UAV/UGV Search and Capture of Goal-oriented Uncertain Targets*

Mor Sinay, Noa Agmon, Oleg Maksimov, Guy Levy, Moshe Bitan, Sarit Kraus

***Abstract*— This paper considers a new, complex problem of UAV/UGV collaborative efforts to search and capture attackers under uncertainty. The goal of the defenders (UAV/UGV team) is to stop all attackers as quickly as possible, before they arrive at their selected goal. The uncertainty considered is twofold: the defenders do not know the attackers' location and destination, and there is also uncertainty in the defenders' sensing. We suggest a real-time algorithmic framework for the defenders, combining entropy and stochastic-temporal belief, that aims at optimizing the probability of a quick and successful capture of all of the attackers. We have empirically evaluated the algorithmic framework, and have shown its efficiency and significant performance improvement compared to other solutions.**

I. Introduction

Searching and capturing attackers (intruders) that wish to arrive at a specific goal is an important problem that can benefit from the deployment of aerial and ground autonomous vehicles. Algorithms that identify strategies for the defenders need to take into consideration the diverse capabilities, movement models and topological environments where the attackers' and the defenders' vehicles (ground and aerial) are operating. Finding intruders is time critical; hence, real-time algorithms are necessary.

Simplified versions of this challenging problem have been studied in previous work by two models: Minimum Time Search (MTS), and Reach-Avoid Games. In the MTS problem, one or more targets (in our case attackers) are in unknown locations and need to be found as quickly as possible. In the reach-avoid game, one or more attackers attempt to reach one or more goals while the defenders seek to prevent the attackers from arriving at their goals. In this paper, we study a complex, realistic model involving real-time, UAV and UGV teams facing multiple goal-oriented attackers in an unknown location. The defenders have imperfect bounded sensing capabilities traveling in two different environment models: a grid for the UAVs and a graph modeling the roads traversed by the UGVs.

We introduce the UAV/UGV Search and Capture of goal-Oriented Uncertain Targets (SCOUT). In SCOUT we have two types of defenders: Unmanned Aerial Vehicles (UAVs) and Unmanned Ground Vehicles (UGVs). The attackers are grounded, and their objective is to reach one of the goals. An attacker chooses its goal at random, and travels in the environment according to a (random) movement model that is *known* to the defender. However, the *specific* chosen goal is unknown to the defenders. The defenders aim to prevent all attackers from reaching their goals. Within the defenders, the UAVs' and UGVs' task is to detect the attackers, so defenders with an interception capability could stop the attackers' movement. The uncertainty in our model is twofold: First, the UAVs have imperfect detection probability. Second, the attackers' movement model is stochastic, and the goal selection is probabilistic.

We propose a two-stage approach: The first stage is when the attackers' location is unknown and the defenders' task is to search for the attackers. The second stage starts when an attacker is spotted by a defender. This defender will track the attackers while a team of defenders with interception capabilities move according to their movement capabilities (on a grid or a graph) and maximize the probability of stopping the attacker. Note that the attacker's goal and the path to this goal remain unknown. Tracking the attacker is only one way to solve the problem. For example, if the attacker can, for a while, move on only one road, the defender can proceed to another task. Our objective is to maximize the probability of stopping all attackers before reaching their goals.

SCOUT is a dynamic problem with high dimensionality since the world representation of a grid and a graph is enormous, adding to the multiple defenders and attackers with uncertainty regarding the attackers' location. Hence, online computation of an optimal solution for the defender is intractable for a multi-player problem. Note that the solution is intractable even when only one defender and one attacker are involved [1].

The main contribution of this paper is a combination of: the algorithmic infrastructure for efficiently solving SCOUT and MTS problems for a general graph, introducing goal-oriented attackers, and considering imperfect detection capability of the defenders with moving targets. This algorithmic infrastructure is generic and can be adapted to solving the problem with heterogenous defenders and attackers, multiple attacks and coping with environmental constraints such as a no flying zone [2].

We tested our algorithms on a realistic border defense simulation demonstrating its high success rate in the detection and interception of the attackers.

II. Related Work

Our first problem is to search for the attackers when there is incomplete information about their location. This is a generalization of the MTS problem, where one or more targets are in unknown locations and need to be found as soon as possible. There are many approaches for handling the MTS problem under the assumption of a static target [3], [4], [5], [6]; however, we are interested in moving targets.

This research was supported in part by ISF grant #1337/15 and part by a grant from MOST, Israel and the JST Japan

Lanillos et al. [7] address the MTS problem using a Partially Observable Markov Decision Process (POMDP) [39] formulation with a single attacker and a single defender. The attacker has a Markovian movement model and is not affected by the defender's location, but the attacker and the defender have the same velocity, and the defender has perfect detection, meaning that if the attacker is in the defender's detection range, the latter will detect the former. Ru et al. [8] present an algorithm for the MTS problem under the assumptions that the UAVs have uncertainty regarding their location and have restrictions over their movement capability (they can move by 45 degrees at each step). In their solution, the environment is represented as a grid with the assumption that each attacker is located in a different cell of the grid at each time step. Perez-Carabaza et al. [2] presented a variant of the MTS problem where the defenders have to avoid collisions and added a constraint of places containing *no flying zones*. They represented the environment as a grid and presented a heuristic approach for finding a single attacker using multiple UAVs when the attacker's movement model is given as a Markovian model. Perez-Carabaza et al. [9] presented a heuristic based ant colony optimization [10] for the MTS problem for multiple defenders and a single attacker. The environment is formed as a grid representation, and the target model is given in advance. However, the UAVs and the attacker have the same velocity, and the UAVs have perfect detection probability and therefore their proposed solution is not applicable for SCOUT.

