A Search and Detection Autonomous Drone System: From Design to Implementation

Mohammadjavad Khosravi, Rushiv Arora, Saeede Enayati, *Member, IEEE*, and Hossein Pishro-Nik, *Member, IEEE*

***Abstract*— Utilizing autonomous drones or unmanned aerial vehicles (UAVs) has shown great advantages over preceding methods in support of urgent scenarios such as search and rescue (SAR) and wildfire detection. In these operations, search efficiency in terms of the amount of time spent to find the target is crucial since with time the survivability of the missing person decreases or wildfire management becomes more difficult with disastrous consequences. In this work, we consider the scenario where a drone is intended to search and detect a missing person (e.g., a hiker or a mountaineer) or a potential fire spot in a given area. To obtain the shortest path to the target, a general framework is provided to model the problem of target detection when the target's location is probabilistically known. To this end, two algorithms are proposed: Path planning and target detection. The path planning algorithm is based on Bayesian inference and the target detection is accomplished by using a residual neural network (ResNet) trained on the image dataset captured by the drone as well as existing pictures and datasets on the web. Through simulation and experiment, the proposed path planning algorithm is compared with two benchmark algorithms. It is shown that the proposed algorithm significantly decreases the average time of the mission.**

***Note to Practitioners*—This article is motivated by the need for an efficient path-planning algorithm for drones during specific SAR operations. In particular, situations where someone is lost in a snow-covered hike and a fire spot that is in its initial levels are of interest. In fact, since the target location is not known, it is required that the UAV be able to efficiently search the entire area until it finds the target in the shortest possible time. The proposed Bayesian framework along with the ResNet learning algorithm shows an efficient performance in terms of average time duration and accuracy, respectively. The framework developed in this paper can be extended to a multi-UAV scenario where UAVs coordinate to optimize the overall performance.**

***Index Terms*— Autonomous drones, unmanned aerial vehicles (UAVs), search and rescue (SAR), fire detection, path planning, machine learning.**

Manuscript received 14 May 2023; revised 2 December 2023 and 23 March 2024; accepted 13 April 2024. Date of publication 8 May 2024; date of current version 14 February 2025. This article was recommended for publication by Associate Editor T. Zhang and Editor Z. Li upon evaluation of the reviewers' comments. This work was supported by NSF under Grant CNS-1932326 and Grant CNS-2150832. *(Corresponding author: Saeede Enayati.)*

Mohammadjavad Khosravi and Hossein Pishro-Nik are with the Department of Electrical and Computer Engineering, University of Massachusetts at Amherst, Amherst, MA 01003 USA (e-mail: mkhosravi@umass.edu; pishro@engin.umass.edu).

Rushiv Arora is with the Manning College of Information and Computer Sciences, University of Massachusetts at Amherst, Amherst, MA 01003 USA (e-mail: rrarora@umass.edu).

Saeede Enayati was with the Department of Electrical and Computer Engineering, University of Massachusetts, Amherst, MA 01003 USA. She is now with Tiami Networks, Greater Sacramento, Elk Grove, CA 95624 USA (e-mail: saeedenyt@gmail.com).

Digital Object Identifier 10.1109/TASE.2024.3395409

I. Introduction

Autonomous drones or unmanned aerial vehicles (UAVs) offer numerous advantages over conventional human-intervened target detection methods. These advantages include rapid deployment, autonomous mobility capability, low cost, and the ability to reach hard-to-access areas. In a typical scenario, UAVs are deployed in an area of interest to perform sensory operations, collect evidence of the presence of a target, and report their findings to a remote ground station or rescue team [1], [2].

However, to detect objects in these scenarios using learning algorithms, especially neural networks, large and relevant datasets are required [3]. On the other hand, to decrease the detection and rescue operation's delay, it is essential to design time-efficient path planning algorithms for the UAVs which is not as straightforward as other operations such as object tracking, package delivery, imaging, etc., as the object's location is not known, or it is stochastically known.

This paper considers scenarios in which an autonomous drone is deployed to detect targets in a region, with a focus on two specific systems as examples: (1) A UAV can be employed to locate a missing individual, such as a mountain climber who may have become stranded due to an avalanche or snowfall, and may require urgent assistance. This scenario is referred to as the search and rescue (SAR) operations. (2) The UAV is deployed in a forest area in surveillance mode to detect a potential fire spot.[1] This scenario is referred to as fire surveillance. Both applications are of great interest [4], [5], [6], [7], [8], [9], [10], [11], [12], [13], [14], [15], [16], [17].

When a drone is used to search for a lost person or a potential fire spot, it is critical that the target is located as quickly as possible. Any delays in finding the target can decrease the person's chance of survival or make wildfire management more challenging. Therefore, the top priority is to minimize detection time.

Furthermore, in such operations, the drone uses its camera to monitor the area and find the target. Since the target objects are relatively small and often camouflaged within the environment, it is very important that a detection algorithm can accurately recognize the target from its surroundings. A typical aerial image of a snow-covered area is shown in Figure 1 where

[1]In this paper, the fire is assumed to be in its initial state. Therefore, it is treated as a point object and no propagation model has been considered for the fire.

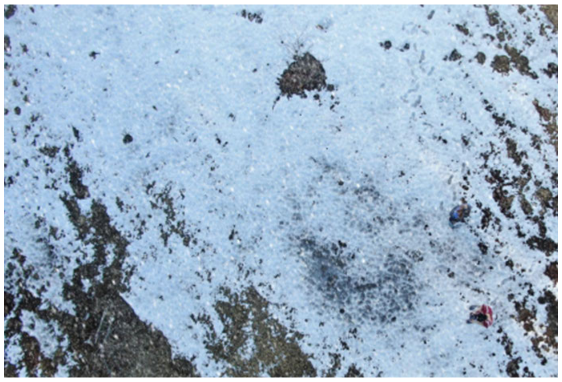


Fig. 1. An example of an aerial image from a snowy mountain with a person at the bottom right corner.

a person is also located at the bottom right corner of the picture.

Therefore, in this paper, a path-planning algorithm for a drone is developed to find the shortest path to the target while the location of the target is not deterministically known. Using prior knowledge of the target location's distribution, the path planning algorithm covers the whole area in order to detect the target, while the drone will follow the chosen path and collect data for processing.

A Machine learning (ML) technique is implemented to quickly and accurately detect the target. Specifically, a residual neural network (ResNet) is utilized in which the dataset used for the training and validation is a combination of the existing datasets as well as pictures taken by the drone where in the case of fire surveillance, mostly include small fire spots or smoke in a rural area with other objects around (spare). This is because the goal is to achieve early detection when the fire is in its initial stages.

The most similar works to this paper are [18], [19], [20] where UAVs are used to detect fire locations. While these works share some similarities, such as the target location is not known, there are indeed differences: In this paper, we propose a path planning algorithm based on the target location probability distribution updates using which will help the algorithm to find the target in the shortest amount of time. This has not been considered in the previous works. Furthermore, in addition to a detailed analysis of the problem, we conducted experiments to demonstrate the superior performance of our proposed algorithm in practice.

The contributions of this paper are as follows:

- An end-to-end setup for early fire detection or a SAR operation is considered where a UAV is looking for a target. The target's location is not known or is known only probabilistically. Using the target's location probability distribution, the proposed method minimizes the target detection time among the baseline approaches.
- We have developed a general framework to model and analyze the problem of finding a target through the shortest path when the location is probabilistically known. First, a Bayesian framework is utilized to estimate the target location. Then, using the sliding windowing technique for the path planning problem a suboptimal solution is obtained.
- Using ResNet, a target detection algorithm is designed with an accuracy of greater than 91% for the training and validation pictures taken from the existing datasets and images generated by the drone.
- Through simulation, the proposed path planning algorithm is compared with two benchmark algorithms and it is shown that the proposed algorithm significantly decreases the average time of the mission and the energy consumption of the drone in the two types of probability maps considered.
- Through experimentation, the proposed framework is applied to a fire detection scenario where it is shown that the proposed algorithm outperforms the benchmark algorithms in terms of time spent to find the target and the energy consumption of the drone.
- Our code and other relevant materials can be accessed online.[2]

The rest of the paper is organized as follows: in Section II, related works are provided. In Section III, a simplified scenario, a Naive algorithm, and the proposed system model are presented. In Section IV, the proposed algorithm and machine learning architecture are provided. Section V provides the simulation and experimental results, and Section VI concludes the paper.

II. Related Work

UAV-based target detection and target tracking have been increasingly attracting attention in civil applications [21], [22], [23], [24], [25], [26], [27]. Meanwhile, there have been numerous proposals for utilizing UAVs in SAR operations in recent years, from post-natural or human-made disasters such as earthquakes and explosions to offshore SAR operations in oil-rigs platforms and monitoring wildfire spread rate [28], [29], [30]. In the sequel, we will specifically review studies that have investigated problems involving UAVs in wildfire and SAR operations.

SAR and wildfire detection operations can be investigated from different points of view, e.g., object detection, optimal routing, optimal resource allocation, etc. Generally speaking, the majority of works in these scenarios have been allotted to develop efficient and accurate object detection algorithms, for example, see [3], [9], [31], [32], [33], [34], and [35] where different image datasets were used to train the learning algorithms.

A. Target Detection

In recent years, the use of UAVs for wildfire detection has garnered significant attention, e.g., [36], [37], [38], [39], and [40] where different methods of imaging and detection such as color and motion features, Infrared images, and smoke sensors along with different learning paradigms were employed. In [41], in order to use a smaller dataset and reduce the computational complexity, the authors proposed to use a pre-trained mobileNetV2 architecture to implement transfer learning. Recently, millimeter-wave radar sensing has been

[2]https://github.com/RushivArora/Incendium-Autonomae, and https://github.com/RushivArora/Incendium-Data

TABLE I
SUMMARY OF NOTATIONS

Parameter	Notation
UAV altitude	H
Width of projected area	w
Length of projected area	l
Total number of cells	M
Horizontal overlap	r_x
Vertical overlap	r_y
Target detection random variable in cell i	D_i
UAV's presence random variable in cell i	I_i
Missed detection probability	e_d
False alarm probability	e_f
Target detection random variable at time step t	B_t
Total target detection time	T
Sliding window length	W
Random variable of target existing in region l	R_l

employed for target positioning in [42]. Furthermore, a multi-UAV-based target positioning problem has been investigated in [43] where the target location is estimated based on the UAVs group measurements.

B. UAV Path Planning

In terms of UAV path planning [44] and [45] address minimal route design and energy consumption estimation, respectively. For multiple UAV collaborations, [46] proposes a layered SAR algorithm to minimize mission time and maximize rescues. Trajectory planning for persistent surveillance is discussed in [47] for fixed-wing drones, while [48] considers covertness in trajectory planning for mobile object surveillance with UAVs. Additionally, [49] and [50] address multi-robot task allocation problems for SAR missions using UAVs and unmanned ground vehicles. Path planning for a UAV to maximize the number of ground vehicles covered to be informed about an existing fire has been investigated in [51]. Finally, [18] utilizes a genetic algorithm to optimize a rectangular path for a patrolling UAV, [19] uses a Levy flight model to identify the target location, and [20] leverages the Pareto Monte Carlo Tree Search (PMCTS) to design the most informative exploratory searching path. We have conducted a performance comparison of our algorithm and the PMCTS developed in [52].

C. Coordination and Task Allocation

In wildfire management, a coordination problem was considered in [53] where UAVs were used along with unmanned ground vehicles (UGVs) to fight the fire front. Furthermore, investigating the optimal UAV coalition to fully cover the area, spectrum sharing plus cell assignment, and the optimal number of UAV and IoT devices for a maximum detection probability are the other problems that have been considered in wildfire management where the location of the fire is known [54], [55], [56], [57].

A list of notations and their definition used in the rest of the paper is presented in Table I.

III. Scenario and System Model

A. Simplified Settings and Intuition

In this part, the intuition behind the problem considered in this paper is elaborated. In fact, through a simplified scenario and a Naive algorithm, it is shown that there is a need to provide a general algorithm that is capable of efficiently finding the target when the location is stochastically known.

