Aerial Vehicle Path Planning for Monitoring Wildfire Frontiers

Ryan C. Skeele and Geoffrey A. Hollinger * †

Abstract This paper explores the use of unmanned aerial vehicles (UAVs) in wildfire monitoring. To begin establishing effective methods for autonomous monitoring, a simulation (FLAME) is developed for algorithm testing. To simulate a wildfire, the well established FARSITE fire simulator is used to generate realistic fire behavior models. FARSITE is a wildfire simulator that is used in the field by Incident Commanders (IC's) to predict the spread of the fire using topography, weather, wind, moisture, and fuel data. The data obtained from FARSITE is imported into FLAME and parsed into a dynamic frontier used for testing hotspot monitoring algorithms. In this paper, points of interest along the frontier are established as points with a fireline intensity (British-Thermal-Unit/feet/second) above a set threshold. These interest points are refined into hotspots using the Mini-Batch K-means Clustering technique. A distance threshold differentiates moving hotspot centers and newly developed hotspots. The proposed algorithm is compared to a baseline for minimizing the sum of the max time untracked $J(t)$. The results show that simply circling the fire performs poorly (baseline), while a weighted-greedy metric (proposed) performs significantly better. The algorithm was then run on a UAV to demonstrate the feasibility of real world implementation.

1 Introduction

Recent developments in sensing technology have made possible low cost, reliable unmanned aerial vehicles (UAVs). These field robots are being implemented in var-

ious application domains, but specifically show promise in applications hazardous for humans. Studying wildfires has an obvious benefit when considering the human cost spent combating them.

One of the main issues in combating wildfires is monitoring the progression of the fire over time (Kremens et al, 2001). Live fire frontier monitoring can help produce quicker decisions and result in better resource allocation and fire management (Koulas, 2009). During wildfires, the information available to the Incident Commander (IC) is critical. Current methods of tracking a fire involve a human pilot flying several miles away from the fire and verbally reporting to the IC what trends they see in the fire. Satellite imaging is also available but is often rendered useless by smoke. In 2012, there were a total of 67,774 fires, destroying 9.3 million acres, and costing over $1.9 billion to suppress in the U.S. alone (Center, 2015). Large aircraft can negatively affect the fireline, for example if flown too low (below 1,000 ft), wake vortices from the windtips produce wind gusts which can cause torching and spotting (Group, 1996).
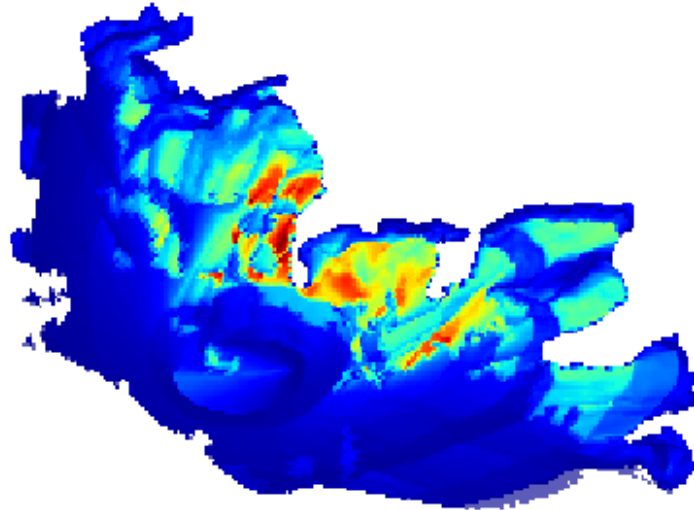

Fig. 1: Wildfire simulation example (red areas correspond to hotter areas of the fire). We propose a weighted-greedy algorithm for optimizing the monitoring trajectories of aerial vehicles in wildfire scenarios.