In our searching solution, we use the entropy measurement [11], a standard measurement for estimating uncertainty; it is commonly used for problems with incomplete information [12], [13], [14]. For example, Kaufman et al. [13] introduced an algorithm to explore a grid map using robots where the probability of each cell being either occupied or free was considered. The robot chooses the trajectory that maximizes the map information gain. Blanco et al. [14] presented an entropy-based algorithm for the robot localization problem. They proposed an approach to measure the certainty of a robot's location based on its previous estimated location.

Previous work on reach-avoid games commonly assume that the objective function of the attackers is to arrive at a goal as quickly as possible, and therefore no model is considered for the attackers' actions beyond the limits imposed by the attackers' capabilities and the defenders' location. The classic reach-avoid game is also assumed to have perfect information of the attackers' and defenders' locations [15], [16], [17], [18], [20], [21], [22], [23], [25].

Huang et al. [15] presented a framework for analyzing and solving a two-player capture-the-flag game with full knowledge as a zero-sum differential game. This work was extended in Huang's thesis [16] where he presented a formulation of the reach-avoid game as a differential problem and solved the related Hamilton-Jacobi-Isaacs (HJI) equation [24]. Unfortunately, computing solutions to HJI equations are computationally infeasible for large problems. Even smaller problems such as 2-player games typically cannot be solved in real-time. In a more recent work, Huang et al. [17] presented a variant of the game where there are also *safe* places that attackers can travel freely without worrying that a defender will intercept their movements. They presented an algorithm for the defender that stops the attacker from reaching a goal, also considering these *safe* locations. A further study [18] addresses this same variant of the reach-avoid game, and uses an MST solver [19] to find the safest path for the attacker to reach a goal. However, there is no reference in their solution to the position or movement of the defenders (only maximization of the safest path).

Zhou et al. [20] presented an open loop reach-avoid game for two players and extended the solution for more players in [21]. Their approach is based on assuming the worst case for every player and acting accordingly. They do not assume to have an attacker model; hence, each game is played conservatively from the perspective of one player. Maximizing the cost of the game represents the part of the attackers, and minimizing this value represents the defenders.

Chen et al. [22], [23] introduced a multi-player reach-avoid game where the number of the attackers and the number of the defenders are equal. They presented a few methods of matching between an attacker and a defender and solved the game for each as a two-player game using HJI [24]. They elaborate their solution where the number of attackers and the number of defenders are not equal, as in [25]. However, HJI equations are computationally infeasible for multi-player problems and do not apply for online settings.

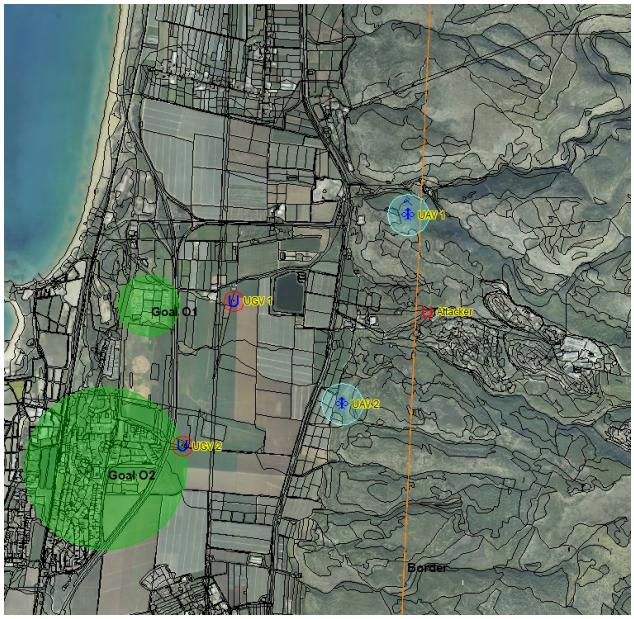
In all of these works, the attackers aim to reach a goal as fast as possible, and there is no assumption of the attackers' movement model. As opposed to all of these methods we will present a new variant of the reach-avoid game that combines complete and incomplete information about the attackers' location and destination. We assume that an attacker movement model exists. Our solution works for attackers with a simple movement model such as moving to a goal as quickly as possible (as in previous work) and more complex models such as deceptive attackers [26].

Our defenders are assumed to be UAVs and UGVs. The combination of UAVs and UGVs has gained much attention in recent research and in the development community due to their strong potential in high-risk missions [27], [28], [29], [30]. For example, Phan et al. [27] presented cooperative control of UAVs and UGVs in forest fire monitoring, detection and fighting. Bertolaso et al. [29] introduced an algorithm using UAV/UGV cooperation for a landing task problem.

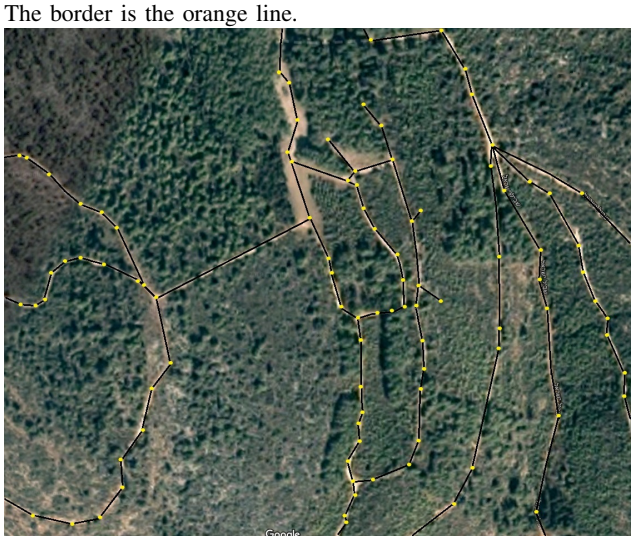
III. SCOUT Formulation

We consider a problem where a team of m defenders $D = \{d_i\}_{i=1}^{m}$ attempt to stop a set of n attackers $A = \{a_i\}_{i=1}^{n}$. The environment is modeled as a graph $G = (V, E)$ nested in a 2D environment, representing the roads on which the attackers and the ground defenders can move. Each attacker's objective is to reach a specific goal chosen in advance, where this goal is unknown to the defenders. Each goal O_i is a subset of the graph G as shown in Figure 1a, and the entire

goal set is denoted by $O = \bigcup_{i=1}^{\bar{k}} O_i$. The defenders try to protect this set of goals and prevent all of the attackers from reaching their goal. The defenders are familiar with the set O but do not know the specific goal(s) chosen by the attackers. Denote the location of a defender $d_i \in D$ and attacker $a_i \in A$ at time $t \geq 0$ by $l_{d_i}^t \in E$ and $l_{a_i}^t \in E$, respectively.