To this end, a scenario is considered where a target (e.g., a lost hiker or a fire spot) needs to be found within an area $S \subset \mathbb{R}^2$ partitioned into roughly equal cells. The location of the target known probabilistically is somewhere among the cells, i.e., its PDF is known or can be estimated through information sources. If no information is available, a uniform PDF would be assumed. A UAV is employed to fly over the region and search for the target based on the given PDF. The UAV uses its camera to monitor the area and detect the target. It is crucial that the target is found within the shortest amount of time, as any delay in finding the target could have irrecoverable and disastrous consequences. Thus, the most important goal is to minimize the *detection time*.

Intuitively, to find the target as soon as possible, one would be interested in visiting the cells with a higher probability sooner. However, this implementation is challenging as the search path must be a continuous path suitable for UAV's flying. Besides, the path must be chosen to be efficient in terms of energy consumption. Hence, an important question is whether visiting the cells simply based on their probabilities from the highest to the lowest one leads to an efficient path planning algorithm in terms of detection time or not. Through the following examples, a few insights are provided on the above question.

- **Simplified Scenario:**

Consider a drone is supposed to find a target in an area partitioned into cells denoted by index i where $i = 1, 2, \ldots, M$. The missed detection probability denoted by e_d is defined as the probability that the drone has reported no target while the target is indeed in the cell. We also define the false alarm probability as the probability that the object is reported as detected while it is not true and we show it by e_f. Initially, no false alarm is considered, i.e., $e_f = 0$. In this simplified scenario, the time required to be passed when the drone is traveling from its current cell to another one is not considered. In other words, the time amount passed from one cell to any other cell is exactly one unit of time regardless of the distance between the two cells.

The random variable that the target is in cell i is denoted by I. In other words, $p_i = P(I = i)$ denotes the probability that the target exists in cell i. Furthermore, without loss of generality, it is assumed that $p_1 \geq p_2 \geq, \ldots, \geq p_M$. The assumptions for the simplified scenario are summarized as below:

- One UAV is deployed for the operation.
- The probabilities of cells are of the form $p_1 \geq p_2 \geq, \ldots, \geq p_M$.
- The UAV simply visits the cells considering the probabilities order.
- No probabilities update is considered.

With this scenario and the idea that intuitively the most probable cells are selected first to search in, the UAV is assumed to choose the cells with respect to their probabilities. In this case, it's important to note that the probability of detection remains the same in every cell (e_d) and isn't affected by the other cells. In other words, it is independent of the other cells. With these assumptions, the UAV visits i-th cell at times $t_i = Mk+i$, where $k = 0, 1, \dots,$ and $i = 1, 2, \dots, M$. This is because the UAV will start over again to look for the target in the same order if it has not detected anything in the previous round and so on. If the amount of time spent by the drone to find the target is represented by T, with this scenario in hand, the average time is obtained in the following lemma.

Lemma 1. The average time of the simplified scenario is obtained as

$$E[T] = M\left(\frac{1}{1-e_d} - 1\right) + E[I], \tag{1}$$

where $E[.]$ denotes the expectation.

Proof: For the proof, first note that given $I = i$, the random variable T can be written as

$$T \mid (I = i) = M(Y-1) + i, \tag{2}$$

where $Y \sim Geometric(1-e_d)$. Now from the linearity of expectation, it is concluded that

$$E[T|I = i] = M\left(\frac{1}{1-e_d} - 1\right) + i, \tag{3}$$

and from the law of total expectation, $E[T]$ can be obtained as below

$$E[T] = E[E[T|I = i]] = M\left(\frac{1}{1-e_d} - 1\right) + E[I]. \tag{4}$$

□

Using Lemma 1, $E[T]$ can be obtained for the worst case scenario where $p_1 = p_2, \dots, p_M = \frac{1}{M}$. In fact, for the worst-case scenario, $E[T]$ is as follows

$$\begin{aligned} E[T] &= M\left(\frac{1}{1-e_d} - 1\right) + \frac{M+1}{2} && (5)\\ &= \frac{M(1+e_d)}{2(1-e_d)} + \frac{1}{2}. && (6) \end{aligned}$$

Equation (6) is actually an upper bound for $E[T]$, since given any distribution for I, $E[I]$ is bounded as below

$$1 \le E[I] \le \frac{M+1}{2}. \tag{7}$$

If $e_f \neq 0$, whenever a false alarm happens, it can be assumed that the cell is searched by a backup team and since it was a false alarm and nothing is found, an excess delay of Δ_f is added to the operation's time. Hence, given $I = i$ and $Y = k$, the false alarm delay denoted by Z_k is a Binomial random variable i.e., $Z_k \sim Binomial((k-1)(M-1) + i - 1, e_f)$. Therefore, $T = Z_k + (k-1)M$ and,

$$P(Y = k|I = i) = P(T = Z_k + (k-1)M + i|Y = k, I = i). \tag{8}$$

Finally, since Y and I are independent, the expectation of T is obtained as

$$\begin{aligned} E[T] &= E[E[T|Y, I]] && (9)\\ &= ((E[Y]-1)(M-1) + E[I] - 1)\Delta_f e_f \\ &\quad + (E[Y]-1)M + E[I] && (10)\\ &= \left(\left(\frac{1}{1-e_d} - 1\right)(M-1) + E[I] - 1\right)\Delta_f e_f \\ &\quad + \left(\frac{1}{1-e_d} - 1\right)M + E[I], && (11) \end{aligned}$$

where (10) is obtained by inserting $T = (Y-1)(M-1) + I - 1$ and taking the expectations with regard to Y and I, respectively, and (11) is obtained as $Y \sim Geometric(1-e_d)$. Hence, we note that the first term in (11) represents the delay added by the false alarm.

The simplified scenario has two main problems: First, it is not optimal in terms of $E[T]$ since at each time that it visits a cell and if nothing is detected, it chooses its next destination simply considering the same probability order. Whereas when the UAV does not detect the target in the cell with the highest probability, the probabilities of cells need to be updated according to the e_d. The second problem is that it does not consider a continuous trajectory. In fact, it virtually assumes that the UAV simply jumps from its current cell to the cell with the next highest probability which is not feasible in practice.

In this paper, these problems are addressed by proposing a path-planning method based on updating the probabilities after each observation.

Before going to the next section, another problem is presented which occurs when the probabilities are not updated but the amount of time required to detect the algorithm matters.

- **Naive Algorithm:**

Now assume that the PDF is a bimodal Gaussian distribution shown in 2(a). In the Naive algorithm, the drone visits the cells based on the cells' probabilities from the highest to the lowest one without a probability update. The assumptions for this scenario are listed below:

- One UAV is deployed for the operation.
- The probabilities of cells follow a bimodal Gaussian distribution
- The UAV simply visits the cells considering the probabilities order.
- No probabilities update is considered.

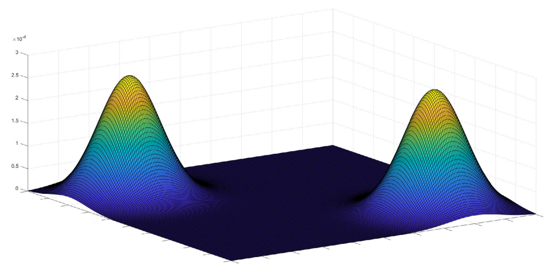
The first 300 visits are shown in 2(b). As can be seen, the drone spends most of its flight time between the two peaks and misses detecting other regions resulting in a highly inefficient path-planning algorithm.

It can be seen that neither the simplified scenario nor the Naive algorithm, although intuitive, can provide efficient performance.

In the next sections, the system model is presented in more detail.

B. Scenario and Assumptions

Suppose a drone is in charge of finding a missing person or a fire spot in the area. The area is divided into a grid of cells, each of which has a probability of the target existing in it. Note


(a) Probability distribution of the target's location.

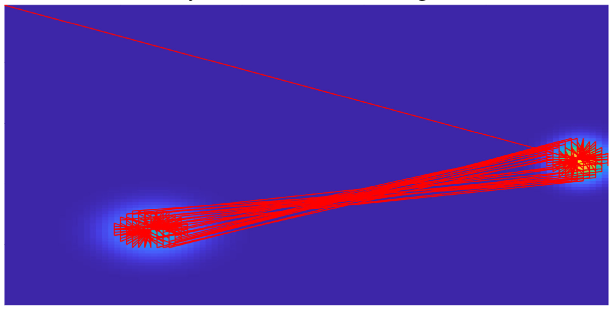

(b) UAV's path for first 300 visits.

Fig. 2. UAV's path based on naive algorithm for a bimodal gaussian PDF.

that this probability, in the case of finding the missing person, can be obtained from utilizing some interpretation of the missing person's behavior, and in the case of detecting the wildfire, can be obtained from the analysis of the area. In particular, by considering hiking trail maps in the former and noticing the vegetation coverage, and recognizing frequently visited areas in the latter, prior information can be obtained. In this regard, in [58] the authors estimate a risk/occupancy value for each cell of the area using different sources of data. Therefore, considering a PDF for the object location is potentially a valid assumption that leads to a more efficient detection algorithm. Nevertheless, if there is no prior information available, the initial probability is assumed to follow a uniform distribution across the region.

Now to efficiently detect a target, two collaborating algorithms are developed: The first algorithm is the object detection algorithm using ML. And the second one is the path planning algorithm, the input of which is the output of the ML algorithm.

It is assumed that the UAV visits one cell at each time step t. Hence it takes one time-step to go from one cell to one of its neighbors. It is also assumed that if detection is reported at a cell, the rescue team will search that cell to find the target. Note that the delay induced by the rescue team is not considered. If there is no target, a false alarm will be reported and the probabilities are updated accordingly. The detection process will continue until the UAV's energy is depleted or the target is found. Note that during the path planning algorithm, cells may be visited multiple times since the probabilities are updated regularly.

The problem considered in this paper could be thought of as a variation of the traveling salesman problem and is known to be NP-Hard [59]. Finding a globally optimal solution requires considering each possible action and each possible observation, which implies that the cost of a solution grows exponentially with the number of cells and actions. Hence, the sliding window technique with a length of W is used to find a sub-optimal path from the current location.

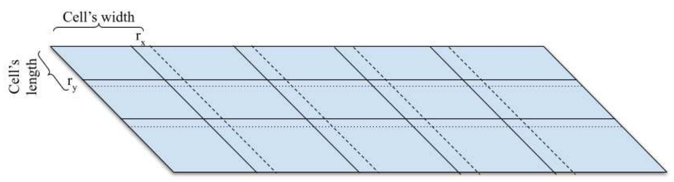


Fig. 3. Projected areas with overlaps. The centers of the rectangles are the path waypoints.

The general idea of the path planning algorithm is to search the most likely cells first; however, there is a trade-off between the time to reach a cell and the success probability of visiting that cell. The idea is that visiting the cell with the maximum probability may be less attractive than visiting the adjacent cells because it takes a long time to fly to the cell that has the maximum probability, but it may slightly improve the success rate. Therefore, we design a windowing-based path planning, the details of which will be discussed in Section IV-B.

C. Area Decomposition

In the first step, the area should be segmented through an approximate cellular decomposition. In this method, the area is divided into rectangles, and a point is placed in the center of each rectangle. The UAV is assumed to fully cover the cell when it reaches that particular point. The size of the rectangles is calculated from the field of view (FOV) of the camera.

The camera's FOV is the area covered by UAV's camera when it is flying at altitude H. The size (w, l) of the projected area can be obtained by the following equations

$$w = 2H.\tan\left(\frac{\alpha}{2}\right), \tag{12}$$

$$l = 2H.\tan\left(\frac{\beta}{2}\right), \tag{13}$$

where α and β are the vertical and horizontal angles of the camera, respectively.

