and suspect that they are under observation. Thus, the intruders move stochastically to reduce the predictability of their actions. However, the intruders do not know how they are being observed and cannot perceive the UAVs flying above [3]. If the intruders can detect the UAVs in the area, then a different intruder model is needed, e.g., using results from the pursuit-evasion literature or Stackelberg games.

The intruder moves from one node to the next in the graph according to a first order Markov process, *i.e.*, the next location that it visits only depends on its current location. The central authority is assumed to possess the Markov model for the intruder's movement; this model could have been obtained through intelligence or deduced from prior observation. The intruder enters the graph at a random node according to the initial distribution of the Markov model. Since the intruder's movement is memoryless, the UAVs and central authority only use the most recent intruder detection for their computations and do not keep track of the trail of UGSs detections left by the intruder.

The distance traveled by the intruder at time t is modeled as the process X_t:

$$\begin{aligned} X_0 &= 0, \\ X_t - X_\rho &= \mathcal{N}(\mu_v \times (t-\rho), \sigma_v^2 \times (t-\rho)^2), t > \rho > 0 \end{aligned} \tag{7}$$

where $\mu_v > 0, \sigma_v > 0$. The fuel consumption of the intruder's vehicle is assumed to be negligible.

2.3. Intruder Interception

To direct the UAVs, such that interception is likely to occur, the probability of the intruder passing by a UGS that a UAV is loitering above must be calculated.

2.3.1. Probability of Intruder Passing a Node in a Given Time Window

Given a path s of UGSs, where $s(i)$ indicates the $i-th$ UGS visited and the fact that the intruder passed $s(1)$ at time zero, the probability of the intruder passing $s(2)$ between times t_i and t_f (where $t_i > 0$ and $t_f > t_i$) is:

$$P_{int}(s(2), s, t_i, t_f) = \int_{t_i}^{t_f} P[X_\rho = d_{s(1),s(2)}]d\rho \tag{8}$$

The probability of the intruder passing $s(3)$ between times t_i and t_f is:

$$P_{int}(s(3), s, t_i, t_f) = \int_{t_i}^{t_f} P[X_\rho = d_{s(1),s(2)} + d_{s(2),s(3)}]d\rho \tag{9}$$

The probability of the intruder passing $s(b+1)$ between times t_i and t_f is:

$$P_{int}(s(b+1), s, t_i, t_f) = \int_{t_i}^{t_f} P[X_\rho = \sum_{f=1}^{b} d_{s(f),s(f+1)}]d\rho \tag{10}$$

Generalizing, the probability of the intruder passing node j (while traveling along the path s) between times t_i and t_f is:

$$P_{int}(j, s, t_i, t_f) = \sum_{b=1}^{|s|-1} \delta_{s(b+1),j} \int_{t_i}^{t_f} P[X_\rho = \sum_{f=1}^{b} d_{s(f),s(f+1)}]d\rho \tag{11}$$

The probability that the intruder will pass node j between times t_i and t_f before reaching the base is:

$$P_{int}(j, s, t_i, t_f) = \sum_{b=1}^{|s|-1} (1 - \mathbf{1}_{s(1:b)}(n+1))\delta_{s(b+1),j} \int_{t_i}^{t_f} P[X_\rho = \sum_{f=1}^{b} d_{s(f),s(f+1)}]d\rho \tag{12}$$

where $\mathbf{1}_{s(1:b)}(n+1)$ is the indicator function.

A set of paths S is introduced, where $S(a)$ indicates the ath path within the set and $S(a,f)$ indicates the fth node visited in the ath path within the set. $P(S(a))$ is the probability of the ath path occurring in the set where $\sum_{a=1}^{|S|} P(S(a)) = 1$; $P(S(a))$ is computed using the Markov model for the intruder's motion along the road network. Thus, the probability that the intruder will pass node j between times t_i and t_f before reaching the base given a set of paths S is:

$$P_{int}(j,S,t_i,t_f) = \sum_{a=1}^{|S|} P(S(a)) \sum_{b=1}^{|S(a)|-1} (1-\mathbf{1}_{S(a,1:b)}(n+1))\delta_{S(a,b+1),j} \int_{t_i}^{t_f} P[X_\rho = \sum_{f=1}^{b} d_{S(a,f),S(a,f+1)}]d\rho \tag{13}$$

2.3.2. Probability of Intruder Interception

Let $g_i(j,k)$ be the Euclidean distance between the positions of UAV i at steps j and k. Given the probability of an intruder passing a UGS during a certain time window, the probability of intruder interception can be calculated by accounting for the UAV locations:

$$P_{capture}(S,y(l),y(l+1)) = \sum_{j=1}^{m} \alpha_j \times P_{int}(p_j(l+1),S,\tau(l),\tau(l+1)) \tag{14}$$

$$\text{where: } \alpha_j = \begin{cases} 1 & \text{if } g_j(l,l+1)\text{=0} \\ 0 & \text{otherwise} \end{cases} .$$

This equation consists of a sum over all of the UAVs, where j indicates the UAV index. α_j is only one when UAV j is loitering over a UGS during the step, in which case, the probability of an intruder passing the node where the UAV is located during the time window of the step is added.

3. Problem Formulation

Using the models for the defenders and intruders presented in the previous section, the problem is now formulated. Given the UAV states' at the current step, the UGSs' revisit deadlines and the results from the UGSs' queries, the UAVs are to find paths that satisfy the revisit deadlines and maximize the probability of intercepting the intruder. This revisit deadline satisfaction version of the problem does not allow for missing any revisit deadline and, hence, limits the UAVs' ability to pursue the intruder.