This paper presents tests of different hotspot monitoring algorithms to gather important information for the Incident Commander (IC) managing the wildfire. This research aims to help improve a UAV's effectiveness in gathering valuable information for the IC. To simulate wildfires, a program developed by the Department of

Agriculture and Forest Service is used. FARSITE is a free program used by the U.S. Forest Service, National Park Service, and more specifically ICs, to predict the fire's behavior using data on the topography, weather, wind, moisture, and fuel (USD, 2014). FARSITE exports various characteristics of the fire. While our simulation (FLAME) uses fireline intensity data (BTU/ft/s), other fire metrics like flame length and rate of spread also be incorporated. See figure 1 for a fireline intensity map of a simulated fire.

Robotic monitoring has become a hot research topic in recent years, due to robots playing a more integral role in collecting environmental data. This has led to a variety of monitoring algorithms (Cassandras et al, 2011, Lan and Schwager, 2013, Smith et al, 2011, Smith and Rus, 2010, Smith et al, 2012). Wildfires are highly unpredictable, acting as a unique dynamic frontier. Dynamic monitoring has been explored (Bertozzi et al, 2005, Smith et al, 2011), but fire frontier monitoring is a largely unexplored domain. Our simulator (FLAME) models a dynamic fire frontier and uses techniques like Mini-Batch K-Means Clustering to achieve a similar problem formulation as related monitoring research.

Tracking the most volatile locations on the fireline will give valuable information for the IC. These hotspots will be intelligently monitored by the UAV, using algorithms for minimizing the time hotspots are left unmonitored.

While wildfires were the chosen domain, this research is not limited to wildfires. Similar application domains with dynamic frontiers include: algae blooms, pollution spills, and military battles (Marthaler and Bertozzi, 2004, Susca et al, 2008). These similar domains can also be analyzed using the techniques developed in this paper.

The main novelties of this paper include: (1) a simulation (FLAME) which uses realistic fire modeling software for accurate fire characteristics, (2) a novel fire tracking algorithm which outperforms existing methods, (3) the first investigation into adaptive monitoring of hotspots along a dynamic frontier, and (4) hardware experiments demonstrating the ability to implement our work with existing technology. Taken together, these contributions provide a new approach to the general problem of monitoring dynamic frontiers.

The remainder of this paper is organized as follows. First, we will establish the current state of similar research (Section 2). Following that, we will overview the problem and assumptions we made during our investigation (Section 3). Next, we describe the simulation and a novel approach to frontier monitoring (Section 4). Finally, the algorithm is described in detail in (Section 5), and the simulation results are presented (Section 6), and the hardware experiments are discussed (Section 7).

2 Related Work

Work on autonomous information gathering began with early work in sequential hypothesis testing (Wald, 1945), which focused on determining which experiments could efficiently classify the characteristics of an unknown. This line of research developed into more general approaches and evolved into the field of active per-

ception (Bajcsy, 1988). Similar insights led to using optimization techniques on robotic information gathering problems, and researchers later developed algorithms for minimizing long-term information uncertainty (Cassandras et al, 2011, 2013).

Robotic systems are becoming more commonly used as mass data-gathering tools by scientists (Dunbabin and Marques, 2012, Hollinger et al, 2012, Smith et al, 2011). Robots are already collecting large datasets on environmental change. Algae blooms, pollution, and other climate variables are application domains for persistent monitoring techniques. Persistent/adaptive monitoring in robotics is currently a growing research topic. Prior work has explored different approaches to monitoring stationary and dynamic feature points. While these adaptive sampling techniques focus on optimizing uncertainty levels in static (Cassandras et al, 2011, Lan and Schwager, 2013, Smith and Rus, 2010) and in dynamic environments (Smith et al, 2011, 2012), prior work often focuses on systems in obstacle-free environments. Some research has examined collision avoidance (Hollinger and Sukhatme, 2013, Soltero et al, 2011), but adaptive sampling along dynamic frontiers remains an ongoing research problem.