Therefore, the area of interest is decomposed into a sequence of rectangles, which is denoted by C_i where $1 \leq i \leq M$. Then, the drone must be programmed to fly over each subregion's center. The complete path is then stored as a list of coordinates, called waypoints, and the drone moves from one waypoint to the next until it finds the missing object or it is out of energy. To cover the area completely, projected areas must overlap as shown in Figure 3. The amount of overlap can be chosen and can vary on each side. The horizontal and vertical overlaps are denoted as r_x and r_y, respectively.

At every UAV flight time step, the camera footprint of the search area is treated as a glimpse. This way, the search area can be discretized, and the UAV path-planning problem can be modeled as a discrete combinatorial optimization problem.

D. System Model

Since there is uncertainty and randomness in the operation (i.e., the location of the target is not predetermined, and could follow a known or unknown statistical distribution), it is essential to consider a probabilistic view. More specifically, let the underlying probability space be represented as $(\Omega, \mathcal{F}, P)$, where Ω, $\mathcal{F}$, and P represent the sample space, the event space, and the probability function, respectively.

The main assumptions for the system model are as below:

- One UAV is deployed for the operation.
- The probabilities of cells follow an arbitrary distribution model.
- The UAV visits the cells based on probability orders while considering the time constraint.
- Probability update is considered.

Now the first step is to determine the probabilistic map which combines all available information including past experiences. Each cell i where $i = 1, 2, \ldots, M$ in the probabilistic map has a probability of $P(I_i)$ which shows the probability that the target exists in that cell. Thus, with this definition, it is concluded that

$$\sum_{i=1}^{M} P(I_i) = 1. \tag{14}$$

It is assumed that first, the drone makes a new observation at time step t and then the ML algorithm determines whether the target is in the captured image or not. In this regard, D_i denotes the target detection random variable by the ML algorithm upon visiting the cell i.

Note that even if the drone captures an image of the target, it is possible that the object will not be detected, such possibilities are referred to as *missed detection probability* which is denoted by e_d and is defined as

$$P(D_i^c|I_i) = e_d, \tag{15}$$

where I_i shows the existence of the target in i-th cell and D_i^c shows the no-detection event in the cell i. Missed detection may be due to the inefficiency of the detection algorithm or a defect in the camera.

On the other hand, it is possible that the detection algorithm misidentifies the object in the image captured by the UAV. These possibilities are referred to as *false alarm probability* which is denoted by e_f and defined as

$$P(D_i|I_i^c) = e_f, \tag{16}$$

where I_i^c shows the nonattendance of the target in the i-th cell and D_i shows the detection of the target in the i-th cell.

In the next section, the probability updating framework and the path planning algorithm are provided.

IV. Probability Updating and Path Planning Algorithm

A. Updating Target Detection Probabilities

To update the detection probabilities for the cells, the Bayesian rule is utilized using the information obtained from observations. At each time step t, the conditional updated probability at cell i is obtained as below:

$$P(I_i|D_i^c) = \frac{P(D_i^c|I_i)P(I_i)}{P(D_i^c)} \tag{17}$$

$$= \frac{P(D_i^c|I_i)P(I_i)}{P(D_i^c|I_i)P(I_i) + P(D_i^c|I_i^c)P(I_i^c)}, \tag{18}$$

where (18) is obtained using the law of total probability. Now, using (15) and (16), one can obtain

$$P(I_i|D_i^c) = \frac{e_d p_i}{e_d p_i + (1-e_f)(1-p_i)} \tag{19}$$

$$= \frac{e_d p_i}{b_i}, \tag{20}$$

where $p_i \triangleq P(I_i)$ and $b_i \triangleq e_d p_i + (1-e_f)(1-p_i)$. Similarly, the probability map is updated for other cells, i.e., $1 \le j \le M$ where $j \neq i$, as below

$$P(I_j|D_i^c) = \frac{P(D_i^c|I_j)P(I_j)}{P(D_i^c)} \tag{21}$$

$$= \frac{(1-e_f)p_j}{b_i}. \tag{22}$$

Therefore, after visiting the i-th cell and conditioned on D_i^c, from (20) and (22), we conclude that the probability of each cell are updated as below:

$$P(I_j) = \begin{cases} p_i \dfrac{e_d}{b_i} & j = i \\ p_j \dfrac{1-e_f}{b_i} & j \neq i. \end{cases} \tag{23}$$

B. Windowing-Based Path Planning Algorithm

Now the details of the proposed path-planning algorithm are provided in the sequel. The goal of this algorithm is to minimize the average time of finding the target denoted as $E[T]$. Assume that B_t denotes the target detection at time step t. Therefore, f_t defined as the probability that the target is detected for the first time at time step t, can be obtained as follows

$$f_t = \left(1 - P\left(B_t^c|B_{1:t-1}^c\right)\right) \prod_{j=1}^{t-1} P\left(B_j^c|B_{1:j-1}^c\right). \tag{24}$$

Note that in (24), the term $\left(1 - P\left(B_t^c|B_{1:t-1}^c\right)\right)$ is the probability that the target is detected at time step t, while the rest is the probability that the target has not been detected until time step t. Assuming no false detection is reported from the ML algorithm, the expectation of the number of steps required to detect the target can be obtained as

$$E[T] = \sum_{t=1}^{\infty} t.f_t, \tag{25}$$

where $t \in N$ is the number of time steps to detect the target for the first time.

Note that this problem is a variation of the traveling salesman problem and is known to be NP-Hard [59]. Finding a globally optimal solution requires considering all the possible actions and observations, which implies that the cost of a solution grows exponentially with the number of cells and actions. Rather than finding a globally optimal path, the sliding window technique with length W is used to find the sub-optimal path from the current location. It should be noted that in order not to compromise the global optimality of

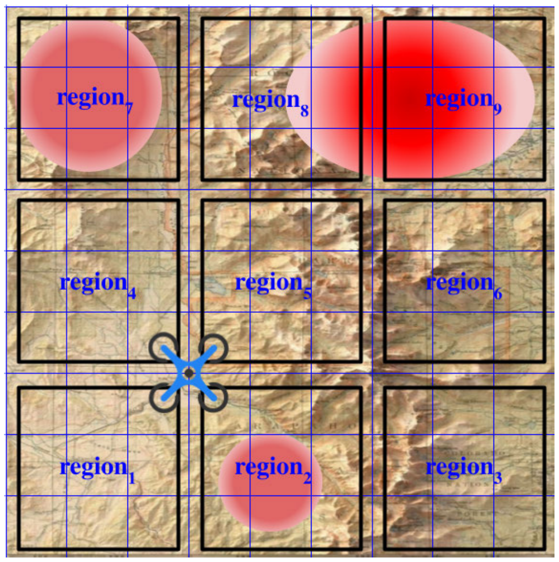


Fig. 4. Projected areas with overlaps. Centers of rectangles are the path waypoints.

trajectory, after visiting each cell and updating the probability map, the proposed algorithm again finds a sub-optimal path with length W from the current location. In other words, it is desirable to visit each cell of the window in a way that minimizes the expected time of the target detection by minimizing $E[T]$.

Note that if the target exists in the cells with almost an equal probability, it is better to visit the cells in a zigzag manner instead of prioritizing visiting the cell with the maximum probability. In this case, it is easy to realize that the proposed algorithm would take a long time while only improving the success probability slightly.

In order to make a trade-off between the time to reach a cell and the success probability of visiting that cell, W by W cells are aggregated to form non-overlap regions. Therefore, denoting by $P(R_l)$ the probability of the target existing in the region l, $P(R_l)$ is obtained as

$$P(R_l) = \sum_{k=1}^{W^2} p_{((l-1)W^2+k)}, \tag{26}$$

where $1 \leq l \leq \lceil \frac{M}{W^2} \rceil$ shows the number of regions. Note that (26) is obtained by simply considering the probabilities associated with the individual cells comprising a region as the regions are non-overlapping.

The distance between the regions R_l and R_k denoted by d_{R_l,R_k} is obtained from the distance between the centers of these two regions. Figure 4 shows an area of interest partitioned to 9 regions where each region includes $3 \times 3 = 9$ cells, i.e., $W = 3$.

Let $R_{\max}$ denote the region with the highest probability among all the regions. In other words,

$$R_{\max} = \arg \max_{l \in [1:\frac{M}{W^2}]} P(R_l). \tag{27}$$

Furthermore, let $R^{\text{local}}_{\max}$ denote the region with the highest probability among the adjacent regions of the current one written as below

$$R^{\text{local}}_{\max} = \arg \max_{l \in A(R_c)} P(R_l), \tag{28}$$

where R_c denotes the current region and $A(R_c)$ is the set of adjacent regions for the current region. Finally, let $R_p \in A(R_c)$ indicate the region located in a straight line between R_c and $R_{\max}$. Now the next region, denoted by R_n is chosen as below:

$$R_n = \begin{cases} R_{\text{p}} & \dfrac{P(R_{\max})}{P\left(R^{\text{local}}_{\max}\right)} > \dfrac{d_{R_c,R_{\max}}}{d_{R_c,R^{\text{local}}_{\max}}} \\ R^{\text{local}}_{\max} & \text{Otherwise} \end{cases}. \tag{29}$$

Note that by the next region, it is meant the next region chosen as the next destination that the UAV is supposed to fly to. It is obvious that in order to fly to the next region, the UAV first may visit some middle cells in the subsequent time steps.

Intuitively, with this comparison, it can be determined whether flying to the region with the maximum probability is attractive given the time it takes to get there or not. Therefore, in this step, it is decided which region should be visited next. This step will be repeated after W^2 observations.

After selecting the region, the cells' order is investigated in the selected region. If the cell index chosen to be visited at time t is denoted by o_t, the next cell index is denoted by o_{t+1} which is an adjacent cell of the current cell, o_t, and can be selected as below

$$o_{t+1} = \arg \min_{a \in A(o_t)} E[T_a], \tag{30}$$

where $A(o_t)$ is the set of adjacent cells for the current cell. Furthermore, $E[T_a]$ is obtained from the following equation

$$E[T_a] = P_{T|D} E\left[T|D_{a:a+W}\right] + P_{T|D^c} E\left[T|D^c_{a:a+W}\right] \tag{31}$$

$$= \sum_{i=1}^{W} t_i p_{a+i} + (M - W)\left(1 - \sum_{k=1}^{W} p_{a+k}\right), \tag{32}$$

where (31) is obtained from the law of total expectation in which $P_{T|D}$ and $P_{T|D^c}$ are the probabilities and $E\left[T|D_{a:a+W}\right]$ and $E\left[T|D^c_{a:a+W}\right]$ are the corresponding expectations. Also, in (32), M is the maximum time to visit all the cells in order. Therefore, one of the adjacent cells of the current cell which minimizes (31) is chosen. Intuitively, this implies that the cells with higher probabilities are visited more promptly.[3] Algorithm 1 represents the proposed path-planning method for target detection.

Also, Figure 5 demonstrates the block diagram for Algorithm 1.

Lemma 2. Given there is an object in S and assuming $e_d + e_f < 1$, Algorithm 1 converges to locate the object.

Proof: Assume that there is an object in S and $e_d + e_f < 1$. We consider a cell $i = k$ with the object in it with initial probability p_k and cells without the object, i.e., $i \neq k$ with probability p_i. Now we assume that the

[3]It is worth mentioning that there might be obstacles in the path to the next cell determined by Algorithm 1. In this situation, collision avoidance has been taken care of in the implementation phase where the UAV uses its sensors to avoid accidents with obstacles.