(a) Circled in green are the goals $O_i \subset G$, circled in blue are the UAVs' detection range and in red the UGVs' detection range. The border is the orange line.



(b) Zoom-in view of the graph nested in a 2D environment. Between every two vertices is a straight line.

Fig. 1: 1a shows the environment overview from the bird's eye and 1b shows an example of the graph.

A. Defenders' Model

As mentioned before, we assume to have two types of defenders: UAVs and UGVs. We denote the UAV team by $D_u = \{d_i\}_{i=1}^{m_u}$ and the UGV team by $D_v = \{d_i\}_{i=1}^{m_v}$ where $D = D_u \cup D_v$ $(m = m_u + m_v)$. The UAVs and UGVs have limited visibility, hence we denote the detection range (radius) of defender d_i as r_i. Denote the UAVs' minimum detection range by $r = \min_{1 \leq i \leq m_u} \{r_i\}$. Defender d_i can detect and track an attacker if it is within its detection range r_i. We assume that only a UGV can stop an attacker[1]. A UGV can stop an attacker if it is within its detection range. The UAVs have imperfect detection probability. That is, if an attacker is within the detection range of a UAV defender d_i, it will detect it with a probability of $0 < p_i^d \leq 1$. Denote the minimum probability of detection by $p^d = \min_{1 \leq i \leq m_u} p_i^d$.

We assume perfect detection probability for every UGV $d_i \in D_v$ (that is, $p_i^d = 1$). The UAVs and UGVs have different movement capabilities. This will be elaborated in section IV.

B. Attackers' Model

We initially assume that all n attackers are assumed to enter the graph at the same arbitrary location[2] denoted by l_a^0. Each attacker selects a goal O_i at random (if not specified otherwise we assume with uniform distribution). The attacker follows a Markovian movement model towards its chosen goal. For each edge $e_i \in E$, $M_a(e_i)$ defines the probability distribution that an attacker coming from edge e_i will choose to move to any of its neighbors $\{e_i^j\}$. That is, $M_a(e_i) = \{(e_i^j, p_i^j)\}$ and $\sum_j p_i^j = 1$. $\bar{M}_a(e_j) = \{(e_j^i, p_j^i)\}$ defines the probability distribution that an attacker coming from edge e_j^i will choose to move to edge e_j. We use a general model and could apply different movement models to the different attackers, for simplicity in this paper, we will assume the attackers are homogeneous and therefore use the same movement model (though their actual random choices may be different).

C. Problem Formulation

Given a team of defenders and attackers as defined above, the game starts at time $t = 0$ when the attackers are located at l_a^0. All players move simultaneously according to their movement capabilities.

We say that a UGV defender d_i stopped an attacker a_j if and only if $\exists t$ such that $l_{a_j}^t$ is within the detection range of d_i (which is located at $l_{d_i}^t$) and $l_{a_j}^t \notin O$.

If $\exists t$ such that $l_{a_i}^t \in O$, then the defenders lose. Hence, the defenders' goal is to prevent all attackers from arriving at their goals. Our objective is to determine, for each defender $d \in D$ at each time t, a destination that will maximize the probability that the team of defenders will stop all attackers.

IV. Solving SCOUT

We divide the solution concept into two main steps: search and intercept. First, we search for the attackers when we only know their initial location and movement model. When an attacker is spotted, we move to the second step. A defender

[1]Giving the UAVs the ability to stop the attacker would simplify the problem.

[2]We can generalize SCOUT to include heterogeneous attackers with different possible entry points. (See discussion in section VI

would track the attacker's progress and try to intercept its movement. If there is more than one attacker, some defenders will search for attackers while the other defenders try to catch the attackers that have already been found. The robot heterogeneity dictates that not all robots can perform all tasks, hence we address the problem sequentially. The UAVs' defenders will perform the search task, mainly because they are fast and have no limitation on where they can move, as opposed to the UGVs that can only travel along the graph's edges. If a UAV detects an attacker, it will track it while a subset of UGVs is assigned to intercept its movement. Although the location of the attacker is now known, the problem is still not a solved version of the reach-avoid game since the attacker is heading towards a specific goal (unknown to the defenders) and not just any goal. Predicting the attacker's path to any destination is a difficult task. In our case, we need to predict the attacker's path to any of the goals and choose the most likely one. The prediction error in our case could be crucial because the goals are not necessarily close to each other. Adding to the complexity of our problem is that each attacker could choose a different goal.

We create a probabilistic belief model $P(e_i, t)$. That is, our belief as to the possibility that an attacker is located at edge $e_i \in E$ at time t, and $\forall t \sum_{e_i \in E} P(e_i, t) = 1$. Note that $P(l_a^0, 0) = 1$ and $\forall e_i \in E, e_i \neq l_a^0\ P(e_i, 0) = 0$, as l_a^0 is the initial location of the attackers. The transition between $P(e_j, t)$ to $P(e_j, t+1)$ is described in Eq. 1 and depends on the attackers' movement model $\bar{M}_a(e_j)$ (described in subsection III-B).

$$P(e_j, t+1) = \sum_{(e_j^i, p_j^i) \in \bar{M}_a(e_j)} P(e_j^i, t) \times p_j^i \qquad (1)$$

The state space is huge (for example, in our simulation $|E| = 16000$), and we need to propose an online real-time solution. Thus, we create the following infrastructure that combines offline preprocessing to support the online decision-making.

