AN OPTIMAL CONTROL APPROACH TO WILDFIRE RESCUE

M. FENNIMORE, J. HOAGLAND, E. SAMPSON, M. SHUMWAY

Abstract. In this project, we explore optimal path planning in the context of wildfire rescue, where fires evolve dynamically over time. Using Pontryagin's Maximum Principle, we solve a boundary value problem to determine rescue paths that minimize time, risk, and control input. We begin with a Gaussian model for fire spread, whose mean and variance evolve in the direction of wind, and address challenges such as preserving fire intensity, incorporating wind penalties, and modeling realistic wind patterns. We extend the problem to include multiple rescue targets, using a heuristic to prioritize them based on danger. Finally, we outline a more physically accurate fire model governed by a convection–diffusion–reaction PDE, which offers a promising direction for future work.

1. Problem Statement and Motivation

Optimal path planning is a fundamental application of control theory, with broad relevance across many domains. In this work, we focus on optimal path planning in the context of wildfire rescue. Specifically, we aim to compute the optimal trajectory of an agent tasked with rescuing people, animals, or valuable assets, while avoiding areas threatened by active wildfires. The objective is to maximize rescue efficiency and safety, ensuring timely extraction before the fire reaches the targets. Our approach is grounded in Pontryagin's maximum principle (PMP), which provides necessary conditions for optimality in control systems.

Wildfire avoidance is a critical and timely challenge. While the methods presented here are not yet suitable for immediate real-world deployment, we believe they offer a principled foundation for applying optimal control techniques to wildfire rescue scenarios.

Importantly, the framework we develop, which is centered on dynamically evolving obstacle avoidance, is not exclusive to wildfire settings. The key distinction between this and other applications, such as crowd navigation, traffic-aware autonomous driving, or evacuation planning during natural disasters, lies primarily in the modeling of the dynamic obstacles. This highlights the adaptability of our approach across diverse real-time, safety-critical environments.

2. Mathematical Formulation

This section introduces the mathematical model for the agent's control dynamics and fire obstacles.

2.1. **System Dynamics and Cost Functional.**

To achieve the goal of safely and efficiently reaching a target, the optimal path must balance minimizing travel time, avoiding hazardous fire zones, and limiting control effort. To formalize this, we define the following cost functional:

Date: April 16, 2025.

$$J[u] = \underbrace{\int_0^{t_f} \alpha dt}_{\text{time cost}} + \underbrace{\int_0^{t_f} \sum_{i=1}^{K} \beta_i F_i(t, x(t), y(t)) dt}_{\text{obstacle cost}} + \underbrace{\int_0^{t_f} \frac{\lambda}{2} \|u(t)\|^2 dt}_{\text{control input cost}} \tag{1}$$

where

- $u(t) \in \mathbb{R}^2$ is the control input applied at time t.
- $(x(t), y(t)) \in \mathbb{R}^2$ denotes the position of the agent at time t.
- $F_i(t, x, y) \in \mathbb{R}$ represents the fire-induced cost associated with obstacle i at position (x, y) and time t.
- $\alpha, \beta_i, \lambda \in \mathbb{R}_{\geq 0}$ are scalar weights that define the relative importance of each component of the cost functional.
- $t_f \in \mathbb{R}_{\geq 0}$ is the final time, which is free and also optimized over.

We assume that the controller directly manipulates the agent's acceleration in the x- and y-directions. As such, we define the control inputs u_1, u_2 to be $\ddot{x}$ and $\ddot{y}$. This motivates the following state transition equations.

$$\frac{d}{dt} \begin{bmatrix} x & y & \dot{x} & \dot{y} \end{bmatrix}^\top = \begin{bmatrix} \dot{x} & \dot{y} & u_1 & u_2 \end{bmatrix}^\top \tag{2}$$

For clarity in what follows, let $\mathbf{s} = \begin{bmatrix} x & y & \dot{x} & \dot{y} \end{bmatrix}^\top$. Assume that the agent begins at the origin and that the target is located at $(x_{\text{tar}}, y_{\text{tar}})$. Further, it is reasonable to assume that the agent begins from rest and also must end with zero velocity, so that it can interact with the target. This leads to the following boundary conditions.

$$\mathbf{s}(0) = \mathbf{0} \tag{3}$$

$$\mathbf{s}(t_f) = \begin{bmatrix} x_{\text{tar}} & y_{\text{tar}} & 0 & 0 \end{bmatrix}^\top \tag{4}$$

2.2. **Necessary Conditions with PMP.** To derive the necessary conditions for optimality, we apply PMP. This involves constructing the Hamiltonian, introducing the costate variables, and characterizing optimal trajectories and controls.