In (Casbeer et al, 2005), fire frontier tracking was integrated into a simulation for determining UAV tracking accuracy of the fire perimeter. Their UAVs follow a circular path around the fire similar to our baseline. However, our metric is to track the most active parts of the fire. We compare the baseline against our weighted-distance algorithm. Our research presents the first investigation into adaptive monitoring of hotspots along a dynamic frontier. We span the domains of hotspot monitoring and dynamic frontier tracking to evaluate path planning techniques in our FLAME simulator. This line of work allows us to test new algorithms in real-world scenarios.

3 Problem Formulation

We will now formally introduce the problem domain and the assumptions we made. We will also introduce the metric we use to evaluate our algorithm against the baseline.

We assume that GPS and communication between the IC and the vehicle are always available. This means the UAV can always localize itself and never needs to return to the starting location to transfer collected data. We assume the UAV always has the simulated fire frontier in order to find the hotspot locations. Additionally, the UAV is assumed, for comparison purposes, to have unlimited endurance.

Each hotspot has a corresponding time since last tracked by the UAV and the maximum time its been left untracked (ϕ) in the past. The sum of ϕ of all hotspots was chosen as the metric to evaluate the effectiveness of an algorithm. In this paper, fireline intensity is used as the crucial information needed by the IC. The intensity is monitored through the clustering into hotspots, directly relating to the goal of providing the IC with up-to-date information about the fire progression.

$$J(t) = \sum_{i=0}^{hotspots} \phi_i \,, \tag{1}$$

where ϕ is max time untracked

The goal is to minimize the metric $J(t)$, which corresponds to timely hotspot monitoring, through an optimized trajectory for the UAV.

4 FLAME Simulation

We will now explain our simulation and how we developed each of the different components. Figure 2 should be used as a reference of the state transitions in the simulation. There are two aspects to the weighted-greedy algorithm, picking which hotspot to go to, and how to get there.
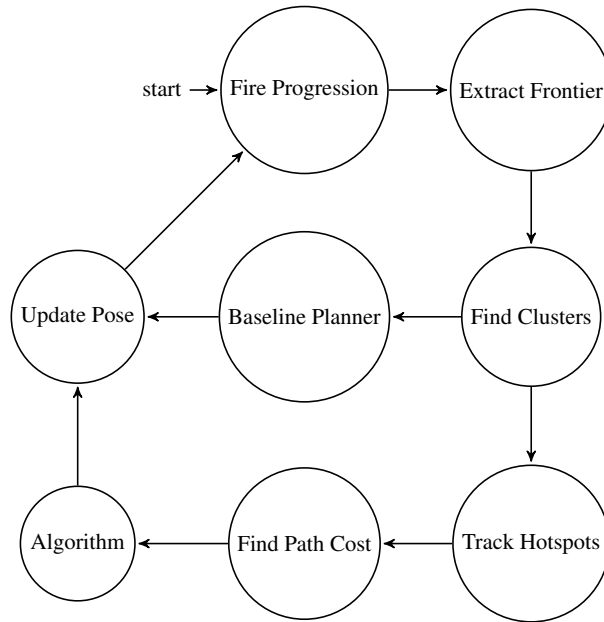
Fig. 2: State diagram of FLAME

Fire data is generated using FARSITE, the wildfire simulator currently used by ICs during wildfire management (USD, 2014). The data is exported in the form of time of arrival, and a measurable characteristic of the fire. In this work we use the fireline intensity at each location. As stated above, the task is to minimize the sum

of max time untracked (ϕ) over all hotspots. At mission start, the UAV must first find the fire and begin identifying the hotspot regions.

Tracking a hotspot is done by calculating the distance between a previous set of hotspots relative to a new set. To determine when a hotspot moved as the fire progressed, a threshold is implemented. If a hotspot is not within the distance threshold of any previous hotspots, it is then classified as a new hotspot. Even after careful tuning, this approach can still lead to some untracked hotspots where the hotspot existence is too short for any response by the UAV.