A. Offline Preprocessing To Generate a Grid/Graph

The UAV is not limited to traveling along the graph. Thus, when a UAV is located on an edge e, it will potentially gain information from all edges in G that are within its detection range. Therefore, we create a grid representation $C = \{c_i\}_{i=1}^N$ layered on top of the graph (see Figure 2). Each cell c_i in the grid is of size $\sqrt{2}r \times \sqrt{2}r$, that is the maximum square that is contained in a circle with radius r (the minimum detection range capability of the UAVs).

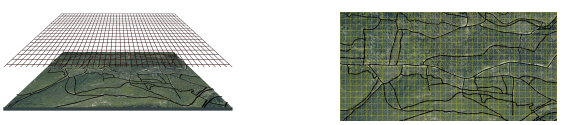


Fig. 2: A grid layered on top of the graph

After constructing the grid and layering it on top of the graph, we will cut the roads (edges of G) into multiple edges (and vertices) so that each edge will be in a single cell.

The attackers move along the graph edges. But the length of each edge is different because this graph is constructed from a roadmap. To achieve higher resolution for the probability belief of the attackers' location, we defined a threshold Th^e and split all edges (roads) with a length of more than Th^e into new edges (and vertices) with a length that is less than or equal to Th^e.We split an edge e to $\lceil |e|/Th^e \rceil$ edges. All new edges are of length Th^e aside from one edge that could be less than or equal to Th^e.

This offline process produces a new graph $\bar{G} = (\bar{V}, \bar{E})$ and a grid representation $C = \{c_i\}_{i=1}^N$. We denote by $P(c_i, t)$ our belief as to the possibility that an attacker is located in *cell* $c_i \in C$ at time t. $P(c_i, t) = \sum_{\bar{e}_j \in c_i} P(\bar{e}_j, t)$.

B. Search task

The defenders' goal is to stop the attackers before the latter arrive at their goals. Although only the UGVs can stop the attackers, the UAVs can reduce the uncertainty on the attackers' location during the search. A search strategy for a UAV should specify to which cell to move in order to perform detection of an attacker. We first considered the Monte Carlo Tree Search (MCTS) approach since the SCOUT problem can be modeled as a POMDP. Given the MCTS's failure to solve the SCOUT problem (see Section IV-B.1), we propose the entropy-based heuristic called Max Gain max Probability (MGP) and show its success.

1) MCTS Approach: MCTS [31] is a simulation-based search algorithm for finding optimal strategies. Information set MCTS (ISMCTS)[32] is an extension of MCTS to Imperfect Information games. ISMCTS variations have shown great success in imperfect information games such as poker and hedge. For the UAV search phase, the Smooth-UCT [33] variation was implemented. While Smooth-UCT does not guarantee convergence to Nash-Equilibrium, it does converge to a sub-optimal strategy much faster than other variations that offer such guarantees (e.g. Online Outcome Sampling [34]). For this reason, Smooth-UCT was selected as the online MCTS algorithm in our simulations. However, the search space was proven to be more challenging than what ISMCTS can handle in a real-time environment. More specifically, for a grid size of 16 by 30, there are 480 possible actions a single defender/UAV can perform. For multiple UAVs, the number of possible actions the defenders can perform are 480^k where k is the number of UAVs. In addition, the average number of steps in each simulation is 35. This results in an approximate search space size of $10^{92} \approx 480^{35}$ for a single UAV. This search space increases exponentially as the number of UAVs grows. Furthermore, in our settings, the defenders have 10 seconds to calculate their route and make a decision on where to deploy the UAVs. In our simulations, the Smooth-UCT was able to perform approximate $250k$ simulations, which is an order of magnitude less than what is needed to converge to an optimal

strategy. For compression, the search space in computer poker is 10^{18} [35], and at least 10 million simulations are used [33]. This suggests that it is more plausible to use a heuristic based approach to the UAV's deployment problem.

2) MGP Approach: We first consider the entropy measurement to greedy allocate cells for the UAVs in this task. The entropy measure based on our belief model is defined by:

$$E(C,t) = -\sum_{c_i \in C} P(c_i,t) log(P(c_i,t)) \quad (2)$$

Denote by k the number of UAVs that search for the attackers ($k \leq m_u$). Assume that a UAV is located in cell c_i at time t. If the UAV detects an attacker, then the search task is done for this UAV, and it will start tracking the attacker. However, if the UAV did not detect an attacker this is not necessarily because the attacker is not located in cell c_i. The probability that an attacker is at cell c_i and the UAV did not detect it is $1-p^d$. Therefore, if a UAV did not find an attacker at cell c_i at time t, we will update the belief model for each $e_i \in c_i$ according to the probability ($P(e_i,t) = (1-p^d)P(e_i,t)$). The delta between these two beliefs would split relatively between all cells, similar to a POMDP update (see Eq. 3 for single cell update).

$$P(e_i,t) = \begin{cases} \frac{P(e_i,t)}{P(c_i,t)\times p^d} & \text{if } e_i \notin c_i \\ \frac{P(e_i,t)\times(1-p^d)}{P(c_i,t)\times p^d} & \text{if } e_i \in c_i \end{cases} \quad (3)$$

The entropy of our belief at time t was defined in Eq. 2. The temporal entropy $E_u(\{c_i\}_{i=1}^k, t)$ returns the new entropy after sending k UAVs to k cells under the assumption that the UAVs did not detect any attacker (shown in Alg. 1).