Thus, a revisit deadline optimization version problem is formulated to give the UAVs flexibility in meeting revisit deadlines and pursuing the intruder. Instead of satisfying the revisit deadlines, the amount by which revisit deadlines are missed is minimized. Let $t_{mission}$ be the duration of the UAVs' base defense mission. Let S_l be the set of possible intruder paths given the most recent intruder detection at step l. Let $\lfloor h \rfloor(u,t)$ indicate the smallest step in sequence u of UAV actions where $\tau(\lfloor h \rfloor(u,t)) \geq t$. Let $\tilde{x}_i(t)$ and $\underset{\sim}{x}_i(t)$, respectively, be the slack time left and the slack time overdue at time t,

$$\tilde{x}_i(t) = \frac{(|x_i(t)| + x_i(t))}{2} \tag{15}$$

$$\underset{\sim}{x}_i(t) = \frac{(|x_i(t)| - x_i(t))}{2} \tag{16}$$

Let β be the cost of not capturing the intruder, and let γ be the cost associated with missing a deadline, where $\beta > 0$ and $\gamma > 0$. Let $C(l)$ be the cost of the UAV actions taken at step l,

$$\begin{aligned} C(l) =& \beta \times (\tau(l+1) - \tau(l)) \times (1 - P_{capture}(S_l, y(l), y(l+1)) \\ &+ \frac{\gamma}{n} \times \sum_{i=1}^{n} (\tau(l+1) - \tau(l) - \tilde{x}_i(\tau(l))) \times \frac{\underset{\sim}{x}_i(\tau(l+1)) + \underset{\sim}{x}_i(\tau(l))}{2} \end{aligned} \tag{17}$$

The first component of the cost penalizes the UAVs for not intercepting the intruder over the course of the step. The second component of the cost penalizes revisit deadlines that are overdue by adding the integral of the slack time missed over the course of the step.

Based on the cost function above, the revisit deadline optimization version of the problem is formulated as follows: the central authority is to find a sequence $\boldsymbol{u_j(k)},\ \boldsymbol{1 \leq j \leq m, k} \in \mathbb{N}$, such that, under Equations (1)–(6), $\sum_{\boldsymbol{k=1}}^{\lfloor \boldsymbol{h} \rfloor (\boldsymbol{u}, \boldsymbol{t_{mission}})} \boldsymbol{C(k)}$ is minimized and $\boldsymbol{0 \leq t \leq t_{mission}, 1 \leq j \leq m,\ f_j(t) \geq 0}$.

3.1. Problem Complexity

Proposition 1. *The revisit deadline satisfaction version problem of cooperative surveillance and pursuit is NP-hard.*

Proof. If no intruders are present in the road network, then an optimal solution is one where the slack times are kept positive. The problem of keeping slack times positive for a single UAV is the persistent visitation problem and is proven to be NP-complete in [25]. The persistent visitation problem can be reduced to the problem of cooperative surveillance and pursuit by selecting one UAV to patrol the same graph with the same revisit deadlines without intruders present. Thus, the problem of cooperative surveillance and pursuit is NP-hard. □

Corollary 2. *The revisit deadline optimization version problem of cooperative surveillance and pursuit is NP-hard.*

4. UAV Path Selection

The problem is NP-hard, hence a heuristic algorithm is used to select the paths of the UAVs. This algorithm searches ahead for a sequence of UAV actions that minimizes the cost function within a certain time window and executes the first set of UAV actions in the sequence.

4.1. System Structure

The algorithm used to simulate the defenders' system for a cooperative surveillance and pursuit problem is provided in Algorithm 1. The states are first initialized (Lines 1–3); then at each step, the possible intruder paths within t_{search} are computed (Line 5), followed by the computation of the minimal cost action for the UAVs within the same time horizon using the possible intruder paths (Line 6). In addition, at each step k, new detections are added from the UGS queries (Lines 9–11), stale queries rejected (Lines 12–16) and UGS queries without detections added (Lines 17–18). A UGS query is characterized by the status of the UGS, the time of the detection if one occurred (otherwise, the time of the query) and the index of the queried UGS. The resulting data from querying the UGSs is used in the intruder path generation process in the next time step. This process of finding intruder paths, selecting UAV actions and gathering UGSs queries is repeated until the mission completion time is reached (Line 4).

Algorithm 1: Defenders' system for the cooperative surveillance and pursuit problem.

```
Data: $G(N,E)$,$r$,$d$,$m$,$v$,$F$,$\dot{f}_c$,$t_{mission}$,$t_{search}$,$t_{stale}$
1  $k \leftarrow 0; t \leftarrow 0$
2  $(y(k), f(t), x(t)) \leftarrow$ (Equation (1))
3  $t_D \leftarrow \emptyset; n_D \leftarrow \emptyset; T_U \leftarrow \emptyset; N_U \leftarrow \emptyset$
4  while $\tau(k) < t_{mission}$ do
5      $(S_k, P(S_k)) \leftarrow paths(n_D, 1, t_D, t_D, T_U, N_U, \tau(k) + t_{search}, \emptyset, \emptyset)$
6      $u(k) \leftarrow uavsAction(y(k), f(\cdot), x(\cdot), t_{search}, S_k, P(S_k))$
7      $(y(k+1), f(\cdot), x(\cdot)) \leftarrow$ (Equations (2) – (6)), $y(k), u(k)$
8      $k \leftarrow k+1; t \leftarrow \tau(k+1)$
9      for (detection) $\in$ queries(k) do
10         if $(detection).t > t_D$ then
11             $(t_D, n_D) \leftarrow (detection)$
12     if $t_D + t_{stale} < \tau(k)$ then
13         $t_D \leftarrow \emptyset; n_D \leftarrow \emptyset; T_U \leftarrow \emptyset; N_U \leftarrow \emptyset$
14         for $(t_U, n_U) \in (T_U, N_U)$ do
15             if $t_U \leq (\tau(k) - t_{stale})$ then
16                 $(T_U, N_U) \leftarrow (T_U, N_U) \setminus (t_U, n_U)$
17     for ($\neg$detection) $\in$ queries(k) do
18         $(T_U, N_U) \leftarrow (T_U, N_U) \cup (\neg detection)$
Result: $u_j(\cdot)$
```

4.2. Intruder Path Generation

Possible intruder paths are generated using the information from UGS queries. The central authority stores the time and UGS index of the most recent intruder detection. It also stores the times and UGS indices of recent UGS queries without detections. This information is used to generate the potential intruder paths at each step using a recursive breadth first search methodology. If a detection is too stale, then it is ignored. In simulations, the threshold for a detection to be considered stale was selected to be half the mean intruder travel time between the two UGSs most distant from one another.