The Hamiltonian corresponding to the cost function (1) is:

$$H = p_1 \dot{x} + p_2 \dot{y} + p_3 u_1 + p_4 u_2 - \alpha - \sum_{i=1}^{K} \beta_i F_i(t, x, y) - \frac{\lambda}{2} \left(u_1^2 + u_2^2 \right) \tag{5}$$

Let $\mathbf{p}(t) = [p_1(t), p_2(t), p_3(t), p_4(t)]^\top$ denote the costate vector. Its dynamics are given by:

$$\dot{\mathbf{p}} = -\frac{\partial H}{\partial \mathbf{s}} = \begin{bmatrix} \sum_{i=1}^{K} \beta_i \frac{\partial F_i}{\partial x} & \sum_{i=1}^{K} \beta_i \frac{\partial F_i}{\partial y} & -p_1 & -p_2 \end{bmatrix}^\top \tag{6}$$

where $\mathbf{s} = [x, y, \dot{x}, \dot{y}]^\top$ is the state vector.

Since the final time t_f is free, the Hamiltonian must vanish at the endpoint:

$$H(t_f) = 0 \tag{7}$$

Because all state components are fixed at both the initial and final times, no conditions are imposed on the costate variables. We are left with a boundary value problem with nine

unknowns ($\mathbf{s}$, $\mathbf{p}$ and t_f) with nine conditions (given in 3, 4, 7). As such, we use Scipy's 'solve_bvp' method for determining the optimal solution.

Note that the above derivation assumes that the partial derivatives of each F_i exists, which we have intentionally left unspecified. This is addressed in the following section.

3. Modeling Fire

To begin, we consider the following desirable properties for a fire. First, the fire function should be differentiable, as this will allow for the computation of partial derivatives as required for the costate dynamics 6. Second, the fire function should be malleable, meaning that we can easily change the shape throughout time to resemble realistic fire movement. For example, a fire should grow in the direction of wind.

Initially, a clear functional choice that satisfies these requirements is the pdf of the Gaussian distribution.

3.1. Gaussian Fire Model.

We model each fire as a time-evolving bivariate Gaussian distribution. For clarity, we omit the subscript F_i and present the formulation for a single fire. The extension to multiple fires is straightforward.

$$F(t, x(t), y(t)) \sim \mathcal{N}\big(\mu(t), \Sigma(t)\big) \tag{8}$$

The mean $\mu(t)$ represents the fire's center and evolves linearly in time, driven by wind:

$$\mu(t) = \begin{bmatrix}\mu_x(t)\\ \mu_y(t)\end{bmatrix} = \begin{bmatrix}\mu_{x_0} + w_x t\\ \mu_{y_0} + w_y t\end{bmatrix}$$

Here, denote μ_{x_0}, μ_{y_0} as the initial center of the fire and $\mathbf{w} = \begin{bmatrix} w_x & w_y\end{bmatrix}^\top$ is the wind vector (which is possibly a function of time). This choice reflects the intuitive behavior of fire drifting along with the wind.