Algorithm 1 Path Planning

```
1: function PATHPLANNING($P, w, O, m$)
2:     Inputs:
       $P$ probabilistic map
       $w$ is window size
       $O$ start position
       $m$ is the location of the target
3:     Output:
       Time to find the target
4:     Construct non-overlapping regions by each $W \times W$ cells; call these regions: $R_1, R_2, \ldots, R_Z$, $Z = \frac{M}{W^2}$
5:     $current.region = O$
6:     while no target is found do
7:         for $W^2$ observations do
8:             Among all straight paths from $o_t$, select $o_{t+1}$ that minimizes Eq. (31)
9:             $observation$ = Algorithm 2$(o_t, o_{t+1})$
10:            if $observation$ $!= 0$ then
11:                Report to ground control station
12:            end if
13:            Update probability map using Eq. (23)
14:        end for
15:        Find the highest probability of all regions $R_{\max}$
16:        if $\frac{P(R_{\max})}{P(R^{\text{local}}_{\max})} > \frac{d_{R_c,R_{\max}}}{d_{R_c,R^{\text{local}}_{\max}}}$ then
17:            $R_n = R_p$
18:        else
19:            $R_n = R^{\text{local}}_{\max}$
20:        end if
21:    end while
22: end function
```

UAV has visited cell $i \neq k$ and reported no detection. In this case, the cells' probabilities are updated according to the Bayesian inference in (23). Hence, we have $P(I_i) = p_i \frac{e_d}{b_i}$, and $P(I_k) = p_k \frac{1-e_f}{b_i}, i \neq k$. Note that since $e_d + e_f < 1$, we obtain $\frac{e_d}{b_i} < 1$ and $\frac{1-e_f}{b_i} > 1$ by which over time the former weakens $P(I_i)$ and the latter amplifies $P(I_k)$. Hence, there is an ϵ where $\frac{P(I_i)}{P(I_k)} < 1-\epsilon$, as $T \to \infty$. Therefore, considering a Markov chain with two states $i = k$ and $i \neq k$, and as the algorithm chooses the cell with the higher probability, the UAV visits $i = k$ with a positive probability. In other words, the $i = k$ state is a recurrent state that is visited infinite times, and at some point, the object is detected since the probability of detection becomes a Geometric random variable with parameter $1 - e_d$. □

Note that for the proof of Lemma 2, we consider the worst-case scenario where $T \to \infty$. However, we have shown through simulation and experimentation that the algorithm ends up finding the object in a reasonable running time.

So far, only the path planning algorithm has been developed for which the inputs of the making decisions originated from the ML algorithm. Therefore, the ML algorithm is developed in the next section.

C. Machine Learning Architecture

Generally to solve intelligent decision-making problems, deep neural networks are used since they not only can learn patterns from the raw data themselves but also are able to learn the features automatically by adding more layers in order to learn more complex features. However, they suffer from the vanishing gradient problem [61], where the gradients become vanishingly small when training using gradient-based methods and back-propagation. In this regard, Residual Neural Network (ResNet) introduced by He et al [62], is a class of deep neural networks built for dealing with the vanishing gradient problem, especially for image classification. ResNets approach this problem by adding shortcut connections that skip one or more layers, such that the input of the skipped layers is added to the output of the skipped layers. As a result, it has obtained a 3.57% error on the ImageNet test set as well as won first place in the ImageNetdetection, ImageNet localization, COCO detection, and COCO segmentation in the 2015 ILSVRC and COCO competitions [63], [64]. Therefore, this paper utilizes a ResNet architecture, details of which are provided in the sequel.

1) Data: Experiments are conducted for fire and missing person detection in snow. In the case of fire detection, the dataset consists of 2646 images divided into 3 classes: *Fire*, *Smoke*, and *Spare*. Spare is considered any other object that the UAV might encounter such as trees, houses, and the ground. Out of 2646, 987 are the Fire, 781 are the Smoke, and 818 are

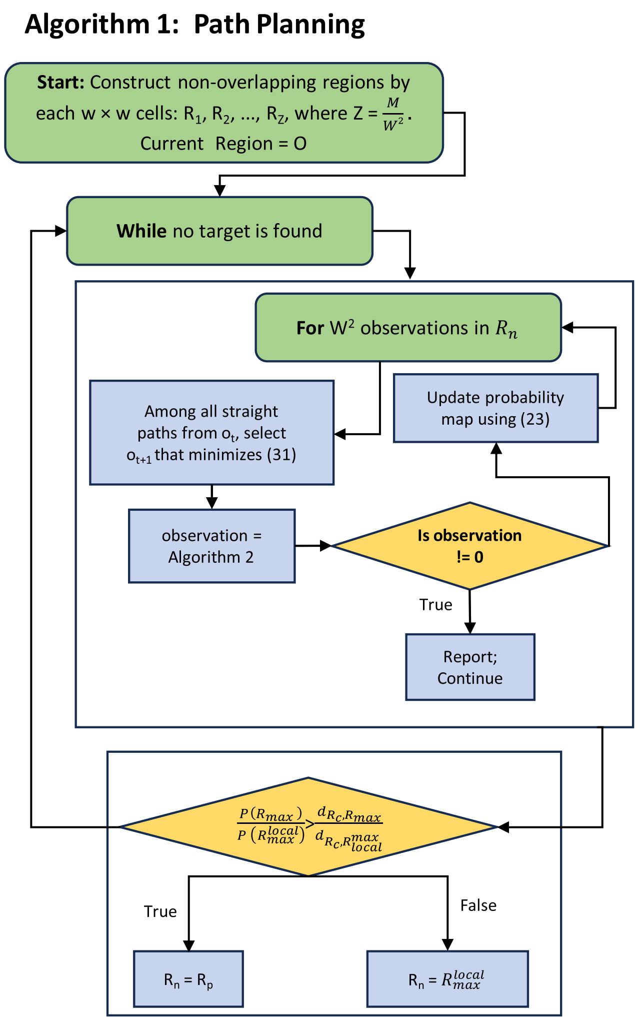

Algorithm 1: Path Planning
Start: Construct non-overlapping regions by each w × w cells: $R_1, R_2, \ldots, R_Z$, where $Z = \frac{M}{W^2}$. Current Region = O
While no target is found
For W^2 observations in R_n
Among all straight paths from o_t, select o_{t+1} that minimizes (31)
Update probability map using (23)
observation = Algorithm 2
Is observation != 0
True
Report; Continue
$\frac{P(R_{max})}{P(R^{local}_{max})} > \frac{d_{R_c,R_{max}}}{d_{R_c,R^{max}_{local}}}$
True
False
$R_n = R_p$
$R_n = R^{local}_{max}$

Fig. 5. Block diagram of algorithm 1.

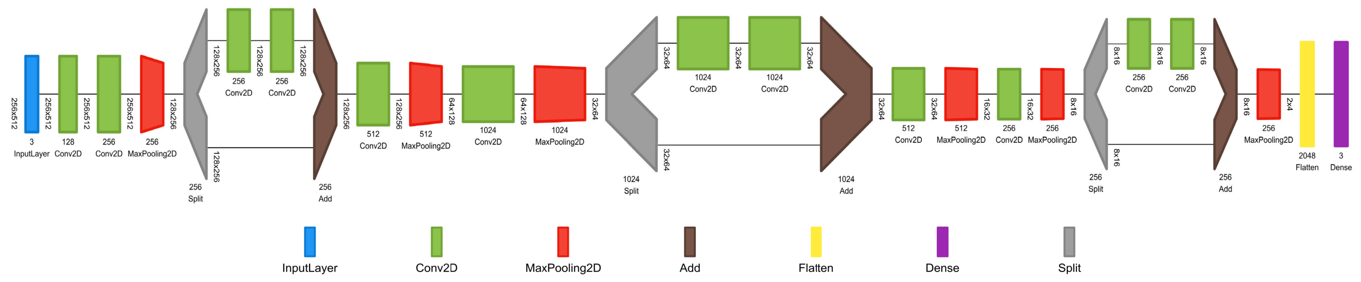


Fig. 6. The architecture of the ResNet. Visualization has been created using [60].

Spare. In the case of the lost hiker detection, the dataset consists of 1400 images divided into 2 classes: a *Snow-Covered Victim*, and *Spare* (any unwanted object such as trees, houses, and snow-covered ground). Out of these 1400 images, 650 are the Snow-Covered Victim and the rest are Spare. In both datasets, roughly 15% of the images are taken by the drone at heights between 15-30 meters during some preliminary flights to help the model generalize better to the real world. The rest of the images are carefully and manually selected from online repositories and available datasets [65], [66] in a way that they best represent different classes of the data in the real-world setting as well.

Furthermore, for the validation procedure, an additional of 300 images are taken by the drone at a second location. Finally, testing is performed at two other new locations. In other words, different locations are used for the three datasets - training, validation, and testing - to ensure that the experiment is authentic and that the model is truly generalizing rather than overfitting to the training or validation data.

2) Model: As discussed earlier, the model used in this paper is a ResNet, the details of which are highlighted in Figure 6. The designed ResNet has 6 convolution and 3 residual blocks, or equivalently 12 convolutional layers, 12 activation layers, and 6 pooling layers. Each convolutional block consists of a sequential and ordered arrangement of a convolutional layer, a batch normalization layer, rectified linearity activation (ReLU) layer, and a maximum pooling layer. A residual block, on the other hand, consists of two sequential convolutional blocks (minus pooling) that are added to the input of the first convolutional blocks.

3) Training & Validation: The model is trained for 8 epochs using an Adam Optimizer and a learning rate scheduler with the Cross-Entropy Loss Function. A detailed list of training parameters has been provided in Table II. It is worth mentioning that a lower learning rate is preferred due to the nature of the data as a larger learning rate would cause jumps around the global minima.

Furthermore, since a small data set is used, several data augmentation strategies such as manual feature extraction, flipping, cropping, and zooming is applied during the training process. In fact, while larger datasets are preferred for generalization, the data augmentation techniques have shown to be successful in feature extraction and synthetically increasing the size of the data.

TABLE II
SUMMARY OF TRAINING PARAMETERS

Parameter	**Value**
Image Size	$256 \times 512 \times 3$
Epochs	8
Optimizer	Adam
Learning Rate	0.0005
Weight Decay	10^{-4}
Gradient Clipping	0.1
Feature Extraction	Z-normalization
Batch normalization	Yes
Dropout p	0.3
Fire Validation Acc.	0.932
SAR Validation Acc.	0.94
Time to Inference (IPhone X) including drone-device latency	3 s

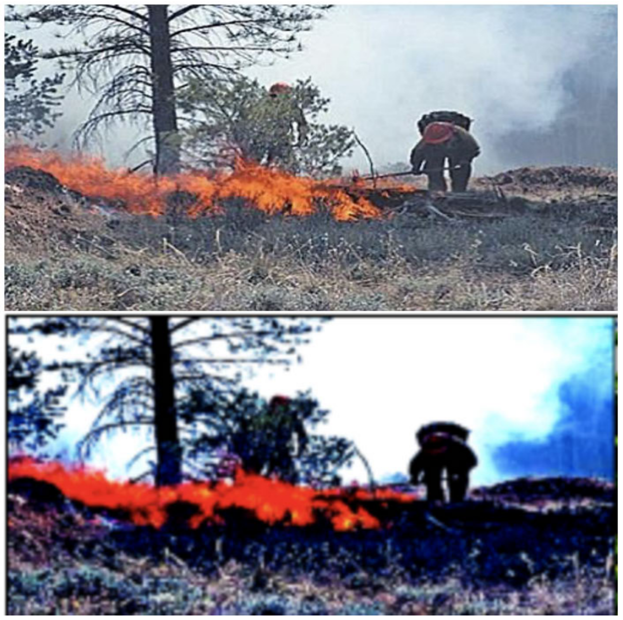


Fig. 7. Data normalization: The image on top represents the original image, while the image below shows the normalized version.

The training process includes the following functions:

– Data normalization is used for training, validation, and the implemented model on the drone. The image tensors are normalized by subtracting the mean and dividing by the standard deviation, both of which are calculated separately. Data normalization highlights the essential elements of the input image, making it easier for machine learning classification. Figures 7 and 8 shows a sample of the data normalization.