To identify hotspots, all points along the frontier with a fireline intensity above a normalized threshold are parsed using a clustering technique called Mini-Batch K-means (Lloyd, 1982). K-means clustering was chosen because it directly relates the number of interest points (how active the fire is) to the number of cluster centers (hotspots). The desirable amount of clusters (K) changes as the fire evolves. We actively determine the K value for adaptive hotspot extraction with the following formula. With K as the number of centers, and N as the number of interest points,

$$K = \sqrt{N/2} \tag{2}$$

FLAME uses A* path planning for generating paths from the UAV to hotspot locations around the fire. This method works better for estimating path cost over a simple Euclidean distance estimate due to the spherical tendency of the fire spread. Other similar methods where explored to increase efficiency, such as Jump Point Search. Jump Point Search gives respectable speed gains in environments with large open spaces, while the UAVs path remained mostly along the fire frontier. Methods like wall-following could provide faster simulation times, but lack expandability to more complex frontiers, and provide less accurate path costs. Due to the shape of the fire, any benefits of these alternatives were determined to be inconsequential. It was therefore determined to use the A* search algorithm as the UAVs path planner.

A cost map is passed to the A* algorithm, and is generated by applying a blur to the map of the fire up to that point in time and assigning a high cost to areas within the fire. This helps ensure the path generated for the UAV is not within dangerous proximity of the fire, but can still be navigated close enough to monitor the hotspots. The algorithms were tested over seven different fires generated in FARSITE. The baseline and proposed algorithm are described in pseudo code in Algorithms 1 and 2. A state diagram of FLAME is provided in figure 2. The algorithm state is weighted, but may be replaced with any tracking algorithm for testing.

5 ALGORITHMS

The proposed algorithm is evaluated against a baseline in the following tests. The following sections will describe each algorithm and how it was implemented in the FLAME simulation. The first monitoring technique described is used as the baseline comparison. It exemplifies current tactics utilized in real world wild fire monitoring,

and prior research UAV fire monitoring (Casbeer et al, 2005). This is compared to our proposed approach, a weighted-greedy algorithm that moves to the hotspot that has remained untracked the longest with a tunable parameter of distance from the UAV. Figure 3 should be used as a reference of the difference between the two algorithms behaviors.
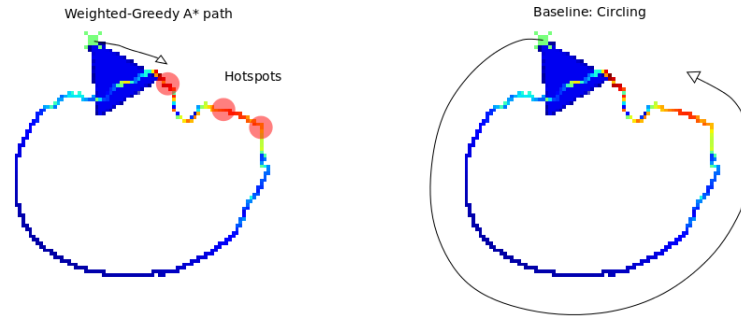
(a) UAV monitoring the fire using proposed algorithm identifies and tracks the most important part of the fire

(b) UAV monitoring the fire by constant circling will continue regardless of the state of the fire.

Fig. 3

5.1 Baseline

Traveling parallel to the dynamic fire frontier is used as a baseline model. Calculating a 90 degree transformation of the vector from the UAVs current location to the nearest point on the fire frontier gives the travel vector of the UAV. Maximum and minimum distance thresholds are imposed on the UAV so it can then move along the frontier monitoring hotspots while maintaining a safe distance from the fire. We use this as a baseline comparison based on the work of (Casbeer et al, 2005).

5.2 Weighted-Greedy

The weighted-greedy algorithm checks the time untracked of every live hotspot, calculates the distance to it, and targets the one with the highest score. Unlike the baseline, the weighted-greedy algorithm makes target decisions based on the current state of the hotspots.