To capture the spread of the fire, we define a time-varying covariance matrix. Let $\sigma_1^2(t)$ and $\sigma_2^2(t)$ denote the variances along and perpendicular to the wind direction, respectively. We model their evolution as

$$\sigma_1^2(t) = \sigma_{1_0}^2 + kt$$
$$\sigma_2^2(t) = \sigma_{2_0}^2 + ct$$

where $\sigma_{1_0}^2, \sigma_{2_0}^2$ are initial variances and $k, c \in \mathbb{R}$ control how the rate of fire spread along and across the direction of the wind. It is sensible to choose these such that $c \leq k$. These terms capture the desired anisotropic and time-dependent fire growth, expressed in the basis aligned with the wind vector $\mathbf{w}$. In this coordinate system, the covariance matrix is diagonal:

$$\Sigma' = \begin{bmatrix}\sigma_1^2 & 0\\ 0 & \sigma_2^2\end{bmatrix}$$

To express this variance in the standard coordinate system, we apply a rotation. First, normalize the wind vector:

$$(u_x, u_y) = \frac{(w_x, w_y)}{\|\mathbf{w}\|}$$

Define the rotation matrix R as:

$$R = \begin{bmatrix} u_x & -u_y \\ u_y & u_x \end{bmatrix}$$

Here, the first column of R is the unit vector in the direction of the wind, and the second column is the unit vector perpendicular to it. This rotation transforms the aligned covariance back into the standard basis:

$$\Sigma(t) = R\Sigma' R^T$$

An example of such fire is displayed in Figure 1. Very clearly, the cost of being in the center of the fire decays very quickly through time. This is a problem we address below in Section 4.

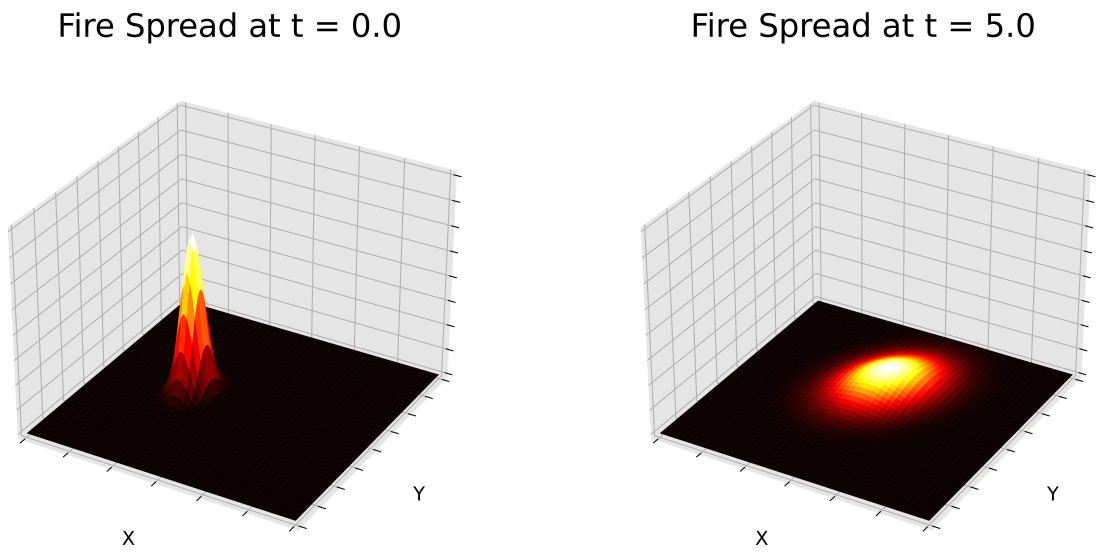


FIGURE 1. An example of fire under the original Gaussian model. Problematically, the cost of being inside the fire decays over time. Wind was taken to be constant with $\mathbf{w} = (1, 1/2)$.

Despite the problem of decaying cost, using this initial fire model and the methods outlined in Section 2.2 still resulted in successful solutions. For one example, see Figure 2.

4. Modeling Improvements

Despite the promising results from the initial cost functional and fire functions, several concerns were brought to our attention.

(1) Decay in the cost of the center of the fire over time. As the variance is increasing in time and the pdf of the Gaussian will always integrate to one, the relative cost of the distribution's peak will decay over time. This could lead to undesired behavior, where the agent eventually passes directly through the center of the fire. Figure 1 demonstrates the decay of the fire over time.