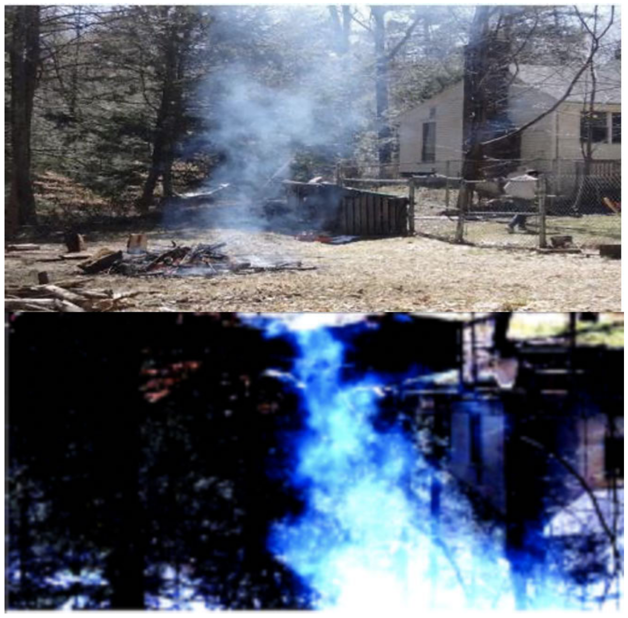


Fig. 8. In addition to Data normalization, Data augmentation is applied to the training data in order to increase new images in each epoch.

- Data Augmentation. Note that the size of the dataset is smaller than many of the widely available datasets. Hence, in order to avoid over-fitting and ensure that the model generalizes well to the real world, the apparent size of the dataset is increased using data augmentation. To this end, the images are 1) *padded on the right*, 2) *randomly cropped*, and then 3) *flipped with a fixed probability*. An example of data augmentation can be seen in Figure 8.
- Batch normalization: The data is initially normalized by subtracting the mean and dividing by the standard deviation (Data normalization). Batch normalization takes the same principle and applies it to each layer in the neural network in order to further extract the features in the outputs of each layer, before feeding it into the next layer [67].

Besides, the machine learning model is trained for 30 seeds and the validation accuracy, validation loss, and training loss are recorded for each seed. The average validation accuracy at the end of the 30 seeds is 93.8% while the average training and validation losses are 0.5857 and 0.6061, respectively. Average metrics vs epoch, with the standard deviation for confidence intervals, are represented in Figure 9.

Table III, demonstrates the results collected on validation data. In this table, Fire has the lowest accuracy and highest precision. This is due to the size of the fire when conducting the validation tests, where for safety, it has been kept small.

Besides, it can be seen that the Fire class has the highest precision due to the fact that it has extremely few false alarms. The Smoke class has fewer false alarms as well where it has a precision of 0.91. The Spare class, on the other hand, has the lowest precision since many misclassifications in Fire testing are classified as Spare when the flames are not visible, giving rise to a larger number of false alarms. The highest precision for Fire and Smoke is advantageous as it is indicative of a

TABLE III

PRELIMINARY CLASSIFICATION METRICS FOR VALIDATION DATA. THE LOW ACCURACY OF FIRE IS ATTRIBUTED TO THE SMALL FIRE SIZE SELECTED FOR SAFETY REASONS. IT HAS BEEN INCREASED SLIGHTLY LATER IN TESTING PRESENTED IN TABLE IV

	Precision	Recall	F1-score	Support
Fire	0.99	0.74	0.85	120
Smoke	0.91	0.95	0.93	120
Spare	0.82	0.99	0.90	120

TABLE IV

CLASSIFICATION METRICS FOR TESTING DATA

	Precision	Recall	F1-score
Fire	0.99	0.93	0.96
Smoke	0.97	0.95	0.96
Spare	0.92	0.99	0.96

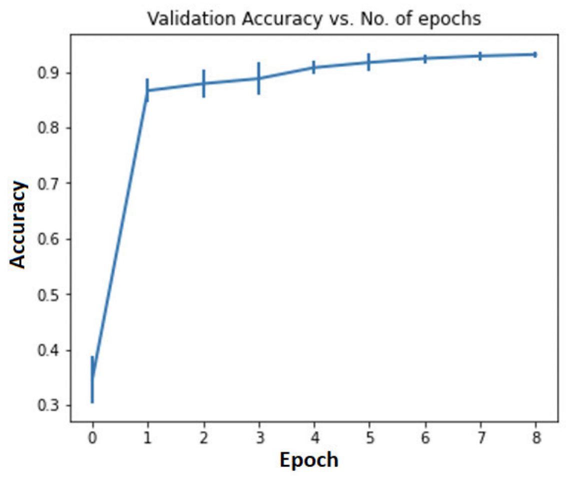
low number of false alarms. The Fire class has a lower recall, while the Smoke and Spare classes have a significantly higher recall which implies that the Fire class has a larger number of missed detection while the Smoke and Spare classes do not. The size of the fire has been increased slightly in the experiments conducted for testing later presented in Table IV.

Finally, to find the ideal stopping point of training, an analysis of the misclassified images is conducted. It has been observed that the misclassified images consisted only in case of outliers, which particularly occurs when the different classes share features. For example, cloud and smoke, or cloud and white houses and snow share features. In this case, training beyond involves overfitting to these outliers in the training dataset which in turn leads to a decrease in the accuracy of the validation data. Therefore, it is concluded that the efficient stopping point is 8 epochs.

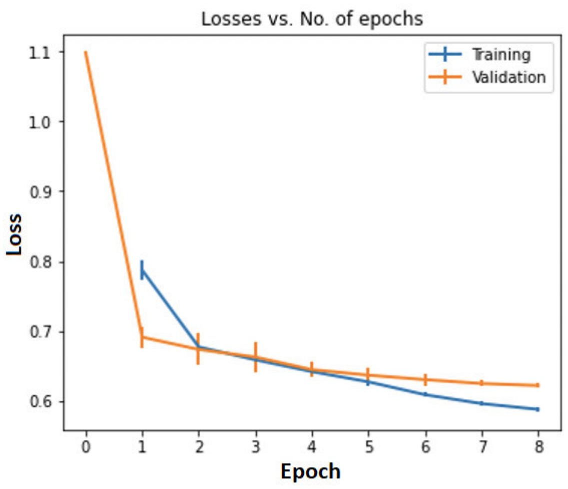
4) Testing: In the validation process, the lower recall for the Fire class happens because it is difficult to view a campfire because it is covered by logs and sticks. This is in contrast to viewing the flames at an angle at which the flames are more visible. To deal with the assumption that the lower accuracy of the Fire is caused due to less visible flames, a significantly larger controlled burn has been conducted in open burning season (ending on May 1) and it is observed that an accuracy of 93% is achieved for the Fire class with the metrics as seen in Table IV.

In this table, due to the visible flames, it can be seen that the metrics are improved. With a higher accuracy reported with more visible flames, the prediction is that on an early forest fire, which can be essentially larger than a campfire of 1 ft tall, the algorithm even performs more desirably.

As mentioned earlier, to ensure the authenticity of the experiment and avoid overfitting the training or validation data, tests are conducted on new fire spots different from those of training and validation. This also allows the measurement of the generalization performance of the machine learning method, presented in Table IV. Depending on the availability and nature of the area surrounding the fire spot, a disturbance level is added to each testing image that measures how much disturbance is being added from the surroundings. For instance, when testing for Fire or Smoke, the drone



(A) Validation accuracy vs epoch number



(B) Training and Validation loss vs epoch number

Fig. 9. Training and Validation for the model over 30 seeds. The average values are plotted while the standard deviation is used for providing confidence intervals. The plots included are (a) Validation accuracy vs epoch number and (b) Training and Validation loss vs epoch number.

TABLE V
ACCURACY OF TESTING DATA FOR SAR IN SNOW

	With Snowflakes	Without Snowflakes
CNN	0.582	0.731
ResNet	0.897	0.943

camera is adjusted to display half or full of the surrounding tree to confuse the machine learning algorithm to determine whether the image is Smoke or Spare. Furthermore, this testing has been conducted for varying amounts of environmental disturbance.

It is worth mentioning that the largest distance for detecting Fire has been measured to be 20.2 meters (66.3 feet) on the firepit and 28 meters (91.86 feet) on the controlled fire, while the highest distance for detecting Smoke has been 37 meters (121.39 feet).

Table V presents the test results for the SAR operation considering two cases: with snowflakes and without snowflakes. In the former, falling snow is superimposed into the images to add noise and further test the model's limits. Also, a comparison of ResNet and CNN has been provided in this table. In this setting, in order to add diversity to the data and further test the model, participants with five different clothes colors were involved. The testing also involved the participants being in different locations in the image to even being partially present which the model has shown to be able to detect. Furthermore, various amounts of snow on the ground have been considered for the model testing, ranging from being completely covered in snow to partial coverage. Finally, the largest height for detecting a person has been measured to be 24 meters (78.74 feet) with snowflakes and 27 meters (88.58 feet) without snowflakes.

The entire machine learning algorithm is shown in more detail in Algorithm 2. First, the UAV automatically flies from the current location to the next location. Next, it will activate the camera in shooting mode, resize the captured image, and retrieve the resized image. Finally, the ML recognition runs on the captured image and returns the detection class to Algorithm 1.

Algorithm 2 Drone and ML Action

```
1:  function DRONE AND ML ACTION($c, n$)
2:      Inputs:
        $c$:= current location of the UAV
        $n$:= next location of the UAV
3:
4:      Output:
        Target detection
5:      Move UAV from location $c$ to location $n$
6:      if Fetch camera then
7:          $image$ = captured image from UAV
8:          Run ML recognition($image$)
9:          return 0, 1, or 2 for $Spare$, $Fire$, or $Smoke$
10:     else
11:         return Fetch camera error
12:     end if
13: end function
```

Figure 10 shows the block diagram of Algorithm 2.

D. Discussion on the Complexity Analysis

Before diving into our complexity analysis, we would like to explain how the algorithm works. The algorithm begins with the initialization of cell probabilities. The primary objective is to optimize between two conflicting goals: the time taken to reach a cell and the likelihood of successfully visiting it. To this end, the algorithm selects regions in a way that it decides between targeting the region with the highest success probability, factoring in the time required to reach it, or directing to adjacent regions, considering the lower probability of success and shorter time required to reach that region. This decision-making process, along with the continuously updated probability map, continues to direct the next steps.

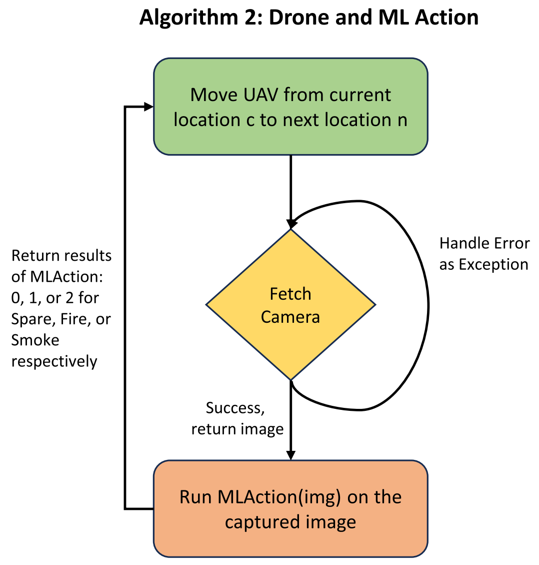

Algorithm 2: Drone and ML Action
Move UAV from current location c to next location n
Fetch Camera
Handle Error as Exception
Success, return image
Run MLAction(img) on the captured image
Return results of MLAction: 0, 1, or 2 for Spare, Fire, or Smoke respectively

Fig. 10. Block diagram of algorithm 2.

Fig. 11. DJI Mavic 2 Zoom with a 4k camera.