Algorithm 1 Entropy Update: $E_u(\{c_i\}_{i=1}^k, t)$

$\eta = 1 - \sum_{\{c_i\}_{i=1}^k} P(c_i,t) \times p^d$
for all $c_j \in C$ **do**
  if $c_j \in \{c_i\}_{i=1}^k$ **then**
    $P_{temp}(c_j,t) = \frac{P(c_j,t)\times(1-p^d)}{\eta}$
  else
    $P_{temp}(c_j,t) = \frac{P(c_j,t)}{\eta}$
  end if
end for
return $-\sum_{c_j \in C} P_{temp}(c_j,t) log_2(P_{temp}(c_j,t))$

Therefore, we define the entropy gain $G(C,t,k)$ as the sum of entropy difference. The entropy difference when a UAV detects an attacker is $p^d \times P(c_i,t)(E(C,t)-0)$ (because the new entropy is zero). Added to the entropy difference is the probability that the attacker is in this cell but it did not detect it, or the probability that the attacker is not in this cell, that is $(1-p^d \times P(c_i,t))(E(C,t) - E_u(\{c_i\}_{i=1}^k, t)))$. The final entropy gain is shown in Eq. 4.

$$G(C,t,k) = E(C,t) - \prod_{\{c_i\}_{i=1}^k} (1-p^d \times P(c_i,t)) E_u(\{c_i\}_{i=1}^k, t) \quad (4)$$

Note that finding a subset of k cells that maximize the entropy gain is equivalent to finding a subset of k cells that minimize $\prod_{\{c_i\}_{i=1}^k} (1-P(c_i,t)p^d) E_u(\{c_i\}_{i=1}^k, t)$.

Finding k cells that maximize the entropy gain is shown in Eq. 5.

$$G_m(C,t,k) = \underset{\{c_i\}_{i=1}^k \in C}{\arg\min} \prod_{\{c_i\}_{i=1}^k} (1-p^d P(c_i,t)) E_u(\{c_i\}_{i=1}^k, t) \quad (5)$$

Entropy is a submodular function [36], [37], meaning that we can use an online greedy algorithm for a near-optimal solution [38] for determining the best assignments of k UAVs to k cells (without checking all $\binom{N}{k}$ combinations of cells).

Lemma 4.1: When $p^d = 1$ then $G_m(C,t,k)$ (Eq. 5) returns the k cells with the highest probability of being the attacker's location (that is, $\underset{\{c_i\}_{i=1}^k \in C}{\arg\max} \sum_{\{c_i\}_{i=1}^k} P(c_i,t)$).[3]

If $p^d < 1$, the maximum entropy gain is not necessarily the maximum cell probability. For example, consider the simple case of two cells $C = \{c_1, c_2\}$, with a probability of $P(c_1,t) = 0.9$ and $P(c_2,t) = 0.1$ that there is an attacker located in that cell at time t and $p^d = 0.9$. In this case, according to Eq. 4 the entropy gain from sending a UAV to c_1 and c_2 is 0.28 and 0.39, respectively. Using *only* the entropy measurement, the assignment for this UAV would be to c_2 although the *probability* of having an attacker there is only 0.1. However, we are not interested only in reducing the entropy, but in detecting the attacker. Therefore, we propose a Max Gain max Probability (MGP) approach that combines maximum gain and maximum probability. We will assign a UAV to search the cell with the maximum probability, and all other $k-1$ UAVs will search according to the maximum gain (see Algorithm 2).

We define a threshold Th^p (hyperparameter) such that if there is a cell c_i at time t with $P(c_i,t) \geq Th^p$, we will assign a UAV to search this cell, and all other $k-1$ UAVs will search according to the maximum gain (see Algorithm 2). Note that when choosing threshold $Th^p = 1$, MGP will assign all k UAVs according to the entropy gain. For $Th^p = 0$, MGP will assign one UAV to the cell with the maximum probability and the rest according to the entropy gain.

Algorithm 2 Cell Search Subset

if $\exists c_j \in C$ such that $P(c_j,t) \geq Th^p$ **then**
  return $c_j \bigcup G_m(C \setminus c_j, t, k-1)$
else
  return $G_m(C,t,k)$
end if

C. Tracking task

When a UAV detects an attacker, it is assigned with a tracking task to continue following this attacker. Hence, the location of the attacker is known. Now we can assign a UGV

[3]Due to space constraint the proof can be found at `https://www.dropbox.com/s/grfz7voryhowkf3/Proof.pdf?dl=0`

subset to begin the interception task. Note, if there are still attackers with an unknown location, *the UAVs that are not tracking an attacker* are still assigned with the search task.

D. Interception task

An attacker's location is known (a UAV is tracking the attacker), but the chosen path of the attacker is unknown. Our goal is to predict the attacker's location and to send a UGV to intercept its movement. We present two algorithms: the Maximize Probability min Distance (MPD) algorithm and the Min Max Probability (MMP) algorithm. In both heuristics we take two factors into consideration: first, the probability that the attacker will be in a specific location. Second, the distance of that location from the UGV's location.

Denote by $L^p(a_i, t+1)$ and $L^p(d_i, t+1)$ the set of possible locations of an attacker a_i and a defender d_i, respectively, at time $t+1$ (as shown in Figure 3). MPD (Alg. 3) chooses the UGV's next location while taking into account all the possible locations of the attacker, and MMP (Alg. 4) considers only the maximum probability location. $D(e_i, e_j)$ denotes the shortest path between edges e_i and e_j (as shown in Figure 3).