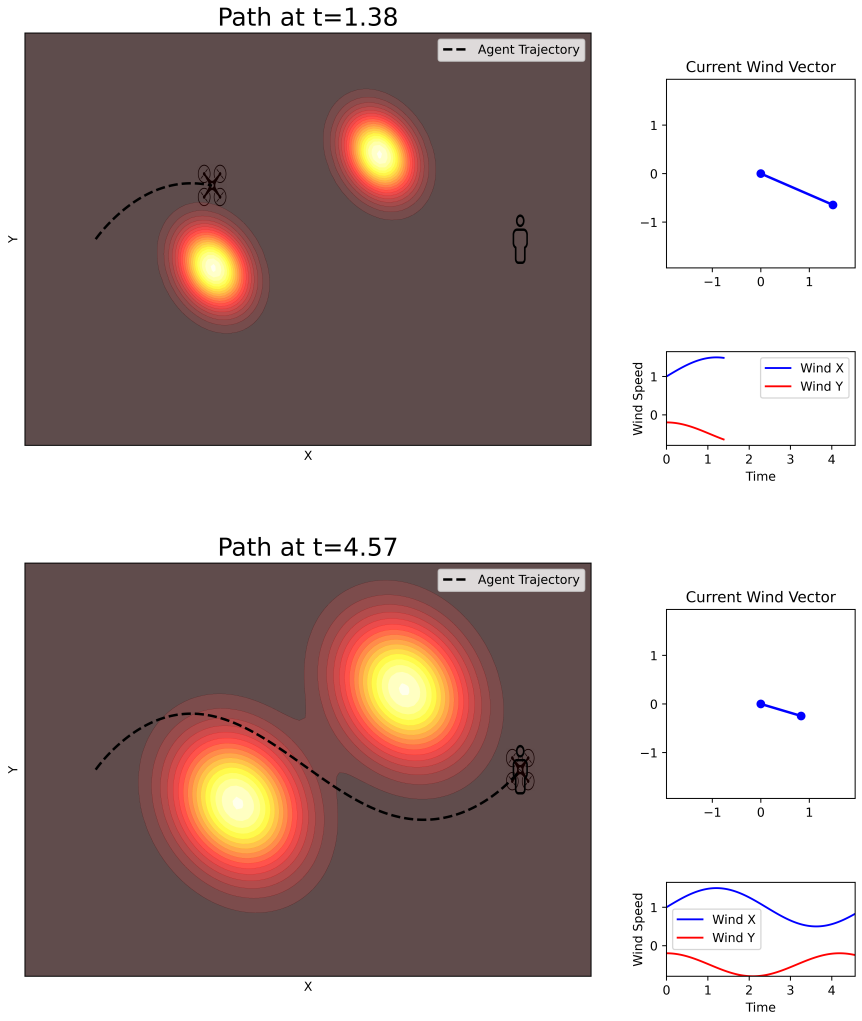


FIGURE 2. A solution with two fires and varying wind under the initial Gaussian fire model (unscaled). Despite the decay in fire cost, the agent still avoids both fires. The wind was given by $\mathbf{w}(t) = (1 + 0.5\sin(1.3t), -0.5 + 0.3\cos(1.5t))$. The current wind vector and overall wind trajectory is plotted.

(2) Wind not effecting the flight of the drone. Our cost functional did not punish the drone for flying against the wind.
(3) The functions defining the wind were not realistic enough. As seen in the bottom right of figure 2, the functions defining the wind were simple trigonometric functions. While fairly realistic in the short term, once it had cycled past it's period, the fires appeared to undulate.

4.1. **Adaptive Fire Scaling.** To address problem (1), we needed to scale the fire density function to prevent it from decaying over time. More precisely, we required the peak of the probability density function to maintain a constant value over time.

To enforce that the maximum of the fire distribution remains 1 for all t, we define a time-dependent scaling factor $h(t)$. We impose the condition

$$1 = h(t)F(\mu)$$

where $F(\mu)$ is the value of the fire pdf at its mean μ. Solving for $h(t)$, we obtain:

$$h(t) = \frac{1}{F(\mu)}.$$

By multiplying the original fire distribution F by this factor $h(t)$, we ensure that the peak value remains constant in time, thereby preventing the fire from "eroding" as it spreads.

See Figure 3 for a visualization of this rescaled fire model.