Now, for the complexity analysis, we note that the proposed algorithm starts with M cells' probability. In order to make a tradeoff between the time to reach a cell and the success probability of visiting that cell, W by W cells are aggregated to form non-overlap regions. The first step of the proposed algorithm is to find $R_{\max}$ which represents the highest probability among all the regions. Finding the maximum probability and sorting it takes time proportional to $\frac{M}{W^2}+log(\frac{M}{W^2})$. However, since this only needs to be done once, it does not significantly affect the overall computational complexity. After finding $R_{\max}$, the algorithm should select the next region to visit using Eq. (15). In the worst-case scenario, we would need the UAV to visit every region and there are $\frac{M}{W^2}$ regions in total. This would result in a total of $\frac{M}{W^2}$ observations of regions and a comparison operation according to Eq. (15). In each region, we have to select the best adjacent cell using Eq. (17) which takes time proportional to $9W$. Then, we make a new observation and run ML algorithm. Let t_d denote the time needed to process the image using the ML algorithm. After W^2 observations, we should update the probability map and select the next region. Multiplying the probability map by the observation/detection results also takes time proportional to $\frac{M}{W^2}$. That brings the overall complexity of $\frac{M}{W^2}$ repeats of observations and updating the probability map to $O((\frac{M}{W^2})^2)$ if the time complexity of updating the map is greater than t_d+9W.

V. Simulation and Experimental Results

A. Camera Model

In order to obtain the numerical results through simulation and experiments, a DJI Mavic 2 Zoom with a 4k camera shown in Figure 11, has been used.

B. Power Consumption

One critical issue of UAV operation is the limited onboard energy of UAVs, which renders energy-efficient UAV-based operations particularly important. The UAV energy consumption is in general composed of two main components, namely the communication-related energy and the propulsion energy. Depending on the size and payload of UAVs, the propulsion power consumption may be much more significant than communication-related power. To this end, proper modeling for UAV propulsion energy consumption is crucial. For a rotary-wing UAV with speed V, the propulsion power consumption can be expressed as [68]:

$$P(v)=P_0\left(1+\frac{3v^2}{U_{tip}^2}\right)+P_i\left(\sqrt{1+\frac{v^4}{4v_0^4}}-\frac{v^2}{2v_0^2}\right)^{\frac{1}{2}}+\frac{1}{2}d_0\psi d_A v^3, \tag{33}$$

where P_0 and P_i are constants representing the blade profile power and induced power in hovering status, respectively. P_i depends on the aircraft weight, air density ψ, and rotor disc area d_A, as specified in [68]. Also, U_{tip} denotes the tip speed of the rotor blade, v_0 is known as the mean rotor-induced velocity in hovering, and d_0 and s are the fuselage drag ratio and rotor solidity, respectively.

Therefore, with a given trajectory $q(t)$ where $q(t)\in R^2$ and $0\le t\le T_m$, the propulsion energy consumption can be expressed as

$$E(T_m,q(t))=\int_0^{T_m}P(||v(t)||)dt, \tag{34}$$

where $||v(t)||$ is the instantaneous UAV speed.

C. Simulation Results

A 2×2 km^2 search area is considered to investigate the amount of time required to find the target. The algorithm is evaluated using two different probability distributions: Gaussian mixture and Gaussian-Uniform mixture distributions [69]. A drone starts flying from the left-bottom corner of the search area. Table VI shows the average time to find the missing target for three different algorithms. In fact, the proposed algorithm is compared with two different algorithms: Zigzag algorithm which is achieved via back and forth trajectories in the search area [70]. And, the algorithm used in [71] in which it approximates the probability distribution map using

TABLE VI
AVERAGE TIME (SECONDS) TO FIND THE MISSING TARGET

	Gaussian mixture distribution	Gaussian–Uniform mixture distribution
Zigzag algorithm	9800	10100
PMCTS [52]	2090	9900
[71]	2160	11110
Proposed algorithm	1960	8090

TABLE VII
ENERGY CONSUMPTION (KJ) TO FIND THE MISSING TARGET

	Gaussian mixture distribution	Gaussian–Uniform mixture distribution
Zigzag algorithm	2487	2564
PMCTS [52]	542	2474
[71]	564	2706
Proposed algorithm	503	2132

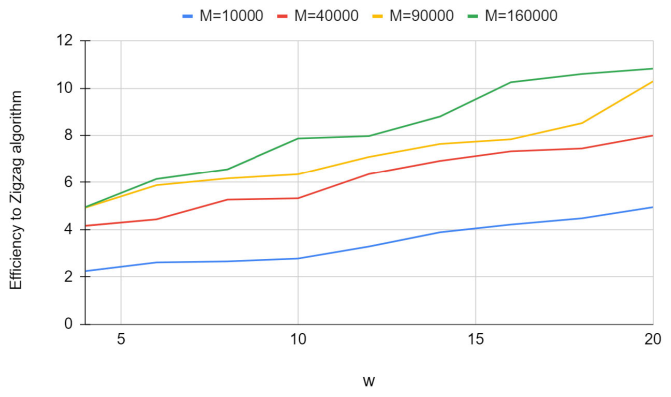


Fig. 12. The effect of W and M in the path planning algorithm.

the Gaussian mixture model (GMM) and plans UAV motions heuristically using Gaussian probability density functions. As can be seen, the proposed algorithm is able to find the target in a shorter time in both distributions. Specifically, in the case of Gaussian mixture distribution, it outperforms the Zigzag algorithm 5 times faster.

The energy performance of the proposed algorithm is evaluated and compared to two other algorithms as well. To this end, the power consumption model and parameters shown in Equation (33) are used for all cases and the efficiency is defined as the ratio of energy corresponding to the benchmark algorithm to that of the proposed algorithm.

Figure 12 shows the effect of changing W and M in the path planning algorithm where the location of the target follows a Gaussian mixture distribution. The vertical axis shows the performance ratio of the proposed algorithm to the Zigzag algorithm. As can be seen, the performance ratio of the proposed algorithm with respect to the Zigzag algorithm improves with increasing W.

D. Experimental Results

For the experimental results, extensive experiments have been conducted on the system to investigate end-to-end functionality, detection accuracy, and algorithm efficiency. In a real experiment, a scenario where the target is a fire spot is considered to be found within a large field of 234000SF. The location of the fire follows a Gaussian-uniform mixture distribution. We utilized Gaussian-Uniform distribution due to its closer alignment with practical scenarios.

TABLE VIII
PARAMETERS VALUES IN EXPERIMENTAL EVALUATIONS

Parameter	**Value**
Drone model	DJI Mavic 2 Zoom
Target	A firepit
Target distribution	Gaussian-Uniform
S: Area	234000SF
H: Flight height	25 m
v: Flight speed	20 $\frac{m}{s}$

TABLE IX
EXPERIMENTAL RESULTS

	Time to complete(S)	Energy consumption(kJ)
Zigzag algorithm	1024	230.4
PMCTS [52]	1008	233.1
[71]	1060	244.8
Proposed algorithm	920	213.4

The drone flies at an altitude of 25m from the ground with a speed of $20\frac{m}{s}$. The boundary of the flight is selected based on the search area. A summary of experiments parameters are shown in Table VIII. The instruments of the UAV Control Module are a phone and a UAV remote-control unit. With DJI Go App and DJI SDK, one can set the flight missions of the UAV and control its flight state. The search path for the flight mission is generated using the probability map, and the drone flies to each point as expected.

The limited energy and computational resources of the drone require the utilization of a remote controller for movement directives. Furthermore, it is required that the ground controller receives the detected target's information in a timely and convenient manner so that it can take the corresponding actions. Thus, the real-time transmission of the associated information is crucial. Real-time transmission is realized both in the UAV control algorithm and the detection algorithm. With DJI Go App and DJI SDK, we can set the flight missions of the UAV and control its flight state. In addition, the real-time video can be displayed on the phone.

At each point of operation, the images are captured and analyzed, then the probability map is updated based on the new observation. The search algorithm finds the next point based on the updated map. Table IX shows the average time to find the target and the corresponding energy consumption for our proposed algorithm and the Zigzag algorithm and the one proposed in [71]. There have been 25 experiments for each case. The average time was evaluated by measuring the amount of time spent to detect the target successfully. The energy consumption during the target detection mission was evaluated by measuring the UAV battery level during the mission and at the end of the mission.

VI. CONCLUSION

In this paper, a general framework was proposed for the problem of object detection in which path planning and machine learning models are investigated. In particular, scenarios were considered in which objects such as a lost hiker in the snow or a fire spot in a given area are intended to be detected. Using a probability map, a path planning algorithm was proposed based on the Bayesian inference so that the shortest path leading to the object detection is designed. Along with the path planning algorithm, a residual neural network was utilized to capture the object. Since in the considered scenario, the object is a person camouflaged in the snow or a fire spot in its initial levels, a new dataset was developed besides the existing ones, for training, validation, and testing of the neural network. The results are verified through simulation and experiments. It has been shown that the proposed method outperforms the existing ones in terms of the average time spent to find the object and the energy consumption of the drone. The present work provides a proof of concept for SAR and fire detection operations. Therefore, there are several avenues for future work: One can apply other ML models such as ensemble ML models, and investigate the existing trade-off between the detection performance and running time as well as approaches to improve this balance. Larger datasets are also of interest to be exploited to consolidate the model's generalization and robustness. Furthermore, extending the proposed framework for a multi-UAV setting is an interesting problem at the analysis and practical levels. In this case, the complexity will increase as efficient coordination among UAVs is required for optimal operation. Another extension is to consider more constraints such as communication cost and schedule of the UAVs operation especially in a multi-UAV setting. Finally, using datasets for different situations such as images taken at night is another potential direction one can investigate.

Appendix

A. UAV Features

For this project, we chose the DJI Mavic 2 Zoom due to its considerable array of features. Some of the most notable features include:

- APAS 3.0 for Multidirectional Obstacle Avoidance.
- 4K Video and raw/HDR photos.
- 35-minute battery life.
- 10 km transmission distance and a maximum speed of 20 m/s.
- Waypoint selection for 99 waypoints with curved and straight-line routes.
- Autonomously focusing and following a point of interest.
- Programming SDK and functionality with Windows, Android, and iOS.
- Autopilot modes that allow for autonomous and dynamic scheduling and determining the flight route.
- Return to home functionality to return to the initial start point.
- Safety Features for loss of connection.

These features collectively contribute to the development of an autonomous drone and pipeline. Some notes on these features are:

- Even though there is a 99 waypoint limit on the waypoint mission, intelligent programming tricks are applied to override this limitation.
- The drone is equipped with a downward vision system and an Infrared sensing system. DJI also provides a thermal SDK for its more advanced drones (Zenmuse, Mavic 2 Enterprise), which can be used to analyze infrared images clicked by the downward infrared system. While this can be used in conjunction with the ML algorithm, it was not used for the purposes of this project. This would, however, be a method of adding confidence to the results.
- There are a total of seven Intelligent Flight Modes: Hyperlapse, Quickshots, ActiveTrack (follow a moving object), Point-of-Interest (hover or circle around a POI), Waypoints, TapFly, and Cinematic.
- To ensure that the flight is safe for both, the drone and the surrounding environment, the following features are present to avoid disasters: Intelligent Smart-Battery, Smart Return-to-Home (RTH), Low-power RTH, loss of connection smart RTH, DJI Simulator, and emergency pause.

B. Path-Planning

The distances between longitude and latitude are calculated using the Haversine Formula, which gives the shortest distance between two points on a sphere [72].

Note that the Earth is not spherical but rather ellipsoid in shape which can lead to errors in distance calculations. To minimize errors in the calculation of distances between two points, the closest estimation to the radius of the Earth (had the Earth been a sphere) in Massachusetts is selected. Depending on the latitude, the value of the radius can be changed to obtain more accurate results. Also, note that the radius of the Earth is independent of longitude and only dependent on latitude. The calculation of the geocentric radius given a geodetic latitude is presented below

$$R(\varphi) = \sqrt{\frac{(a^2 \cos\varphi)^2 + (b^2 \sin\varphi)^2}{(a\cos\varphi)^2 + (b\sin\varphi)^2}}, \tag{35}$$

where $R(\varphi)$ is the radius at the given geodetic latitude φ, $a = 6378137$m is the equatorial radius, and $b = 6378100$m is the polar radius.