A recursive algorithm is used to compute the possible intruder paths (Algorithm 2); this algorithm is used by the heuristic to assist in its decision making process. This algorithm to find intruder paths is the method by which the heuristic uses the information obtained from the UGSs. The input of the algorithm is the current candidate path for the intruder (which at the start of the algorithm's execution, is the most recent detection). At each iteration in the search, the nodes adjacent to the last node of the current working path, s, are obtained (Line 1). Possible travel times to these adjacent nodes are then computed using the intruder's velocity probability distribution. If any of the visits to the adjacent nodes violate the constraints set by recent UGS queries without detections, then they are removed (Lines 2–10). Valid adjacent nodes are then appended to the current working path (Lines 11–12). If the minimum travel time of the new path is larger than the search depth, then the path is admitted to the set of possible paths (Lines 14–15); otherwise, the search continues for that path (Lines 16–17). When all of the candidate paths reach the search depth, the algorithm terminates and returns the possible intruder paths. These possible intruder paths are then used to select the actions of the UAVs.

Algorithm 2: The paths()algorithm to find possible intruder paths.

```
Data: $s$,$p$,$t_{min}$,$t_{max}$,$T_U$,$N_U$,$t_f$,$S$,$P(S)$
1  $\Gamma \leftarrow adjacentNodes(s(|s|), G)); l \leftarrow |\Gamma|$
2  while $l > 0$ do
3      $T'_{max}(l) \leftarrow t_{max} + \frac{d_{s(|s|),\Gamma(l)}}{min(v_{int})}$
4      for $(t_U, n_U) \in (T_U, N_U)$ do
5          if $\Gamma(l) = n_U$ then
6              if $T'_{max}(l) \leq t_U$ then
7                  $\Gamma \leftarrow \Gamma \setminus \{\Gamma(l)\}$
8                  $T'_{max} \leftarrow T'_{max} \setminus \{T'_{max}(l)\}$
9              break
10     $l \leftarrow (l - 1)$
11 for $l \leftarrow 1$ to $|\Gamma|$ do
12     $s' \leftarrow [s\ \Gamma(l)]; p' \leftarrow p \times \frac{1}{|\Gamma|}$
13     $t'_{min} \leftarrow t_{min} + \frac{d_{s(|s|),\Gamma(l)}}{max(v_{int})}$
14     if $t'_{min} \geq t_f$ then
15         $S \leftarrow S \cup \{s'\}; P(S = s') \leftarrow p'$
16     else
17         $(S, P(S)) \leftarrow paths(s', p', t'_{min}, T'_{max}(l), T_U, N_U, t_f, S, P(S))$
Result: $S, P(S)$
```

4.3. Selection of UAV Actions

The following algorithm is used by the heuristic to make its decisions. The algorithm starts by searching for all possible sequences of actions for the team of UAVs within the search horizon t_{search}.

For each sequence of actions available to the team of UAVs, $\tilde{u}$, it then assesses the cost using the following equation:

$$\sum_{l=k}^{\lfloor h \rfloor(\tilde{u}, t_{search})} C(l) \tag{18}$$

where k is the current step. The cost calculations use the intruder paths generated earlier.

The algorithm starts by using the current state of the UAVs to compute feasible actions. The action is then applied, and the corresponding UAV states and costs are computed. These actions, states and costs are then added to sets of candidate sequences of actions, candidate states of the UAVs after the application of the corresponding sequence of actions and candidate costs after the application of the corresponding sequence of actions. The algorithm then proceeds to iterate over the set of candidate sequences of actions for the UAVs.

The procedure for computing potential sequences of actions and their costs is described in Algorithm 3. Several intermediate variables are used: $\tilde{U}$ is the working set of sequences of UAV actions; W is the corresponding set of states after the sequences of actions in $\tilde{U}$ have been applied; and Θ is the set of costs for the sequences of actions. u is the current minimal cost sequence of actions, and ϕ is the current minimal cost. These variables are initialized (Lines 1–2). The algorithm then proceeds to loop over the working set of sequences of UAV actions until none remain (Line 3). At each iteration, the first elements in the working sets of states, sequences of actions and costs are obtained and removed from their parent sets (Lines 4–5). The set of possible actions, Z, given the current working state $w \in W$ is then computed (Lines 6–14). For each possible action for the team of UAVs, $z \in Z$, the next state, λ, is computed (Line 15). If λ results in positive fuel for all of the UAVs and has reached the search depth, then its cost is computed (Lines 16–17); if that cost is smaller than the current minimal cost, then the current minimal cost sequence of actions is set to the sequence of actions that resulted in λ (Lines 18–19). If λ results in positive fuel for all of the UAVs without reaching the search depth, then the working state, sequence of actions and cost are added to the corresponding parent working sets (Lines 20–24). The algorithm terminates when the set of candidate sequences of UAV actions is empty (*i.e.*, all feasible actions in the search horizon have been considered) and returns the minimal cost sequence of actions. The heuristic directs the UAVs to take the first step in this minimal cost sequence of actions; this step results in the UAVs visiting or loitering above certain UGSs, thereby obtaining new information from the UGSs and possibly capturing an image of the intruder. The possibility of imaging the intruder exists when a UAV is loitering above a UGS; the capture of an image is triggered when the UGS which the UAV is loitering detects the intruder.

4.4. Algorithm Completeness

The search depth for the UAV actions affects the existence of solutions; if the search depth is less than the endurance of the UAVs, $t_{search} < (F/\dot{f}_c)$, then there is no guarantee that the generated paths will lead to the satisfaction of the UAVs' fuel constraints. If $t_{search} \geq t_{mission}$, then the heuristic is complete, *i.e.*, it will find a solution if one exists.