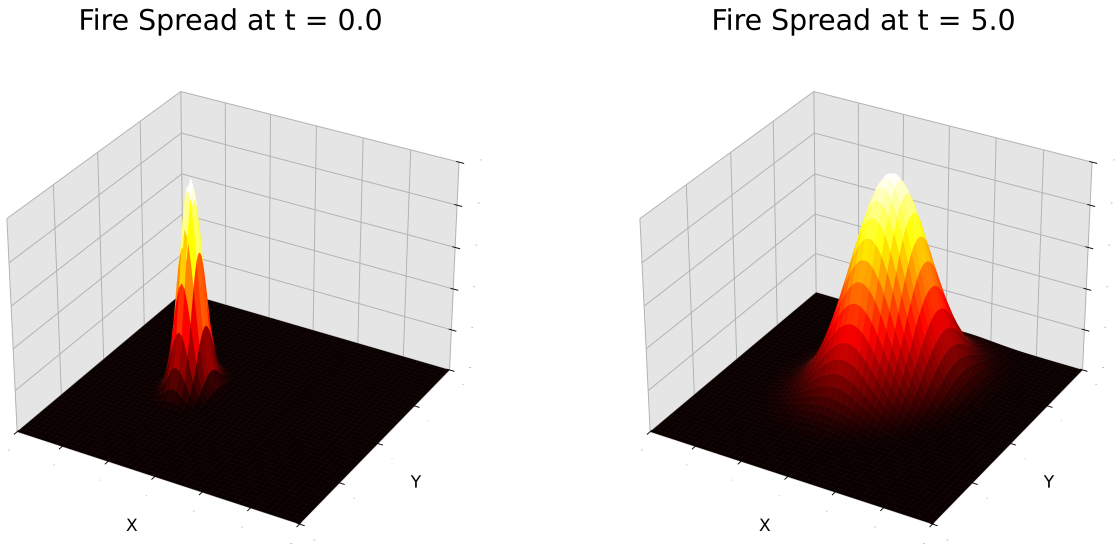


FIGURE 3. An example of the fire under the scaled Gaussian model. Now the fire will 'retain' it's heat with time and not shrink. Wind was taken to be constant with $\mathbf{w} = (1, 1/2)$.

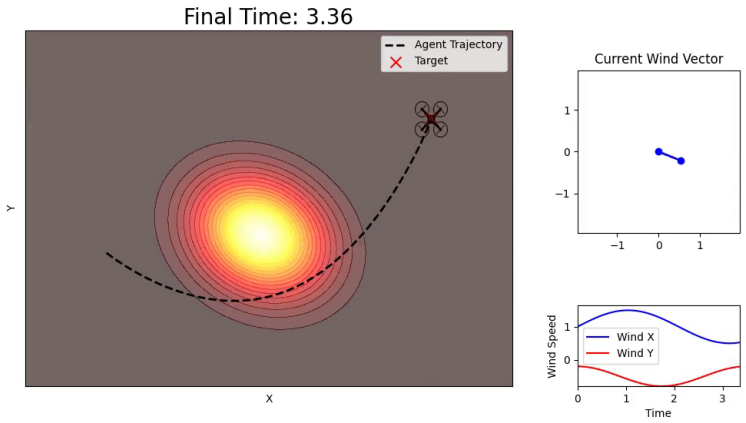
4.2. **Punishing Movement Against Wind.** To address problem (2), we modified the cost functional to penalize the drone for flying against the direction of the wind. This was accomplished by incorporating the wind components directly into the control terms:

$$\begin{aligned} \ddot{x} = u_1 \quad &\longrightarrow \quad u_1 + \gamma w_x \\ \ddot{y} = u_2 \quad &\longrightarrow \quad u_2 + \gamma w_y \end{aligned}$$