This is done using the following formula where $\mathcal{H}$ is the target hotspot, $\mathcal{T}$ is the time untracked of each hotspot and $\mathcal{C}$ is the path cost to each hotspot:

Algorithm 1 Baseline Algorithm

1: Inputs: UAV_Location, frontier
2: **for** all points in frontier **do**
3:     points.distance = $\sqrt{(\text{points.x - UAV\_location.x})^2 + (\text{points.y - UAV\_location.y})^2}$
4: **end for**
5: closest_point = min(points.distance)
6: vector_to_nearest = ([UAV_location.x - closest.point.x], [UAV_location.y - closest.point.y])
7: normalized_vector = vector_to_nearest / distance_to_nearest
8: **if** dist_to_nearest > max_distance_to_fire **then**
9:     travel_vector = vector_to_nearest
10: **else if** dist_to_nearest < min_distance_to_fire **then**
11:     travel_vector = -vector_to_nearest
12: **else**
13:     travel_vector = (-vector_to_nearest.x, vector_to_nearest.y)
14: **end if**
15: path = travel_vector

Algorithm 2 Weighted Algorithm

1: Inputs: hotspots{location, time_untracked}, α, UAV_location
2: **for** all h in hotspots **do**
3:     h.path, h.path_cost = ASTAR(h.location, UAV_location)
4:     h.score = h.time_untracked $- \alpha *$ path_cost(h)
5:     **if** hotspot.score > target_hotspot.score **then**
6:         target_hotspot = h
7:     **end if**
8: **end for**
9: path = target_hotspot.path

$$\mathcal{H} = \operatorname*{argmin}_{h} \mathcal{T}_h - \alpha * \mathcal{C}_h \tag{3}$$

The proposed algorithm accounts for the distance to each hotspot when choosing the targeted hotspot. The weighting factor α is a parameter evaluated in figures 5, 4, and 6. The use of a weighting factor addresses some sub-optimality of using just a greedy algorithm. The weighting parameter helps intelligently pick a hotspot that may not be the longest untracked but is closer to the vehicle. A greedy algorithm will immediately move towards the hotspot with the longest time left untracked, disregarding any nearby hotspots that may not have been untracked for nearly quite as long.

6 Results

Using our FLAME simulator, we can compare our proposed weighted-greedy approach with traditional methods of monitoring of wildfires. The simulation was run on an Intel i7-4702HQ processor with 8 gigabytes of RAM. The UAV's decision and

planning methods took an average of .74 seconds to complete. This is fast enough for a UAV to implement in the field (see 7).

In comparison to the baseline, the weighted algorithm provided substantial improvement over the course of the trials. The plots in Figures 5, 4, and 6 show the two monitoring algorithms performance with different parameter settings. As previously discussed, the weighting parameter (α) is multiplied by path cost to the hotspot location. The hotspot cutoff β is the normalized threshold for a spot along the fire to be considered an interest point. This directly affects the total number of hotspots. Tests ran with a lower β will generate a higher number of hotspots for the UAV to track. Time on the X axis begins at first cluster appearance during the simulation. The Y axis shows the results of the comparison metric $J(t)$.

The averaged score over the seven fires are depicted as the bold lines. Around each line, the standard error of the mean is represented by the shading. Figure 4 shows the simulation results with a hotspot threshold $\beta = .35$. The plot shows the results with a corresponding α value of .5. Our proposed algorithm performs significantly better than currently used approach.
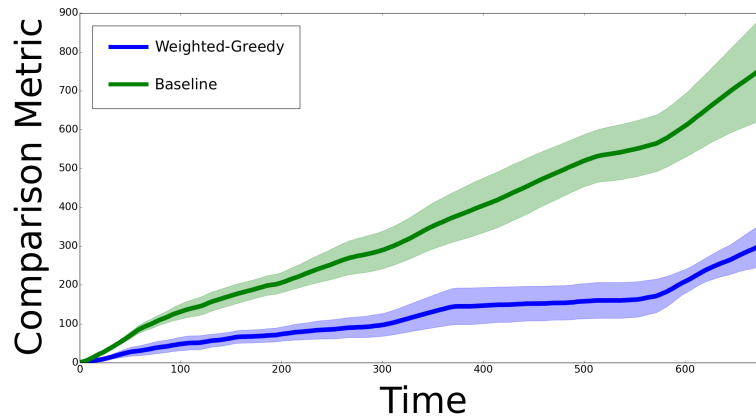
(a) Weighting parameter $\alpha = .5$ Hotspot Threshold $\beta = .35$