Algorithm 3: The uavsAction() heuristic to find the minimal cost action for the UAVs within a given search horizon.

```
Data: $y(k)$,$f(\cdot)$,$x(\cdot)$,$S$,$P(S)$
1  $W \leftarrow \{(y(k), f(\cdot), x(\cdot))\}$
2  $\tilde{U} \leftarrow \emptyset; \Theta \leftarrow \emptyset; u \leftarrow \emptyset; \phi \leftarrow \infty$
3  while $|W| > 0$ do
4      $w \leftarrow W(1); \tilde{u} \leftarrow \tilde{U}(1); \theta \leftarrow \Theta(1)$
5      $W \leftarrow W \setminus \{w\}; \tilde{U} \leftarrow \tilde{U} \setminus \{\tilde{u}\}; \Theta \leftarrow \Theta \setminus \{\theta\}$
6      $Z \leftarrow \emptyset$
7      if $q_1(w.p_1) = 0$ then
8          $Z \leftarrow N \cup \{n+1\}$
9      for $l \leftarrow 2$ to $m$ do
10         if $q_l(w.p_l) = 0$ then
11             $Z \leftarrow Z \times \{N \cup \{n+1\}\}$
12         else
13             $Z \leftarrow Z \times \emptyset$
14     for $z \in Z$ do
15         $\lambda \leftarrow$ (Equations (2) $-$ (6))$, w, z$
16         if $(\forall j, \lambda.f_j(\lambda.\tau) \geq 0) \wedge (\lambda.\tau \geq \tau(k) + t_{search})$ then
17             $\theta' \leftarrow \theta +$ (Equation(17))$, S, P(S), w, \lambda$
18             if $\theta' < \phi$ then
19                 $u \leftarrow [\tilde{u}\ z]; \phi \leftarrow \theta'$
20         else if $\forall j, \lambda.f_j(\kappa.\tau) \geq 0$ then
21             $\theta' \leftarrow \theta +$ (Equation (17))$, S, P(S), w, \lambda$
22             if $\theta' < \phi$ then
23                 $W \leftarrow W \cup \{\lambda\}; \tilde{U} \leftarrow \tilde{U} \cup [\tilde{u}\ z]$
24                 $\Theta \leftarrow \Theta \cup \{\theta'\}$
Result: $u$
```

4.5. Algorithm Complexity

The topology of the road network, the number of UAVs, the UAVs' velocity, the mean intruder velocity and t_{search} affect the algorithm complexity. Let $\bar{d}$ be the mean distance between any two UGSs. By inspection, the time complexity of the heuristic per step is:

$$O\left(|E|^{\frac{t_{search} \times \mu_v}{\bar{d}}} + \left(\frac{n!}{m!(n-m)!}\right)^{\frac{t_{search} \times v}{\bar{d}}}\right) \quad (19)$$

The complexity increases polynomially with respect to the number of UGS and the number of roads, increases exponentially with respect to the vehicle velocities and the search depth and decreases exponentially with respect to the mean distance between any two UGSs.

5. Simulations

To illustrate the performance of the $uavsAction()$ heuristic, several simulations with varying configurations are shown. A local search heuristic is used as a baseline comparison to the $uavsAction()$ heuristic developed in the previous sections. The local search heuristic is detailed in Appendix A.

Four scenarios are considered: Scenario A2 occurs in Area A with a UAV-to-intruder velocity ratio of two; Scenario A3 occurs in Area A with a UAV-to-intruder velocity ratio of three; Scenario B1 occurs in Area B with a UAV-to-intruder velocity ratio of one; and Scenario B2 occurs in Area B with a UAV-to-intruder velocity ratio of two. Two UAVs are operating in Area A, while three UAVs are operating in Area B; UAV fuel consumption is not accounted for in these simulations. Visualizations of Areas A and B are shown in Figure 2. Detailed parameters for the scenarios are given in Tables 1 and 2; these tables contain the locations of the base and UGSs, revisit deadlines for the UGSs and velocity for the UAVs and intruders.

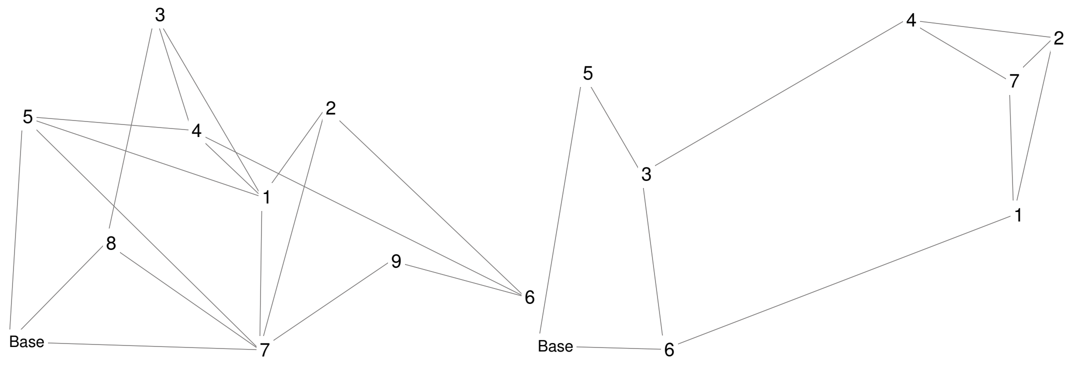

Figure 2. Visualization with base, UGSs and roads of Scenario A (**left**) and B (**right**).

Table 1. Base and unattended ground sensor (UGS) parameters for Scenarios A and B.

	Scenarios A			**Scenarios B**		
Destination	ξ_i	ζ_i	r_i	ξ_i	ζ_i	r_i
Base	6	6	N/A	6	6	N/A
UGS 1	94.66	70.33	11.4	150.03	56.21	8.1
UGS 2	117.05	109.94	12.4	162.18	124.23	10.7
UGS 3	57.17	151.44	10.5	37.41	72.06	3.0
UGS 4	70.00	100.00	2.6	117.54	131.08	8.4
UGS 5	10.79	106.16	10.8	19.65	110.68	9.5
UGS 6	186.80	25.98	8.3	44.29	4.66	7.1
UGS 7	93.88	2.38	5.6	148.70	107.66	12.6
UGS 8	40.00	50.00	7.0	N/A	N/A	N/A
UGS 9	140.00	42.00	11.0	N/A	N/A	N/A

Table 2. Vehicle parameters for Scenarios A and B.