Fig. 3: Possible locations of attacker and defender at $t+1$

Algorithm 3 MPD

$E_d = L(d_j, t+1)$
$E_a = L(a_i, t+1)$
return $\arg\max_{e_d \in E_d} \sum_{e_a \in E_a} P(e_a, t+1)(1/D(e_a, e_d))$

Algorithm 4 MMP

$E_d = L(d_j, t+1)$
$e_a^{max} = \arg\max_{e_a \in L(a_i,t)} P(e_a, t+1)$
return $\arg\min_{e_d \in L(d_j,t)} D(e_a^{max}, e_d)$

V. Experimental evaluation

To evaluate our MGP algorithm we used the Simax Smart scenario Generator (SSG)[4], a graphical online real-time simulation, which simulates a border protection scenario. In this simulation, the defenders have three goals to protect ($|O| = 3$). There are a total of $105,000$ roads in the environment (see partially Fig. 1a). We assume that we have the nested roadmap mapped as a graph.

We have simulated a real-life scenario, in which the mathematical Markovian model of the attacker's movements is not given to us, but its strategy is known. In our experiments, the attacker's strategy is to select an entry point and a goal at random, then choose a set of locations from a predefined radius around the goal, and follow the shortest path towards a randomly-selected location from that set (for creating deception).

We have extensively evaluated the performance of our algorithms in the two stages of SCOUT: the search and the interception. In the experiments, we varied the number of UAVs, the number of UGVs, the number of attackers and the detection probability p^d of the UAVs. In all cases, the velocity of the UAVs is $100km/h$, the UGVs travel at $50km/h$, the attacker travels at a random velocity between $8-10km/h$, and the attackers' initial location is chosen randomly.

A. Search Stage

We have compared our solution to three alternative algorithms, as described in Table I. The *Rand* algorithm chooses naïvely to go to k cells with a *non-zero* probability. In addition, we wanted to evaluate the two components of the MGP algorithm: the component that aims at maximizing the probability of immediately locating the attacker and the component that aims at minimizing the uncertainty by maximizing the entropy gain. The *MaxProb* algorithm chooses to go to the k cells with the maximum probability (this is equivalent to choosing threshold $Th^p = 0$ for a single UAV.) The *EntropyGain* algorithm chooses to go to the k cells that maximize the gain (see Eq.4) (this is equivalent to choosing threshold $Th^p = 1$). As mentioned in Section IV-B, MGP is a combination of the MaxProb algorithm and the EntropyGain algorithm.

We focused on a challenging setting where the UAVs are located *far away from the location on the border* where the attackers enter. This allows the attacker to progress towards its goal before the UAVs begin the search. All of the results presented below are the average of at least 40 simulation runs. There were a total of more than 2500 simulations.

Name	Algorithm
MaxProb	$\arg\max_{\{c_i\}_{i=1}^k \in C} P(c_i, t)$
Rand	$\arg\text{rand}_{\{c_i\}_{i=1}^k \in C} P(c_i, t) \neq 0$
EntropyGain	$G_m(C, t, k)$
MGP	Alg.2 Using $Th^p = 0.4$

TABLE I: Algorithms to compare the search algorithms

Fig. 4 presents the Th^p that maximizes the success of detecting the attacker as p^d changes. As seen, the lower the detection probability is, the higher the chosen threshold

[4] www.hartech.co.il

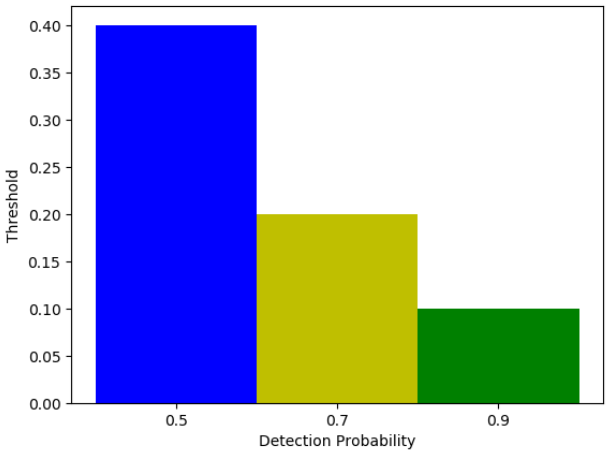
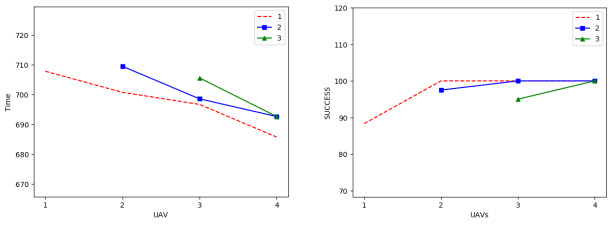


Fig. 4: The threshold Th^p that maximizes the detection probability .

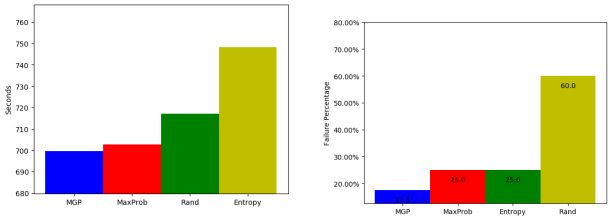
is. Meaning that as the uncertainty of the UAVs increases, the more impact the entropy measurement will have on the selection of the cells (rather than moving to the cell with the highest probability).



(a) Detection time. (b) Success rate.

Fig. 5: MGP performance in the search stage for one to three attackers and one to four UAVs with $Th^p = 0.2$ and $p^d = 0.7$.

Next we wanted to check the influence of the number of UAVs on the MGP's detection performance. We considered one, two and three attackers (the initial location was chosen randomly). As the number of UAVs increases, the success rate of capturing the attackers increases. In particular, when the number of UAVs is at least the number of attackers +1, then the success rate is perfect (1) (see Fig. 5b). The results presented in Fig. 5a indicate that as the number of UAVs increases the average detection time decreases.