Fig. 4: Wildfire simulation, where the comparison metric is J(t) or the sum of max time untracked of all hotspots. Lower is better. The weighting parameter α is set at .5. The normalized threshold β, for a spot along the fire to be considered an interest point, is set to .35. A lower β corresponds to more hotspot locations. Error bars are one SEM.

In figure 5 the simulation is run with a β equal to .25. The β value (.25) is the lowest used and figure 5 shows the performance of both algorithms in an environment with the corresponding large set of hotspots.
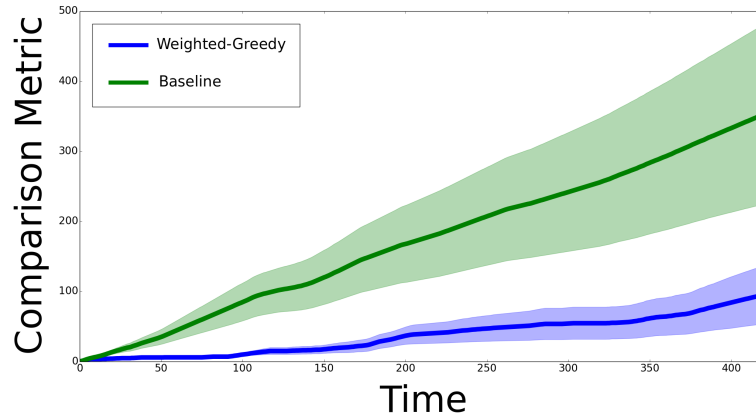
(a) Weighting parameter $\alpha = .5$ Hotspot Threshold $\beta = .25$

Fig. 5: Wildfire simulation, where the comparison metric is J(t) or the sum of max time untracked of all hotspots. Lower is better. The weighting parameter α is set at .5. The normalized threshold β, for a spot along the fire to be considered an interest point, is set to .25. A lower β corresponds to more hotspot locations. Error bars are one SEM.

Figure 6 depicts the simulation results with a hotspot threshold β at .45. This trial uses the highest β (fewest number of hotspots), and shows the plots performance with α value at .5. The standard error of the mean (SEM) for both algorithms is significantly higher in this test environment. The results demonstrate the algorithms ability to outperform the baseline in environments with only few clusters, or many clusters. In all cases presented here the proposed algorithm showed significant improvement over traditional wildfire monitoring methods. Our algorithm better tracks the dynamic regions of a dynamic frontier, providing valuable data to better track the frontier.

An interesting characteristic of the frontier monitoring is that it may be simplified into a 1 dimensional problem. Each timestep the UAV must decide between two options, if it wishes to move clockwise or counter-clockwise. It will be worth further investigation into leveraging this characteristic.

7 Hardware Experiments

To demonstrate the feasibility of the proposed algorithm, we implemented the algorithm on a live test. To test on hardware we set up the FLAME simulation as a ground station that acted as live satellite data would for a real fire. The algorithm
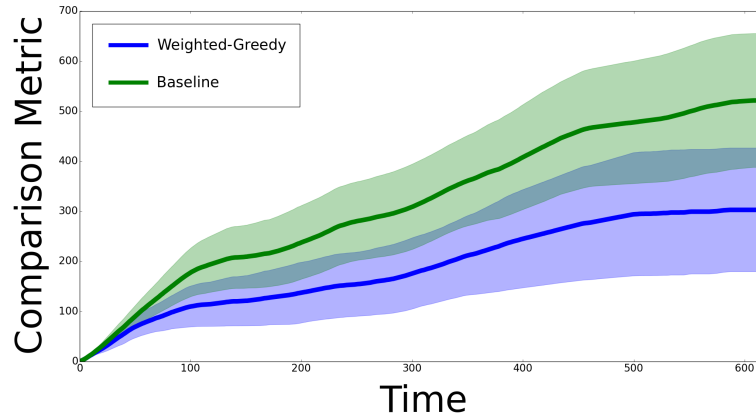
(a) Weighting parameter α = .5 Hotspot Threshold β = .45