	Scenario A2		**Scenario A3**		**Scenario B1**		**Scenario B2**	
Vehicle	μ_v	σ_v	μ_v	σ_v	μ_v	σ_v	μ_v	σ_v
Intruder	37.5	1.57	25	1.57	25	2.33	25	2.33
UAVs	75	0	75	0	25	0	50	0

The metric used to assess the performance of the approaches is the intruder capture index. The intruder capture index is the number of intruders whose image was captured subtracted by the number of intruders that reached the base divided by the total number of intruders over the course of the mission. Three cost configurations, indicated by (β, γ), are considered: $(1, t_{mission})$, $(t_{mission}, 1)$, $(t^2_{mission}, 1)$, where the mission time, $t_{mission}$, is 30 in the simulations. Missing revisit deadlines is weighted heavily in the first cost configuration, while not capturing an image of the intruder is weighted heavily in the last cost configuration. Eighty simulations were run per search depth per scenario per cost configuration; Figures 3–6 show the average intruder capture index (represented on the ordinate) for these 80 simulations for the three different cost configurations (represented by the three different line styles) as a function of search depth for Scenarios A and B (represented on the abscissa).

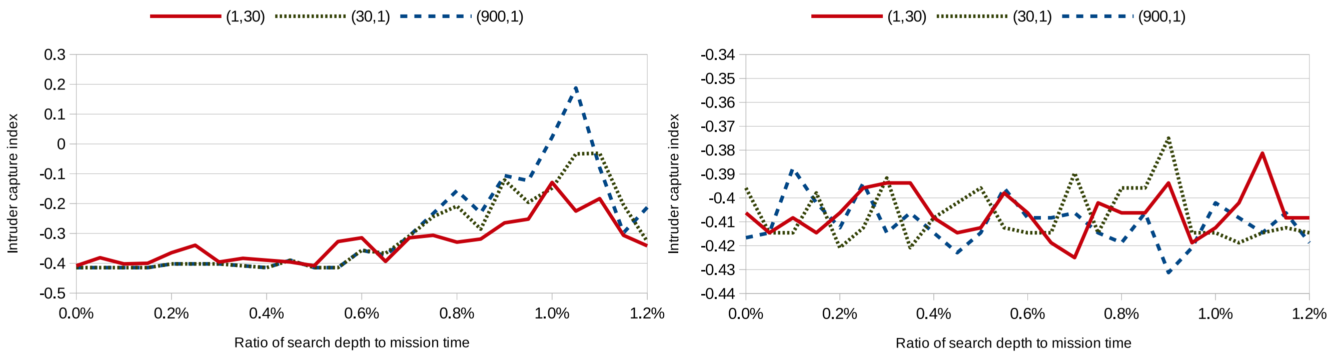

Figure 3. Intruder capture index for Scenario A2 using $uavsAction()$ (**left**) and $localSearch()$ (**right**).

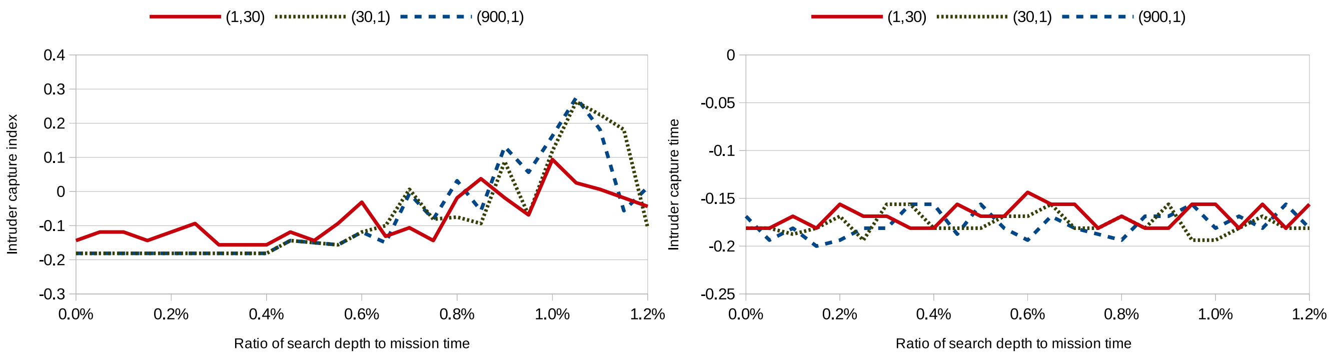

Figure 4. Intruder capture index for Scenario A3 using $uavsAction()$ (**left**) and $localSearch()$ (**right**).

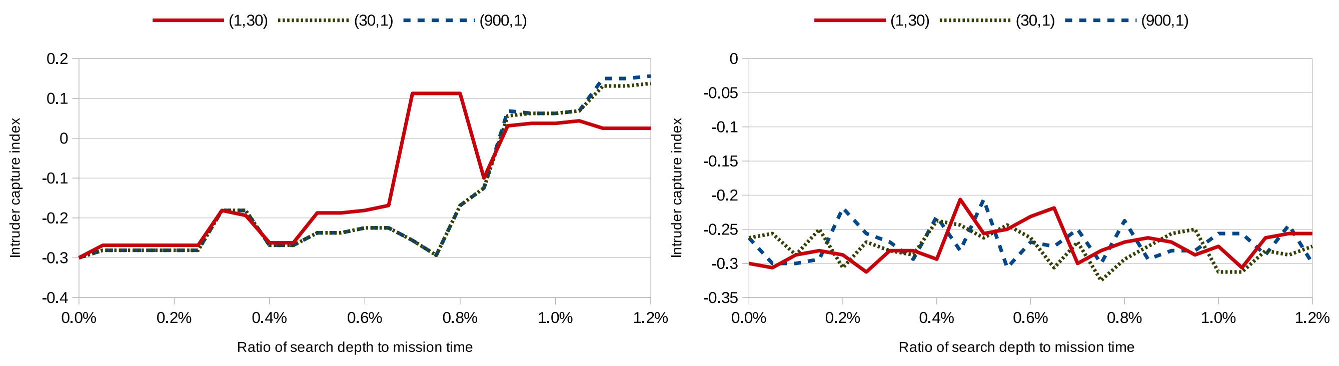

Figure 5. Intruder capture index for Scenario B1 using $uavsAction()$ (**left**) and $localSearch()$ (**right**).