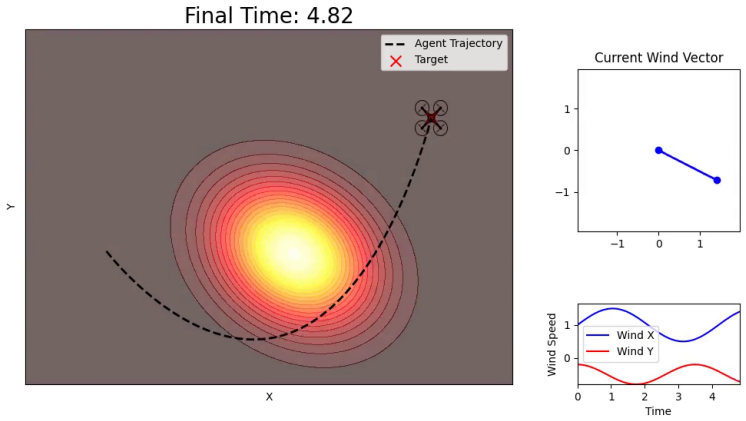
In the presence of wind, the drone must exert more effort to fly against it (i.e., when u_1 and w_x have opposite signs), and less effort when flying with it (i.e., when u_1 and w_x share the same sign), in order to achieve the same net acceleration. The coefficient γ controls how strongly the wind affects the drone's motion.

By embedding the wind terms alongside the control inputs, we effectively shift the frame of reference to the air, rather than the ground. This adjustment more accurately reflects the true energy cost of flying the drone under varying wind conditions.

See Figure 4 for a visualization of this behavior.



(A) $\gamma = 0$



(B) $\gamma = 10$

FIGURE 4. Two plots of the final optimal path for $\gamma = 0$ and $\gamma = 10$. As seen in the lower plot, the higher γ value influenced the path of the drone and caused to flight to take longer.

4.3. **Alternative Wind Functions.** To improve upon our earlier wind model (see Figure 2), we introduced additional linear terms to capture long-term directional trends and gradual changes in wind strength over time. The resulting wind model is given by:

$$w_x(t) = w_{x_0} + c_x t + \sin(k_x t)$$
$$w_y(t) = w_{y_0} + c_y t + \sin(k_y t)$$

Here, the sinusoidal components model short-term oscillations, while the linear terms $c_x t$ and $c_y t$ account for slow, cumulative shifts in wind over the course of the mission.

The effects of this change are visualized in Figure 5.

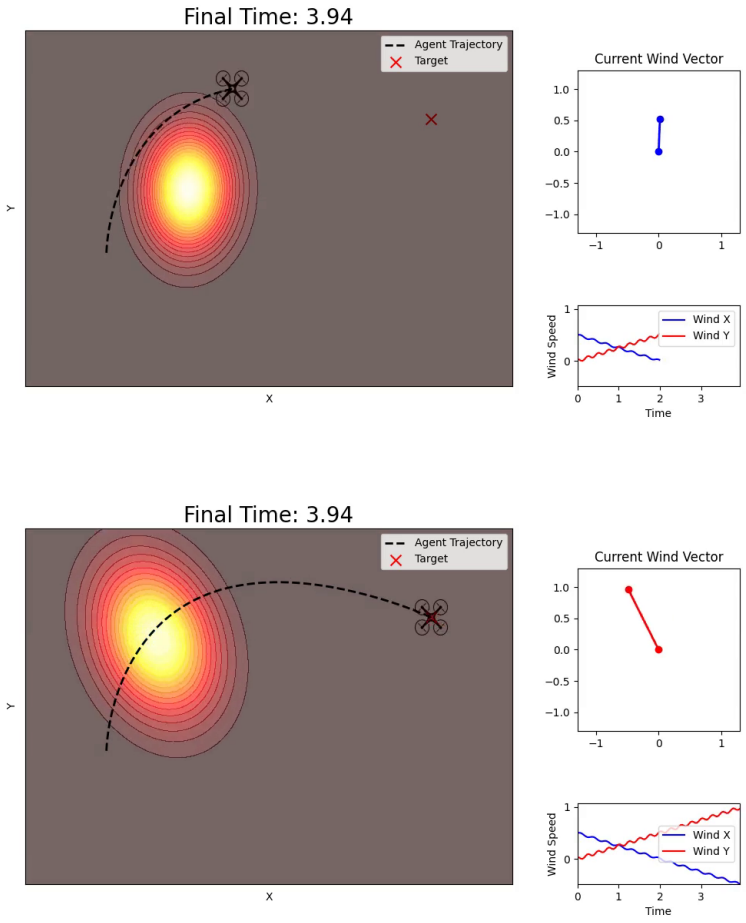


FIGURE 5. An example trajectory generated using the improved wind model from Section 4.3, which incorporates both oscillatory and long-term directional wind behavior. The drone still finds the optimal path, but through a slightly more realistic environment.