In order to further account for the loss of accuracy in calculating the distance, an accuracy rate of 90% is assumed for the Haversine Formula. Adjustments were made to the calculations to account for this accuracy assumption, and the vertical and horizontal distances between the waypoints were adjusted to be 90% of the calculated distances. This accuracy rate was arbitrarily selected, and then tested in the field on the calculated distance (without assumption) and traveled distance (with assumption) for 100 distance measures to get an average accuracy of 92.3 % with a standard deviation of ±4 %. Finally, in order to ensure that the entire area is covered as well as avoid any mistakes in the Haversine formula, the diagonal FOV is subtracted by 3 degrees before performing any distance/area calculations.

C. Implementation

The implementation of the system involved combining different components to create an end-to-end pipeline of autonomous flight and fire detection.

The DJI SDK Timeline Mission was used to create this end-to-end system. Timeline missions allow the user to schedule a chain of Missions (DJIMission) and Actions (DJIMission-Action) to be executed one after the other in the order they were scheduled in. To program the autonomous flight a combination of Waypoint Missions was used, while to program the ML algorithm a custom Action was created by inheriting the DJIMissionAction class. At first, the user is asked to select four points to form a rectangular area that becomes the boundary of the area they would like to survey. The user is also given the option to adjust the flight height, speed, and zoom, which in combination with the boundary is used to generate a list of coordinates as the path for the autonomous flight. Each coordinate in the path is then selected to be its own Waypoint Mission, and therefore a path with n coordinates will be used to create a Timeline with n Waypoint Missions. Since every Waypoint Mission must consist of at least 2 waypoints, one for the start and one for the finish, at each waypoint in the point, the drone is programmed to ascend 0.5 meters above its current altitude and then descend 0.5 meters back down. This sequence creates a total of three waypoints for each Waypoint Mission in the Timeline.

Each Waypoint Mission in the Timeline is programmed with 3 actions to be executed in sequence once the drone reaches the designated waypoint: 1) Rotate the gimbal pitch to 90 degrees to make the camera face down, 2) Capture an image utilizing the camera, and 3) Rotate the gimbal pitch to 0 degrees to orient the camera face in drone heading. These captured images are accessed by the custom Mission Action that was created to run the ML algorithm on each image. Each custom action is scheduled right after each of the Waypoint Missions, giving rise to n waypoint missions and n custom actions that run alternatively in the Timeline.

The custom ML action inherits the DJIMissionAction class which is the base class for all Mission Control Timeline actions. The inheritance requires the creation and programming of the following inherited functions: run, willRun, pauseRun, and stopRun. For this project, only the run and willRun functions were fully implemented while the others were inherited and used to log into the console. The custom ML action executes as follows: 1) Connect to the camera, 2) Download the image using Media Download, 3) Set the camera back to shoot mode, 4) Resize the image to 256×512, 5) Run the ML recognition on the image, and 6) Update the global variable that saves the results. All operations in this chain are scheduled to be executed sequentially thereby ensuring that none of them overlap.

Acknowledgment

The authors would like to thank Jarrett Austin of the Pelham Fire Department (Massachusetts) for all his assistance during the conducting of the experiments.

References

[1] A. L. Adams et al., "Search is a time-critical event: when search and rescue missions may become futile," *Wilderness Environ. Med.*, vol. 18, no. 2, pp. 95–101, 2007.

[2] T. Giitsidis, E. G. Karakasis, A. Gasteratos, and G. Ch. Sirakoulis, "Human and fire detection from high altitude UAV images," in *Proc. 23rd Euromicro Int. Conf. Parallel, Distrib., Netw.-Based Process.*, Turku, Finland, Mar. 2015, pp. 309–315.

[3] B. Mishra, D. Garg, P. Narang, and V. Mishra, "Drone-surveillance for search and rescue in natural disaster," *Comput. Commun.*, vol. 156, pp. 1–10, Apr. 2020.

[4] M. N. H. Khan and C. Neustaedter, "An exploratory study of the use of drones for assisting firefighters during emergency situations," in *Proc. CHI Conf. Hum. Factors Comput. Syst.*, 2019, pp. 1–14.

[5] B. Aydin, E. Selvi, J. Tao, and M. Starek, "Use of fire-extinguishing balls for a conceptual system of drone-assisted wildfire fighting," *Drones*, vol. 3, no. 1, p. 17, Feb. 2019.

[6] C. W. Lum, A. Summers, B. Carpenter, A. Rodriguez, and M. Dunbabin, "Automatic wildfire detection and simulation using optical information from unmanned aerial systems," SAE Tech. Paper 2015-01-2474, 2015.

[7] M. T. Rashid, Y. Zhang, D. Zhang, and D. Wang, "CompDrone: Towards integrated computational model and social drone based wildfire monitoring," in *Proc. 16th Int. Conf. Distrib. Comput. Sensor Syst. (DCOSS)*, May 2020, pp. 43–50.

[8] M. Park, D. Q. Tran, D. Jung, and S. Park, "Wildfire-detection method using DenseNet and CycleGAN data augmentation-based remote camera imagery," *Remote Sens.*, vol. 12, no. 22, p. 3715, Nov. 2020.

[9] A. Viseras, J. Marchal, M. Schaab, J. Pages, and L. Estivill, "Wildfire monitoring and hotspots detection with aerial robots: Measurement campaign and first results," in *Proc. IEEE Int. Symp. Saf., Secur., Rescue Robot. (SSRR)*, Sep. 2019, pp. 102–103.

[10] H. Pan, D. Badawi, and A. E. Cetin, "Computationally efficient wildfire detection method using a deep convolutional network pruned via Fourier analysis," *Sensors*, vol. 20, no. 10, p. 2891, May 2020.

[11] D. C. Schedl, I. Kurmi, and O. Bimber, "An autonomous drone for search and rescue in forests using airborne optical sectioning," *Sci. Robot.*, vol. 6, no. 55, 2021, Art. no. eabg1188.

[12] Y. Karaca et al., "The potential use of unmanned aircraft systems (drones) in mountain search and rescue operations," *Amer. J. Emergency Med.*, vol. 36, no. 4, pp. 583–588, 2018.

[13] S. Hayat, E. Yanmaz, C. Bettstetter, and T. X. Brown, "Multi-objective drone path planning for search and rescue with quality-of-service requirements," *Auto. Robots*, vol. 44, no. 7, pp. 1183–1198, Sep. 2020.

[14] M. H. Dominguez, S. Nesmachnow, and J.-I. Hernández-Vega, "Planning a drone fleet using artificial intelligence for search and rescue missions," in *Proc. IEEE XXIV Int. Conf. Electron., Electr. Eng. Comput. (INTERCON)*, Aug. 2017, pp. 1–4.

[15] J. Eyerman, G. Crispino, A. Zamarro, and R. Durscher, "Drone efficacy study (DES): Evaluating the impact of drones for locating lost persons in search and rescue events Brussels," DJI, Brussels Belgium, Tech. Rep. 19, 2018.

[16] F. L. Pereira, "Optimal control problems in drone operations for disaster search and rescue," *Proc. Comput. Sci.*, vol. 186, pp. 78–86, Jan. 2021.

[17] N. M. K. Dousai and S. Loncaric, "Detection of humans in drone images for search and rescue operations," in *Proc. 3rd Asia Pacific Inf. Technol. Conf.*, 2021, pp. 69–75.

[18] D. Rjoub, A. Alsharoa, and A. Masadeh, "Early wildfire detection using UAVs integrated with air quality and LiDAR sensors," in *Proc. IEEE 96th Veh. Technol. Conf. (VTC-Fall)*, London, U.K., Sep. 2022, pp. 1–5.

[19] K. Harikumar, J. Senthilnath, and S. Sundaram, "Multi-UAV oxyrrhis marina-inspired search and dynamic formation control for forest firefighting," *IEEE Trans. Autom. Sci. Eng.*, vol. 16, no. 2, pp. 863–873, Apr. 2019.

[20] T. Lei, C. Luo, T. Sellers, Y. Wang, and L. Liu, "Multitask allocation framework with spatial dislocation collision avoidance for multiple aerial robots," *IEEE Trans. Aerosp. Electron. Syst.*, vol. 58, no. 6, pp. 5129–5140, Dec. 2022.

[21] S. Minaeian, J. Liu, and Y.-J. Son, "Effective and efficient detection of moving targets from a UAV's camera," *IEEE Trans. Intell. Transp. Syst.*, vol. 19, no. 2, pp. 497–506, Feb. 2018.

[22] C. Wang, J. Wang, Y. Shen, and X. Zhang, "Autonomous navigation of UAVs in large-scale complex environments: A deep reinforcement learning approach," *IEEE Trans. Veh. Technol.*, vol. 68, no. 3, pp. 2124–2136, Mar. 2019.

[23] S. Wang, F. Jiang, B. Zhang, R. Ma, and Q. Hao, "Development of UAV-based target tracking and recognition systems," *IEEE Trans. Intell. Transp. Syst.*, vol. 21, no. 8, pp. 3409–3422, Aug. 2020.

[24] B. Wang, Y. Sun, D. Liu, H. M. Nguyen, and T. Q. Duong, "Social-aware UAV-assisted mobile crowd sensing in stochastic and dynamic environments for disaster relief networks," *IEEE Trans. Veh. Technol.*, vol. 69, no. 1, pp. 1070–1074, Jan. 2020.

[25] N. Patrizi, G. Fragkos, K. Ortiz, M. Oishi, and E. E. Tsiropoulou, "A UAV-enabled dynamic multi-target tracking and sensing framework," in *Proc. IEEE Global Commun. Conf.*, Taiwan, Dec. 2020, pp. 1–6.

[26] H. Huang and A. Savkin, "Navigating UAVs for optimal monitoring of groups of moving pedestrians or vehicles," *IEEE Trans. Veh. Technol.*, vol. 70, no. 4, pp. 3891–3896, Apr. 2021.

[27] X. Chen, Z. Li, Y. Yang, L. Qi, and R. Ke, "High-resolution vehicle trajectory extraction and denoising from aerial videos," *IEEE Trans. Intell. Transp. Syst.*, vol. 22, no. 5, pp. 3190–3202, May 2021.

[28] H. Shakhatreh et al., "Unmanned aerial vehicles (UAVs): A survey on civil applications and key research challenges," *IEEE Access*, vol. 7, pp. 48572–48634, 2019.

[29] S. E. Eid and S. Sham Dol, "Design and development of lightweight-high endurance unmanned aerial vehicle for offshore search and rescue operation," in *Proc. Adv. Sci. Eng. Technol. Int. Conf. (ASET)*, Dubai, United Arab Emirates, Mar. 2019, pp. 1–5.

[30] Z. Lin, H. H. T. Liu, and M. Wotton, "Kalman filter-based large-scale wildfire monitoring with a system of UAVs," *IEEE Trans. Ind. Electron.*, vol. 66, no. 1, pp. 606–615, Jan. 2019.

[31] J. Q. Cui et al., "Drones for cooperative search and rescue in post-disaster situation," in *Proc. IEEE 7th Int. Conf. Cybern. Intell. Syst. (CIS) IEEE Conf. Robot., Automat. Mechatronics (RAM)*, Siem Reap, Cambodia, Jul. 2015, pp. 167–174.

[32] A. Quan, C. Herrmann, and H. Soliman, "Project vulture: A prototype for using drones in search and rescue operations," in *Proc. 15th Int. Conf. Distrib. Comput. Sensor Syst. (DCOSS)*, Santorini, Greece, May 2019, pp. 619–624.

[33] F. M. A. Hossain, Y. Zhang, C. Yuan, and C.-Y. Su, "Wildfire flame and smoke detection using static image features and artificial neural network," in *Proc. 1st Int. Conf. Ind. Artif. Intell. (IAI)*, Shenyang, China, Jul. 2019, pp. 1–6.

[34] Z. Domozi, D. Stojcsics, A. Benhamida, M. Kozlovszky, and A. Molnar, "Real time object detection for aerial search and rescue missions for missing persons," in *Proc. IEEE 15th Int. Conf. Syst. Syst. Eng. (SoSE)*, Budapest, Hungary, Jun. 2020, pp. 000519–000524.