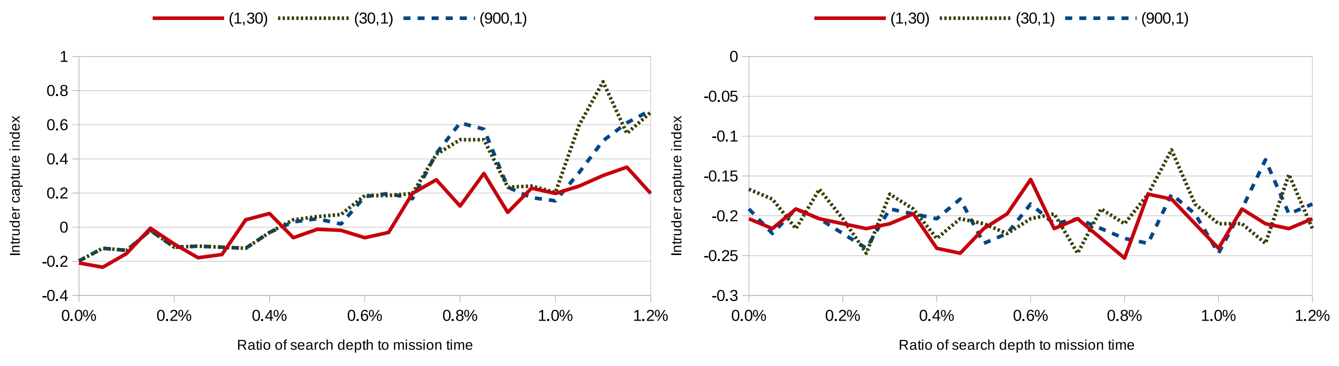

Figure 6. Intruder capture index for Scenario B2 using $uavsAction()$ (**left**) and $localSearch()$ (**right**).

In the left subfigures of Figures 3–6, where $uavsAction()$ is used, the intruder capture index increases as a function of search depth for all cost functions. The intruder capture index for $localSearch()$ (right subfigures of Figures 3–6) does not increase significantly with a larger search depth. While $uavsAction()$ and $localSearch()$ perform similarly for short search depths, $uavsAction()$ performs better than $localSearch()$ for larger search depths.

For Scenarios A2, A3 and B2 (Figures 3, 4 and 6), weighting intruder capture heavily when using $uavsAction()$ resulted in more captures for large search depths, while only a marginal difference between cost configurations is seen for small search depths. In Scenario B1 (Figure 5), weighting missed revisit deadlines more heavily for $uavsAction()$ resulted in more captures for search depths less than 0.9% of the mission completion time. The performance of $localSearch()$ is not significantly affected by different cost configurations.

In Scenario A, an optimal search depth is seen for both velocity advantages at 1.05% of the mission completion time for $uavsAction()$. An optimal search depth is not seen in Scenario B; however, it may occur at a search depth greater than performed in the simulations. As expected, the intruder capture indices in Scenario A3 are larger than in Scenario A2, *i.e.*, the UAVs perform better when they are faster. This trait is also seen when comparing Scenario B2 to Scenario B1. The UAVs do not perform well when they do not have a velocity advantage (Figure 5), even when they are more numerous. When the UAVs are faster than the intruder and the topography is advantageous, as is the case in Scenario B, they can perform very well (Figure 6).

Videos of simulations for a variety of other scenarios can be found at [29].

6. Discussion

The performance of the $uavsAction()$ heuristic largely depends on the problem instance, as can be seen from the differences in the results from the scenarios presented. In the simulations, the UAVs perform better with a larger velocity advantage compared to the intruder; this behavior is expected in all scenarios. Problem instances with bottlenecks in the road network topology lead to better performance, since the UAVs' pursuit and interception of the intruder are simplified. Topologies with highly connected nodes close together lead to poorer performance, since the UAVs cannot adequately pursue the intruder. The effect of the topology on the performance of the heuristic highlights the importance of the selection of UGSs' locations.

The simulations demonstrate that the heuristic can be used to plan the paths of UAVs in cooperative surveillance and pursuit tasks. Cooperative surveillance and pursuit systems, such as the motivating Talisman Saber exercise, are realistic and can be constructed with current technology. Details on a UAV and UGS system to monitor and pursue intruders for the Talisman Saber exercise is provided in [22]. This system uses small, commercially available UAVs and small Doppler radars as UGSs. UAVs and UGSs communicate using WiFi radios and the UAVs capture images of the intruder using a gimballed video camera. The Talisman Saber area is approximately 2450 km^2, and the road network within the area is sparse. The velocity of the UAVs ranges between 50 km/h and 100 km/h, and typical cruising altitude ranges between 60 m and 900 m, which is within the 1-km communication range of the UGSs' WiFi radios. With these parameters, the simulations shown in Section 5 can be viewed as taking place in an area nine-times larger than the Talisman Saber area or with UAVs that travel at a third of the velocity of the commercially available UAVs discussed. Simulating faster UAVs in a smaller area with more UGSs, similar to Talisman Saber, takes much longer to simulate, because of the complexity of the heuristic: in Equation (19), faster UAVs and smaller areas increase the computation time exponentially, and increasing the network size increases the computation time polynomially.

While this work is motivated by the Talisman Saber exercise, the heuristic can be used for other monitoring and pursuit tasks occurring on graphs using vehicles and stationary sensors with the same assumptions and dynamics that are described above; examples of such tasks include a pollutant monitoring task in a body of water using autonomous boats in conjunction with stationary buoys or the monitoring of a forest fire or flood using appropriate sensors on the ground and autonomous aircraft to visit these sensors.