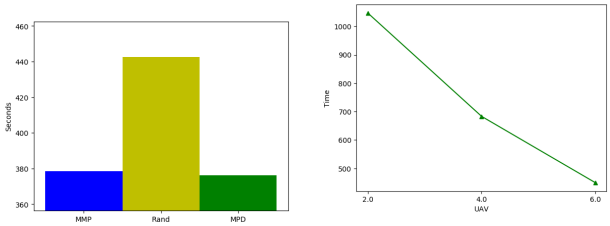
(a) Detection time. (b) Failure rate.

Fig. 6: Comparing search algorithms for one UAV, one UGV and one attacker where $p^d = 0.7$, $Th^p = 0.2$.

As seen from Fig. 6a, MGP's detection time is the lowest compared to all three algorithms (statistically significant using the ANOVA test, with $p - value = 3.7e - 05$). Fig. 6b presents the failure rate (the attacker reached the goal). MGP outperforms the other three algorithms. MGP's search algorithm performance is higher not only in detecting all attackers but also in finding them faster.

B. Intercept Stage

We have compared our methods, the MPD and MMP, to the *Rand* algorithm that chooses to send a UGV defender d_j to a possible location (edge $e \in L(d_j, t)$). The UAVs start tracking the attacker from the border, to compare the time regardless of the searching time. The detection time in MPD and MMP is almost the same (MPD is slightly better), however both algorithms perform much better than random (as can be seen in Fig. 7a). Fig. 7b presents the interception time for a changing number of UGVs with one attacker and one UAV. As can be seen in the figure, as the number of UGVs increases the interception time decreases.



(a) Comparing interception algorithm for one UAV, one UGV and one attacker.

(b) Comparing MPD algorithm for one UAV, one attacker with a changing number of UGVs.

Fig. 7: Intercepting algorithm comparison.

VI. Conclusion

In this paper, we introduced the problem of searching and capturing attackers that wish to arrive at a specific goal, under uncertainty (SCOUT). We presented an algorithmic infrastructure for efficiently solving SCOUT and MTS problems for a general graph nested in a 2D environment, assuming imperfect detection of the robots. We have shown an entropy-based algorithm that also refers to the maximum likelihood of finding an attacker. Our suggested algorithmic solution was tested in a realistic simulation, showing the real-time efficient performance of our framework, also compared to other solutions. For future work, we plan to extend our empirical evaluation to the case of several entry points of the attackers (initial location) and a different start time for every attacker. We would also like to examine the framework in other settings, for example search and rescue.

References

[1] KE Trummel and JR Weisinger. The complexity of the optimal searcher path problem. *Operations Research*, 34(2):324–327, 1986.

[2] Sara Perez-Carabaza, Julian Bermudez-Ortega, Eva Besada-Portas, Jose A Lopez-Orozco, and Jesus M de la Cruz. A multi-uav minimum time search planner based on aco r. In *Proceedings of the Genetic and Evolutionary Computation Conference*, pages 35–42. ACM, 2017.

[3] Sara Perez-Carabaza, Eva Besada-Portas, Jose A Lopez-Orozco, and Jesus M de la Cruz. A real world multi-uav evolutionary planner for minimum time target detection. In *Proceedings of the 2016 on Genetic and Evolutionary Computation Conference*, pages 981–988. ACM, 2016.

[4] Seng Keat Gan and Salah Sukkarieh. Multi-uav target search using explicit decentralized gradient-based negotiation. In *Robotics and Automation (ICRA), 2011 IEEE International Conference on*, pages 751–756. IEEE, 2011.

[5] Pablo Lanillos, Seng Keat Gan, Eva Besada-Portas, Gonzalo Pajares, and Salah Sukkarieh. Multi-uav target search using decentralized gradient-based negotiation with expected observation. *Information Sciences*, 282:92–110, 2014.

[6] Yanli Yang, Ali A Minai, and Marios M Polycarpou. Decentralized cooperative search in uav's using opportunistic learning. 2002.

[7] Pablo Lanillos, Eva Besada-Portas, Gonzalo Pajares, and José J Ruz. Minimum time search for lost targets using cross entropy optimization. In *Intelligent Robots and Systems (IROS), 2012 IEEE/RSJ International Conference on*, pages 602–609. IEEE, 2012.

[8] Chang-jian Ru, Xiao-ming Qi, and Xu-ning Guan. Distributed cooperative search control method of multiple uavs for moving target. *International Journal of Aerospace Engineering*, 2015, 2015.

[9] Sara Perez-Carabaza, Eva Besada-Portas, Jose A Lopez-Orozco, and M Jesus. Ant colony optimization for multi-uav minimum time search in uncertain domains. *Applied Soft Computing*, 62:789–806, 2018.

[10] Marco Dorigo, Vittorio Maniezzo, and Alberto Colorni. Ant system: optimization by a colony of cooperating agents. *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, 26(1):29–41, 1996.

[11] Cyrill Stachniss, Giorgio Grisetti, and Wolfram Burgard. Information gain-based exploration using rao-blackwellized particle filters. In *Robotics: Science and Systems*, volume 2, pages 65–72, 2005.

[12] Joan Vallvé and Juan Andrade-Cetto. Active pose slam with rrt. In *Robotics and Automation (ICRA), 2015 IEEE International Conference on*, pages 2167–2173. IEEE, 2015.

[13] Evan Kaufman, Taeyoung Lee, and Zhuming Ai. Autonomous exploration by expected information gain from probabilistic occupancy grid mapping. In *Simulation, Modeling, and Programming for Autonomous Robots (SIMPAR), IEEE International Conference on*, pages 246–251. IEEE, 2016.