5. Extension: Multiple Targets

We extend our basic obstacle-avoidance approach to handle multiple targets and require that the agent return to its starting location upon completion. This models scenarios such as rescuing multiple individuals or retrieving valuable items, with the goal of returning home safely.

A central challenge in this problem is determining the order in which to visit the targets. While this can be computationally demanding in general, we introduce a simple heuristic algorithm that prioritizes targets based on a notion of "danger." Specifically, we assess whether each target is at risk by constructing a 45° cone originating from each fire location and oriented in the direction of the wind. If a target lies within such a cone, it is considered endangered, and its priority is increased by scaling down its effective distance using a

hyperparameter $\alpha \in (0, 1)$. This encourages the agent to rescue endangered targets sooner. The full procedure is outlined in Algorithm 1.

Algorithm 1 HOAGY: Heuristic Obstacle-Avoiding Greedy Yielder

Input: Set of targets $T = \{t_1, t_2, \ldots, t_n\}$, initial position x_0, fire centers $F = \{f_1, f_2, \ldots\}$, wind function $\vec{w}(t)$, danger scaling factor $\alpha \in (0, 1)$
Initialize: $x \leftarrow x_0$, visited $\leftarrow \emptyset$
while there are unvisited targets **do**
    for each unvisited target $t \in T \setminus$ visited **do**
        $d_t \leftarrow \|t - x\|$ ▷ Euclidean distance from current position
        danger $\leftarrow$ **False**
        for each fire center $f \in F$ **do**
            $\vec{w}_f \leftarrow \vec{w}(t)$
            Define cone $\mathcal{C}_f$ originating at f, opening 45° in direction $\vec{w}_f$
            if $t \in \mathcal{C}_f$ **then**
                danger $\leftarrow$ **True**
                break
            end if
        end for
        if danger **then**
            $d_t \leftarrow d_t - \alpha$ ▷ Boost priority of endangered targets
        end if
    end for
    Select target t^* with minimum adjusted distance d_{t^*}
    Solve BVP from current position x to t^* using `solve_bvp`
    Update:
    - Current position $x \leftarrow t^*$
    - Fire centers F as needed
    - Wind vector $\vec{w}(t)$ for next planning phase

    Mark t^* as visited
end while
Solve final BVP from current position x back to initial position x_0

While this algorithm may be improved upon and is possible future research direction, our experiments yielded successful results. A visualization of one possible scenario is given in Figure 6.

6. A More Realistic Model of Fire: Convection-Diffusion-Reaction

While our Gaussian model of fire, presented in Section 3.1, yielded optimal solutions, we acknowledge that it is not the most physically realistic model. We introduced dynamic scaling so that the cost of being in the center of the fire no longer decays; however, problems still remain. For example, the edges of the fire are still the coldest regions, whereas in reality, they may be the hottest, as these areas are exposed to fresh fuel and air. Additionally, the fact that the fire's entire mass shifts away from its original location oversimplifies real fire dynamics—its influence would likely persist in previously burned areas for a longer period.

One approach that is both more physically realistic and addresses the issues described above is inspired by the work in [2, 1]. Though ambitious, the central idea of this method is