The assumptions made in this work allow for paths to be computed while still capturing the essence of the problem; however, they do not allow for the modeling of certain features that do occur in real scenarios. For example, the UAV constant velocity and constant fuel consumption rate assumption does not handle wind disturbances. Wind disturbances affect the speed of the UAVs and alter the arrival times of UAVs at UGSs. If the wind is steady, the decision making heuristic can account for wind in its computations. If the wind is stochastic, *i.e.*, turbulence, the constant fuel consumption rate assumption needs to be relaxed, so that more fuel can be consumed during turbulence to allow the vehicles to remain on their nominal paths. The assumption that the intruder moves according to a Markov chain may be valid in some scenarios, but does not allow for intelligent intruder behavior, such as reacting to seeing a UAV flying overhead or attempting to deceive the UAVs. The assumptions made for communication reflect the capabilities of the hardware used for the Talisman Saber exercise; however, they do not handle disturbances, such as communication jamming.

7. Conclusions

In this paper, a path planning problem for a team of UAVs patrolling a network of roads and pursuing intruders using UGSs is formulated. The problem is shown to be NP-hard. A heuristic algorithm to solve the problem is presented, and its completeness and complexity are assessed. The heuristic primarily plans the paths for the UAVs; however, it also interacts with the UGSs by acquiring information from the UGSs through the UAVs, using the information to predict possible intruder paths and relying on the UGSs to trigger the capture of an image of the intruder by a loitering UAV.

The heuristic demonstrates that intercepting the intruder is possible using a sensing scheme relying entirely on UGSs. Given a good topology for the road network and well selected revisit deadlines, such as Scenario B in Section 5, the heuristic performs well and can intercept a majority of the intruders. The heuristic also exhibits intuitive behavior at times, such as orbiting several UGSs nearby and trapping the intruder, thereby forcing an interception.

In future work, other heuristics to solve the problem and decentralized approaches will be investigated. The ability to handle multiple intruders simultaneously will also be included. This could be achieved by the UGSs tagging their measurements to certain intruders. Variable levels of communication between the UGSs and UAVs will also be considered.

Acknowledgments

The authors would like to thank Corey Schumacher, David Casbeer and Derek Kingston of the Air Force Research Laboratory for their feedback on this problem. The research was supported in part by the United States Air Force grant, FA 8650-07-2-3744.

Author Contributions

Anouck Girard, Pierre Kabamba and Jonathan Las Fargeas contributed equally in the formulation and analysis of the stated problem as well as in the writing of the manuscript. Jonathan Las Fargeas developed the algorithms and ran the simulations.

Appendix

A. Local Search Heuristic

Algorithm A1: The localSearch() heuristic to find a local minimal cost action for the UAVs.

```
Data: $y(k)$,$f(\cdot)$,$x(\cdot)$,$S$,$P(S)$, $search_{init}$, $search_{iter}$
1  $\tilde{U} \leftarrow \emptyset; u \leftarrow \emptyset; \phi \leftarrow \infty$
2  for $l \leftarrow 1$ to $search_{init}$ do
3      $\kappa \leftarrow$ Generate a feasible sequence of actions of a given depth
4      $S, P(S) \leftarrow$ Compute possible intruder paths for the duration of the sequence of actions
5      $i \leftarrow -1$
6      $\tilde{U} \leftarrow$ Compute the k-neighborhood of the sequence of actions
7      $S, P(S) \leftarrow$ Extend possible intruder paths to the maximum end time of the k-neighborhood
8      $\tilde{U}.cost \leftarrow$ Compute costs for all actions
9      $\kappa_2 \leftarrow \min_{\tilde{u} \in \tilde{U}}(\tilde{u}.cost)$
10     while $\kappa_2 \neq \kappa \wedge i < search_{iter}$ do
11         $\kappa \leftarrow \kappa_2$
12         $\tilde{U} \leftarrow$ Compute the k-neighborhood of the sequence of actions
13         $S, P(S) \leftarrow$ Extend possible intruder paths to the maximum end time of the k-neighborhood
14         $\tilde{U}.cost \leftarrow$ Compute costs for all actions
15         $\kappa_2 \leftarrow \min_{\tilde{u} \in \tilde{U}}(\tilde{u}.cost)$
16         if $search_{iter} > 0 \wedge i < 0$ then
17             $i \leftarrow i + 2$
18         else if $search_{iter} > 0$ then
19             $i \leftarrow i + 1$
20     if $\kappa_2.cost < \phi$ then
21         $u \leftarrow \kappa_2$
Result: $u$
```

For comparison, a local search heuristic is also implemented to solve this problem (Algorithm A1). At each step, the algorithm randomly generates a feasible sequence of actions of a given depth (Line 3) and computes possible intruder paths that can occur in the same time window (Line 4). The k-neighborhood of the initial sequence of actions is then computed (Line 6) for k = 2, and the possible intruder paths are extended due to the increased time window (Line 7). For each sequence of actions, the corresponding cost is computed using the intruder paths (Line 8), and the minimum cost sequence of actions is extracted (Line 9). This procedure of generating a k-neighborhood and selecting the optimal sequence is repeated until convergence of the minimal cost sequence of actions or a number of iterations, $search_{iter}$, is reached (Lines 10–21). The overall procedure can also be repeated for multiple initial random feasible sequences of actions to further increase the search space. $search_{init}$ indicates the number of these initial guesses. For the search to continue until convergence, $search_{iter}$ is indicated as zero. The simulations shown in this paper use $search_{init} = 5$ and $search_{iter} = 0$.

Conflicts of Interest

The authors declare no conflict of interest.