Fig. 6: Wildfire simulation, where the comparison metric is J(t) or the sum of max time untracked of all hotspots. Lower is better. The weighting parameter α is set at .5. The normalized threshold β, for a spot along the fire to be considered an interest point, is set to .45. A lower β corresponds to more hotspot locations. Error bars are one SEM.

then sent a live stream of coordinates to a UAV to monitor the fire. While a real fire was not used for purpose of this test (for safety reasons), we are able to demonstrate that a UAV can effectively perform these tasks.

We converted FLAME into a ROS package to use the MAVROS plugins (Quigley et al, 2009). MAVROS acted as a communication bridge between FLAME and the flight controller on the UAV. This allowed us to update the UAVs path in time with the simulation. We used a tethered IRIS+ quadcopter as the platform for these experiments 7. We ran the experiment for over 10 minutes, about half the max flight time of the vehicle. The experiment was performed outdoors in about a 60ft x 60ft area. The simulation coordinates were scaled and transformed to GPS degrees to support sending waypoints. We present the path of the vehicle around the fire in figure 8.

The UAV successfully followed the trajectories generated in the simulation to the best locations along the fire to monitor it as it spread. This illustrates our ability to begin introducing robotic monitoring into these dynamic monitoring situations and gather valuable data from it.

(a) A 3d view of our flight log in Google Earth showing the flight distance and trajectories.

(b) Experiment set up, computer running live simulation and IRIS+ flying autonomously (tethered).

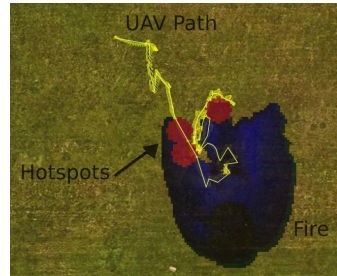Fig. 7: Experimental setup and flight log results.

(a) The UAV first starts off a safe distance away from the fire and must travel to the frontier.

(b) Upon reaching the frontier and identifying a hotspot the UAV stays outside the burn area as it grows.

(c) The UAV moves from one hotspot to another to reduce the time untracked.

(d) Final fire size and flight log of our field experiments.

Fig. 8: Four images demonstrating the algorithm path planning during field tests.

8 Conclusion

In this paper, we have introduced FLAME, a simulation developed for testing monitoring techniques on a dynamic frontier, or more specifically a wildfire. The two algorithms tested in the simulation have demonstrated that there is significant ben-

efit in a weighted-greedy over the baseline method of flying around the fire frontier. Using Mini-Batch K-Means Clustering for identifying hotspots, our proposed weighted-greedy algorithm optimized for $J(t)$, the sum of max time untracked of all hotspots. Three different normalized hotspot thresholds (β) (.25, .35, .45) were used. Data results showed the weighted-greedy algorithm with significant improvements over the baseline.

These algorithms depend on global knowledge of the fire, or more specifically where the hotspots are. Future work will include implementing a probabilistic model of hotspot locations and studying the exploration/exploitation trade-off for tracking and updating the model. In this paper we assume the UAVs have unlimited flight time. However, the cost of flight with limited endurance is an important factor. Additionally, hotspots are not all equal, and things such as risk to critical areas will need to be considered. Continuation of the project will also focus on implementation of multiple UAVs and the introduction of common fire monitoring challenges, including smoke and adverse weather conditions.