[35] S. Sambolek and M. Ivasic-Kos, "Automatic person detection in search and rescue operations using deep CNN detectors," *IEEE Access*, vol. 9, pp. 37905–37922, 2021.

[36] C. Yuan, Z. Liu, and Y. Zhang, "Vision-based forest fire detection in aerial images for firefighting using UAVs," in *Proc. Int. Conf. Unmanned Aircr. Syst. (ICUAS)*, Arlington, VA, USA, Jun. 2016, pp. 1200–1205.

[37] C. Yuan, Z. Liu, and Y. Zhang, "Fire detection using infrared images for UAV-based forest fire surveillance," in *Proc. Int. Conf. Unmanned Aircr. Syst. (ICUAS)*, Miami, FL, USA, Jun. 2017, pp. 567–572.

[38] P. Ramanatha, U. R. Nelakuditi, S. Ravishankar, and V. Ranganathan, "UAV based smoke plume detection system controlled via the short message service through the GSM network," in *Proc. Int. Conf. Inventive Comput. Technol. (ICICT)*, vol. 2, Coimbatore, India, Aug. 2016, pp. 1–4.

[39] G. D. Georgiev, G. Hristov, P. Zahariev, and D. Kinaneva, "Forest monitoring system for early fire detection based on convolutional neural network and UAV imagery," in *Proc. 28th Nat. Conf. Int. Participation (TELECOM)*, Sofia, Bulgaria, Oct. 2020, pp. 57–60.

[40] J. McConkey and Y. Liu, "Semi-autonomous control of drones/UAVs for wilderness search and rescue," in *Proc. 8th Int. Conf. Automat., Control Robot. Eng. (CACRE)*, Hong Kong, Jul. 2023, pp. 317–321.

[41] H. Wu, H. Li, A. Shamsoshoara, A. Razi, and F. Afghah, "Transfer learning for wildfire identification in UAV imagery," in *Proc. 54th Annu. Conf. Inf. Sci. Syst. (CISS)*, Princeton, NJ, USA, Mar. 2020, pp. 1–6.

[42] Y. Xiang, S. Guo, S. Xia, X. Gu, J. Chen, and G. Cui, "NLOS target positioning method based on UAV millimeter-wave radar," *IEEE Sensors J.*, vol. 24, no. 2, pp. 1975–1987, Jan. 2024.

[43] J. Ibenthal, L. Meyer, H. Piet-Lahanier, and M. Kieffer, "Localization of partially hidden moving targets using a fleet of UAVs via bounded-error estimation," *IEEE Trans. Robot.*, vol. 39, no. 6, pp. 4211–4229, Dec. 2023.

[44] R. G. Ribeiro, L. P. Cota, T. A. M. Euzébio, J. A. Ramírez, and F. G. Guimarães, "Unmanned-aerial-vehicle routing problem with mobile charging stations for assisting search and rescue missions in postdisaster scenarios," *IEEE Trans. Syst., Man, Cybern., Syst.*, vol. 52, no. 11, pp. 6682–6696, Nov. 2022.

[45] R. Alyassi, M. Khonji, A. Karapetyan, S. C.-K. Chau, K. Elbassioni, and C.-M. Tseng, "Autonomous recharging and flight mission planning for battery-operated autonomous drones," *IEEE Trans. Automat. Sci. Eng.*, vol. 20, no. 2, pp. 1034–1046, Apr. 2023.

[46] E. T. Alotaibi, S. S. Alqefari, and A. Koubaa, "LSAR: Multi-UAV collaboration for search and rescue missions," *IEEE Access*, vol. 7, pp. 55817–55832, 2019.

[47] J. Keller, D. Thakur, M. Likhachev, J. Gallier, and V. Kumar, "Coordinated path planning for fixed-wing UAS conducting persistent surveillance missions," *IEEE Trans. Autom. Sci. Eng.*, vol. 14, no. 1, pp. 17–24, Jan. 2017.

[48] H. Huang, A. V. Savkin, and W. Ni, "Online UAV trajectory planning for covert video surveillance of mobile targets," *IEEE Trans. Autom. Sci. Eng.*, vol. 19, no. 2, pp. 735–746, Apr. 2022.

[49] M. Rodríguez, A. Al-Kaff, Á. Madridano, D. Martín, and A. de la Escalera, "Wilderness search and rescue with heterogeneous multi-robot systems," in *Proc. Int. Conf. Unmanned Aircr. Syst. (ICUAS)*, Athens, Greece, Sep. 2020, pp. 110–116.

[50] E. P. de Freitas, M. Basso, A. A. S. da Silva, M. R. Vizzotto, and M. S. C. Corrêa, "A distributed task allocation protocol for cooperative multi-UAV search and rescue systems," in *Proc. Int. Conf. Unmanned Aircr. Syst. (ICUAS)*, Athens, Greece, Jun. 2021, pp. 909–917.

[51] F. H. Panahi, F. H. Panahi, and T. Ohtsuki, "A reinforcement learning-based fire warning and suppression system using unmanned aerial vehicles," *IEEE Trans. Instrum. Meas.*, vol. 72, pp. 1–16, 2023.

[52] W. Chen and L. Liu, "Pareto Monte Carlo tree search for multi-objective informative planning," in *Proc. Robot., Sci. Syst.*, Freiburg Im Breisgau, Germany, Jun. 2019.

[53] C. Phan and H. H. Liu, "A cooperative UAV/UGV platform for wildfire detection and fighting," in *Proc. Asia Simulation Conf.-7th Int. Conf. Syst. Simulation Sci. Comput.*, Beijing, China, Oct. 2008, pp. 494–498.

[54] F. Afghah, A. Razi, J. Chakareski, and J. Ashdown, "Wildfire monitoring in remote areas using autonomous unmanned aerial vehicles," in *Proc. IEEE Conf. Comput. Commun. Workshops (INFOCOM WKSHPS)*, Apr. 2019, pp. 835–840.

[55] A. Shamsoshoara, F. Afghah, A. Razi, S. Mousavi, J. Ashdown, and K. Turk, "An autonomous spectrum management scheme for unmanned aerial vehicle networks in disaster relief operations," *IEEE Access*, vol. 8, pp. 58064–58079, 2020.

[56] O. M. Bushnaq, A. Chaaban, and T. Y. Al-Naffouri, "The role of UAV-IoT networks in future wildfire detection," *IEEE Internet Things J.*, vol. 8, no. 23, pp. 16984–16999, Dec. 2021.

[57] J. T. Mawanza, J. T. Agee, and E. Bhero, "Adaptive finite-time time-varying elliptical formation control for a group of quadrotors UAVs for cooperative wildfire monitoring," in *Proc. 30th Southern Afr. Universities Power Eng. Conf. (SAUPEC)*, Durban, South Africa, Jan. 2022, pp. 1–6.

[58] V. San Juan, M. Santos, and J. M. Andújar, "Intelligent UAV map generation and discrete path planning for search and rescue operations," *Complexity*, vol. 2018, Apr. 2018, Art. no. 6879419.

[59] P. R. Sokkappa, "The cost-constrained traveling salesman problem," Ph.D. dissertation, Stanford Univ., Stanford, CA, USA, 1991.

[60] A. Bäuerle, C. van Onzenoodt, and T. Ropinski, "Net2 Vis—A visual grammar for automatically generating publication-tailored CNN architecture visualizations," 2019, *arXiv:1902.04394*.

[61] S. Hochreiter, "The vanishing gradient problem during learning recurrent neural nets and problem solutions," *Int. J. Uncertainty, Fuzziness Knowl.-Based Syst.*, vol. 6, no. 2, pp. 107–116, Apr. 1998.

[62] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," 2015, *arXiv:1512.03385*.

[63] O. Russakovsky et al., "ImageNet large scale visual recognition challenge," 2014, *arXiv:1409.0575*.

[64] T.-Y. Lin et al., "Microsoft COCO: Common objects in context," 2014, *arXiv:1405.0312*.

[65] A. for Mankind. *Wildfire-Dataset*. Accessed: Apr. 2019. [Online]. Available: https://github.com/aiformankind/wildfire-dataset/tree/master/wildfire-smoke

[66] B. Dincer. (May 2021). *Wildfire Detection Image Data*. [Online]. Available: https://www.kaggle.com/brsdincer/wildfire-detection-image-data
[67] S. Ioffe and C. Szegedy, "Batch normalization: Accelerating deep network training by reducing internal covariate shift," 2015, *arXiv:1502.03167*.
[68] Y. Zeng, J. Xu, and R. Zhang, "Energy minimization for wireless communication with rotary-wing UAV," *IEEE Trans. Wireless Commun.*, vol. 18, no. 4, pp. 2329–2345, Apr. 2019.
[69] X. Zhuang, Y. Huang, K. Palaniappan, and Y. Zhao, "Gaussian mixture density modeling, decomposition, and applications," *IEEE Trans. Image Process.*, vol. 5, no. 9, pp. 1293–1302, Sep. 1996.
[70] H. Choset, "Coverage for robotics—A survey of recent results," *Ann. Math. Artif. Intell.*, vol. 31, no. 1, pp. 113–126, 2001.
[71] L. Lin and M. A. Goodrich, "Hierarchical heuristic search using a Gaussian mixture model for UAV coverage planning," *IEEE Trans. Cybern.*, vol. 44, no. 12, pp. 2532–2544, Dec. 2014.
[72] *Haversine Formula*. Accessed: Apr. 2004. [Online]. Available: https://en.wikipedia.org/wiki/Haversine_formula

Mohammadjavad Khosravi received the M.Sc. degree in computer engineering from Isfahan University, Isfahan, Iran, in 2011. He is currently pursuing the Ph.D. degree in computer engineering with the University of Massachusetts at Amherst, Amherst, MA, USA, under the supervision of Prof. Hossein Pishro-Nik. His current research interests include unmanned aerial vehicles, deep reinforcement learning, and machine learning in wireless communication networks.

Rushiv Arora received the bachelor's degree in computer science and the bachelor's degree in computer engineering from the University of Massachusetts at Amherst, where he is currently pursuing the M.S. degree in computer science, where his research is supported by the Bay State Fellowship. His research interests include reinforcement learning and machine learning, with a current emphasis on hierarchical RL and model-based RL, with the long-term goal of realistically building competent autonomous agents that continuously learn about their environment, hopefully (but not necessarily) with a biological foundation. He was awarded the CICS Outstanding Undergraduate Award and the ECE Award of Excellence for the bachelor's degree.

Saeede Enayati (Member, IEEE) received the B.Sc. degree in electrical engineering from Shahid Beheshti University in 2012, the M.Sc. degree from Tarbiat Modares University, Tehran, Iran, in 2015, and the Ph.D. degree from the University of Massachusetts (UMass), Amherst, MA, USA, in May 2023. She was a Post-Doctoral Fellow with UMass until December 2023. She is currently a Research Engineer with Tiami Networks, Elk Grove, CA, USA. Her research interests include wireless network analysis, integrated sensing and communications (ISAC), and 3GPP channel modeling. She has been awarded the Ting-Wei Tang Prize for her Ph.D. dissertation by the ECE Department, UMass.

Hossein Pishro-Nik (Member, IEEE) received the B.Sc. degree in electrical and computer engineering from the Sharif University of Technology, Tehran, Iran, and the M.Sc. and Ph.D. degrees in electrical and computer engineering from Georgia Institute of Technology. He is currently a Professor with the Department of Electrical and Computer Engineering, University of Massachusetts at Amherst, Amherst, MA, USA. His research interests include information theory, wireless networks, vehicular/UAV networks, privacy, statistical learning, and decision-making. His awards include the NSF Faculty Early Career Development (CAREER) Award, the Outstanding Junior Faculty Award from UMass, and the Outstanding Graduate Research Award from Georgia Institute of Technology. He has served as an Associate Editor for IEEE TRANSACTIONS ON WIRELESS COMMUNICATIONS, IEEE TRANSACTIONS ON COMMUNICATIONS, and IEEE TRANSACTIONS ON INFORMATION FORENSICS AND SECURITY.