References

1. Ratches, J. Review of Current Aided/Automatic Target Acquisition Technology for Military Target Acquisition Tasks. *Opt. Eng.* **2011**, *50*, 072001.
2. Kingston, D.B.; Rasmussen, S.J.; Mears, M.J. Base defense using a task assignment framework. In Proceedings of the AIAA Guidance, Navigation, and Control Conference, Chicago, IL, USA, 10–13 August 2009.
3. Krishnamoorthy, K.; Casbeer, D.; Chandler, P.; Pachter, M.; Darbha, S. UAV search and capture of a moving ground target under delayed information. In Proceedings of the 51st IEEE Conference on Decision and Control, Maui, HI, USA, 10–13 December 2012; pp. 3092–3097.
4. Las Fargeas, J.; Kabamba, P.; Girard, A. Optimal Configuration of Alarm Sensors for Monitoring Mobile Ergodic Markov Phenomena on Arbitrary Graphs. *IEEE Sens. J.* **2015**, accepted.
5. Hokayem, P.; Stipanovic, D.; Spong, M. On Persistent Coverage Control. In Proceedings of the 46th IEEE Conference on Decision and Control, New Orleans, LA, USA, 12–14 December 2007; pp. 6130–6135.
6. Nigam, N.; Kroo, I. Persistent Surveillance Using Multiple Unmanned Air Vehicles. In Proceedings of the IEEE Aerospace Conference, Big Sky, MT, USA, 1–8 March 2008; pp. 1–14.
7. Nigam, N.; Bieniawski, S.; Kroo, I.; Vian, J. Control of Multiple UAVs for Persistent Surveillance: Algorithm and Flight Test Results. *IEEE Trans. Control Syst. Technol.* **2011**, *20*, 1–17.
8. Elmaliach, Y.; Agmon, N.; Kaminka, G. Multi-Robot Area Patrol under Frequency Constraints. In Proceedings of the IEEE International Conference on Robotics and Automation, Roma, Italy, 10–14 April 2007; pp. 385–390.
9. Elmaliach, Y.; Shiloni, A.; Kaminka, G. A Realistic Model of Frequency-Based Multi-Robot Fence Patrolling. In Proceedings of the 7th International Joint Conference on Autonomous Agents and Multi-Agent Systems, Estoril, Portugal, 12–16 May 2008; Volume 1, pp. 63–70.
10. Cassandras, C.; Ding, X.; Lin, X. An Optimal Control Approach for the Persistent Monitoring Problem. In Proceedings of the 50th IEEE Conference on Decision and Control and European Control Conference, Orlando, FL, USA, 12–15 December 2011; pp. 2907–2912.
11. Chevaleyre, Y. Theoretical Analysis of the Multi-agent Patrolling Problem. In Proceedings of the IEEE/WIC/ACM International Conference on Intelligent Agent Technology, Beijing, China, 20–24 September 2004; pp. 302–308.
12. Wolfler Calvo, R.; Cordone, R. A Heuristic Approach to the Overnight Security Service Problem. *Comput. Oper. Res.* **2003**, *30*, 1269–1287.

13. Pasqualetti, F.; Durham, J.; Bullo, F. Cooperative Patrolling via Weighted Tours: Performance Analysis and Distributed Algorithms. *IEEE Trans. Robot.* **2012**, *28*, 1181–1188.
14. Oberlin, P.; Rathinam, S.; Darbha, S. Today's Traveling Salesman Problem. *IEEE Robot. Autom. Mag.* **2010**, *17*, 70–77.
15. Savla, K.; Bullo, F.; Frazzoli, E. On Traveling Salesperson Problem for Dubins' Vehicle: Stochastic and Dynamics Environments. In Proceedings of the 44th IEEE Conference on Decision and Control and European Control Conference, Seville, Spain, 15 December 2005; pp. 4530–4535.
16. Quilliot, A. A Short Note About Pursuit Games Played on a Graph with a given Genus. *J. Comb. Theory Ser. B* **1985**, *38*, 89–92.
17. Nowakowski, R.; Winkler, P. Vertex-to-vertex Pursuit in a Graph. *Discrete Math.* **1983**, *43*, 235–239.
18. Aigner, M.; Fromme, M. A Game of Cops and Robbers. *Discrete Appl. Math.* **1984**, *8*, 1–12.
19. Chung, T.H.; Hollinger, G.A.; Isler, V. Search and Pursuit-evasion in Mobile Robotics. *Auton. Robot.* **2011**, *31*, 299–316.
20. Basilico, N.; Gatti, N.; Amigoni, F. Developing a Deterministic Patrolling Strategy for Security Agents. In Proceedings of the IEEE/WIC/ACM International Joint Conference on Web Intelligence and Intelligent Agent Technologies, Milan, Italy, 15–18 September 2009; Volume 2, pp. 565–572.
21. Basilico, N.; Gatti, N.; Villa, F. Asynchronous Multi-Robot Patrolling against Intrusions in Arbitrary Topologies. In Proceedings of the 24th AAAI Conference on Artificial Intelligence, Atlanta, GA, USA, 11–15 July 2010; pp. 1224–1229.
22. Kingston, D. Intruder Tracking using UAV Teams and Ground Sensor Networks. In Proceedings of the German Aerospace Congress, Berlin, Germany, 11 September 2012; German Society for Aeronautics and Astronautics: Bonn, Germany, 2012.
23. Tsukamoto, K.; Ueda, H.; Tamura, H.; Kawahara, K.; Oie, Y. Deployment design of wireless sensor network for simple multi-point surveillance of a moving target. *Sensors* **2009**, *9*, 3563–3585.
24. Niedfeldt, P.; Kingston, D.; Beard, R. Vehicle State Estimation within a Road Network using a Bayesian Filter. In Proceedings of the American Control Conference, San Francisco, CA, USA, 29 June–1 July 2011; pp. 4910–4915.
25. Las Fargeas, J.; Hyun, B.; Kabamba, P.; Girard, A. Persistent Visitation under Revisit Constraints. In Proceedings of the 2013 International Conference on Unmanned Aircraft Systems, Atlanta, GA, USA, 28–31 May 2013; pp. 952–957.
26. Las Fargeas, J.; Hyun, B.; Kabamba, P.; Girard, A. Persistent Visitation with Fuel Constraints. *Proc. Soc. Behav. Sci.* **2012**, *54*, 1037–1046.
27. Klesh, A.; Kabamba, P.; Girard, A. Path Planning for Cooperative Time-optimal Information Collection. In Proceedings of the American Control Conference, Seattle, WA, USA, 11–13 June 2008; pp. 1991–1996.
28. Dai, R.; Cochran, J. Path Planning and State Estimation for Unmanned Aerial Vehicles in Hostile Environments. *J. Guid. Control Dyn.* **2010**, *33*, 595–601.

29. Las Fargeas, J. Videos of Cooperative Surveillance and Pursuit Simulations. Available online: http://arclab.engin.umich.edu/?page_id=450 (accessed on 7 January 2015).