References

(2014) Farsite: Fire, fuel and smoke. URL `http://www.firelab.org/project/farsite`, accessed: 2014-09-27

Bajcsy R (1988) Active perception. Proceedings of the IEEE 76(8):966–1005

Bertozzi AL, Kemp M, Marthaler D (2005) Determining environmental boundaries: Asynchronous communication and physical scales. In: Cooperative Control, Springer, pp 25–42

Casbeer DW, Beard R, McLain T, Li SM, Mehra RK (2005) Forest fire monitoring with multiple small uavs. In: Proceedings of the American Control Conference 2005, IEEE, pp 3530–3535

Cassandras C, Ding XC, Lin X (2011) An optimal control approach for the persistent monitoring problem. In: Decision and Control and European Control Conference (CDC-ECC), 2011 50th IEEE Conference on, pp 2907–2912

Cassandras C, Lin X, Ding X (2013) An optimal control approach to the multi-agent persistent monitoring problem. Automatic Control, IEEE Transactions on 58(4):947–961

Center NIF (2015) Federal fire fighting costs. URL `http://www.nifc.gov/fireInfo/fireInfo_documents/SuppCosts.pdf`, accessed: 2014-09-26

Dunbabin M, Marques L (2012) Robots for environmental monitoring: Significant advancements and applications. IEEE Robotics & Automation Magazine 19(1):24–39

Group NWC (1996) Wildland fire suppression tactics reference guide. URL `http://www.coloradofirecamp.com/suppression-tactics/suppression-tactics-guide.pdf`, accessed: 2014-09-26

Hollinger G, Choudhary S, Qarabaqi P, Murphy C, Mitra U, Sukhatme G, Stojanovic M, Singh H, Hover F (2012) Underwater data collection using robotic sensor networks. IEEE Journal on Selected Areas in Communications 30(5):899–911

Hollinger GA, Sukhatme G (2013) Sampling-based motion planning for robotic information gathering. In: Robotics: Science and Systems

Koulas CE (2009) Extracting wildfire characteristics using hyperspectral, lidar, and thermal ir remote sensing systems. In: SPIE Defense, Security, and Sensing, pp 72,983Q–72,983Q

Kremens R, Seema A, Fordham A, Luisi D, Nordgren B, VanGorden S, Vodacek A (2001) Networked, autonomous field-deployable fire sensors. Proceedings of the international wildland fire safety summit

Lan X, Schwager M (2013) Planning periodic persistent monitoring trajectories for sensing robots in gaussian random fields. In: Robotics and Automation (ICRA), 2013 IEEE International Conference on, pp 2415–2420

Lloyd S (1982) Least squares quantization in pcm. IEEE Transactions on Information Theory 28(2):129–137

Marthaler D, Bertozzi AL (2004) Tracking environmental level sets with autonomous vehicles. In: Recent developments in cooperative control and optimization, Springer, pp 317–332

Quigley M, Conley K, Gerkey B, Faust J, Foote T, Leibs J, Wheeler R, Ng AY (2009) Ros: an open-source robot operating system. In: ICRA workshop on open source software, vol 3, p 5

Smith RN, Schwager M, Smith SL, Jones BH, Rus D, Sukhatme GS (2011) Persistent ocean monitoring with underwater gliders: Adapting sampling resolution. Journal of Field Robotics 28(5):714–741

Smith SL, Rus D (2010) Multi-robot monitoring in dynamic environments with guaranteed currency of observations. In: 2010 49th IEEE Conference on Decision and Control (CDC), IEEE, pp 514–521

Smith SL, Schwager M, Rus D (2012) Persistent robotic tasks: Monitoring and sweeping in changing environments. IEEE Transactions on Robotics 28(2)

Soltero DE, Smith S, Rus D (2011) Collision avoidance for persistent monitoring in multi-robot systems with intersecting trajectories. In: Intelligent Robots and Systems (IROS), 2011 IEEE/RSJ International Conference on, pp 3645–3652

Susca S, Bullo F, Martínez S (2008) Monitoring environmental boundaries with a robotic sensor network. IEEE Transactions on Control Systems Technology 16(2):288–296

Wald A (1945) Sequential tests of statistical hypotheses. The Annals of Mathematical Statistics 16(2):117–186