[14] Jose-Luis Blanco, Juan-Antonio Fernández-Madrigal, and Javier Gonzalez. An entropy-based measurement of certainty in rao-blackwellized particle filter mapping. In *Intelligent Robots and Systems, 2006 IEEE/RSJ International Conference on*, pages 3550–3555. IEEE, 2006.

[15] Haomiao Huang, Jerry Ding, Wei Zhang, and Claire J Tomlin. A differential game approach to planning in adversarial scenarios: A case study on capture-the-flag. In *Robotics and Automation (ICRA), 2011 IEEE International Conference on*, pages 1451–1456. IEEE, 2011.

[16] Haomiao Huang. *Reachability-based control for human and autonomous agents in adversarial games*. PhD thesis, Stanford University, 2012.

[17] Zhenqi Huang, Yu Wang, Sayan Mitra, Geir E Dullerud, and Swarat Chaudhuri. Controller synthesis with inductive proofs for piecewise linear systems: An smt-based algorithm. In *Decision and Control (CDC), 2015 IEEE 54th Annual Conference on*, pages 7434–7439. IEEE, 2015.

[18] Zhenqi Huang, Yu Wang, Sayan Mitra, and Geir Dullerud. Controller synthesis for linear dynamical systems with adversaries. In *Proceedings of the Symposium and Bootcamp on the Science of Security*, pages 53–62. ACM, 2016.

[19] Leonardo De Moura and Nikolaj Bjørner. Z3: An efficient smt solver. *Tools and Algorithms for the Construction and Analysis of Systems*, pages 337–340, 2008.

[20] Zhengyuan Zhou, Ryo Takei, Haomiao Huang, and Claire J Tomlin. A general, open-loop formulation for reach-avoid games. In *Decision and Control (CDC), 2012 IEEE 51st Annual Conference on*, pages 6501–6506. IEEE, 2012.

[21] Zhengyuan Zhoua, Jerry Dingb, Haomiao Huangc, Ryo Takeid, and Claire Tomline. Efficient algorithms for open-loop reach-avoid games.

[22] Mo Chen, Zhengyuan Zhou, and Claire J Tomlin. Multiplayer reach-avoid games via low dimensional solutions and maximum matching. In *American Control Conference (ACC), 2014*, pages 1444–1449. IEEE, 2014.

[23] Mo Chen, Zhengyuan Zhou, and Claire J Tomlin. A path defense approach to the multiplayer reach-avoid game. In *Decision and Control (CDC), 2014 IEEE 53rd Annual Conference on*, pages 2420–2426. IEEE, 2014.

[24] Lawrence Craig Evans and Panagiotis E Souganidis. Differential games and representation formulas for solutions of hamilton-jacobi-isaacs equations. Technical report, WISCONSIN UNIV-MADISON MATHEMATICS RESEARCH CENTER, 1983.

[25] Mo Chen, Zhengyuan Zhou, and Claire J Tomlin. Multiplayer reach-avoid games via pairwise outcomes. *IEEE Transactions on Automatic Control*, 62(3):1451–1457, 2017.

[26] Peta Masters and Sebastian Sardina. Deceptive path-planning. In *IJCAI 2017*, pages 4368–4375. AAAI Press, 2017.

[27] Connie Phan and Hugh HT Liu. A cooperative uav/ugv platform for wildfire detection and fighting. In *System Simulation and Scientific Computing, 2008. ICSC 2008. Asia Simulation Conference-7th International Conference on*, pages 494–498. IEEE, 2008.

[28] Steven L Waslander. Unmanned aerial and ground vehicle teams: Recent work and open problems. In *Autonomous control systems and vehicles*, pages 21–36. Springer, 2013.

[29] Andrea Bertolaso, Masoume M Raeissi, Alessandro Farinelli, and Riccardo Muradore. Using petri net plans for modeling uav-ugv cooperative landing. In *ECAI*, pages 1720–1721, 2016.

[30] Lukas Klodt, Saman Khodaverdian, and Volker Willert. Motion control for uav-ugv cooperation with visibility constraint. In *Control Applications (CCA), 2015 IEEE Conference on*, pages 1379–1385. IEEE, 2015.

[31] Rémi Coulom. Efficient selectivity and backup operators in monte-carlo tree search. In *International conference on computers and games*, pages 72–83. Springer, 2006.

[32] Peter I Cowling, Edward J Powley, and Daniel Whitehouse. Information set monte carlo tree search. *IEEE Transactions on Computational Intelligence and AI in Games*, 4(2):120–143, 2012.

[33] Johannes Heinrich and David Silver. Smooth uct search in computer poker. In *IJCAI*, pages 554–560, 2015.

[34] Viliam Lisỳ, Marc Lanctot, and Michael Bowling. Online monte carlo counterfactual regret minimization for search in imperfect information games. In *Proceedings of the 2015 International Conference on Autonomous Agents and Multiagent Systems*, pages 27–36. International Foundation for Autonomous Agents and Multiagent Systems, 2015.

[35] Darse Billings, Neil Burch, Aaron Davidson, Robert Holte, Jonathan Schaeffer, Terence Schauenberg, and Duane Szafron. Approximating game-theoretic optimal strategies for full-scale poker. In *IJCAI*, pages 661–668, 2003.

[36] András Frank. Submodular functions in graph theory. *Discrete Mathematics*, 111(1-3):231–243, 1993.

[37] Mokshay Madiman. On the entropy of sums. In *Information Theory Workshop, 2008. ITW'08. IEEE*, pages 303–307. IEEE, 2008.

[38] Uriel Feige and Jan Vondrak. Approximation algorithms for allocation problems: Improving the factor of 1-1/e. In *Foundations of Computer Science, 2006. FOCS'06. 47th Annual IEEE Symposium on*, pages 667–676. IEEE, 2006.

[39] George E Monahan. State of the art-a survey of partially observable markov decision processes: theory, models, and algorithms. *Management Science*, 28(1):1–16, 1982.