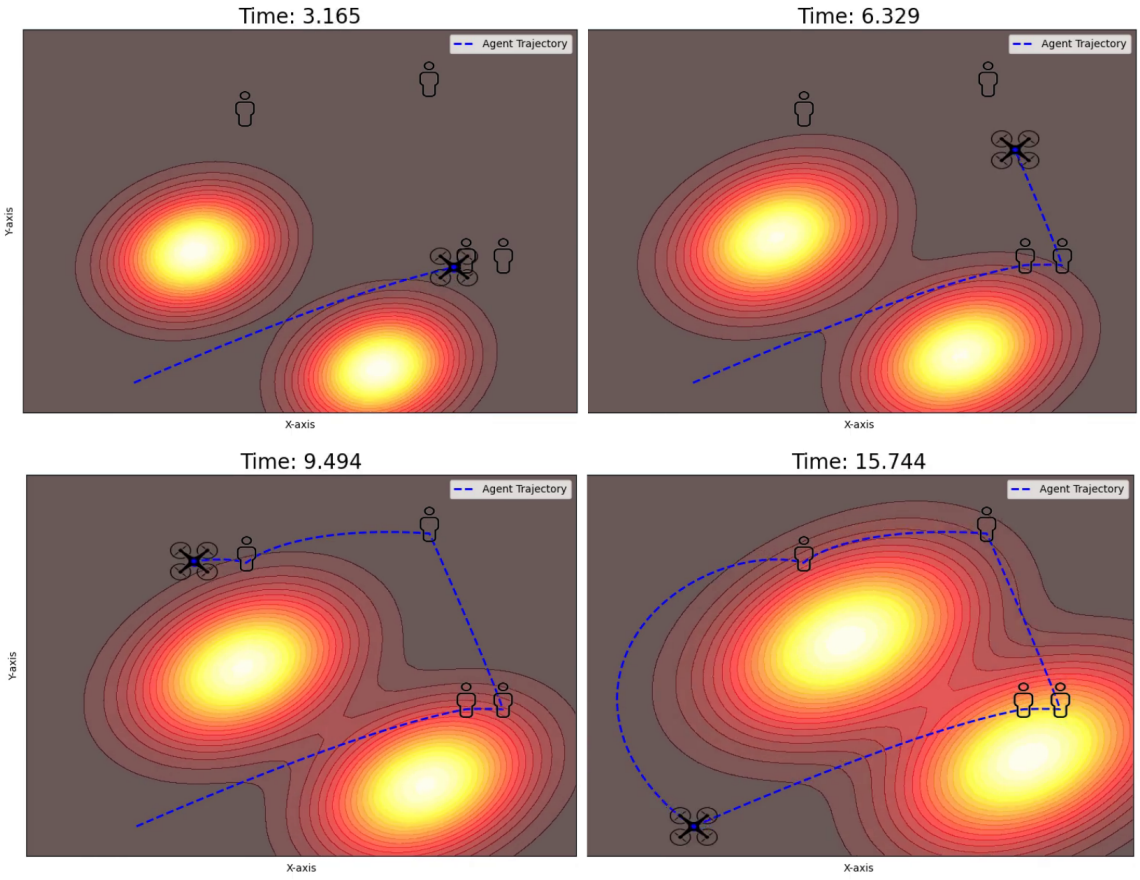


FIGURE 6. A possible optimal solution to a scenario with four targets and two fires. Wind was chosen to be constant with $\mathbf{w} = (2, 1)$. The order of pickup was determined by Algorithm 1.

to model the fire as a solution to a convection–diffusion–reaction partial differential equation (PDE):

$$\frac{\partial \xi}{\partial t} + \nabla \cdot (\mathbf{w}(t, \mathbf{x})\xi) = \nabla \cdot (K(\xi)\nabla\xi) + f(\xi, v, \mathbf{x}), \quad \frac{\partial v}{\partial t} = g(\xi, v) \tag{9}$$

where t is time, $x \in \Omega \subseteq \mathbb{R}^2$ is the spatial variable representing the forest in which the fire may propagate, and $\xi = \xi(t, \mathbf{x})$ and $v = v(t, \mathbf{x})$ are the scalar unknowns. Here ξ represents the non-dimensionalized temperature and v the non-dimensionalized mass fraction of solid fuel. Additionally, $K = K(\xi)$ is a given diffusion coefficient, and $\mathbf{w}$ is the advection velocity representing wind speed. The functions $f(\xi, v, \mathbf{x})$ and $g(\xi, v, \mathbf{x})$ are the reactive part of the model. Reasonable boundary and initial conditions are

$$(\xi\mathbf{w} - K(\xi)\nabla\xi) \cdot \mathbf{n} = 0, \qquad (t, \mathbf{x}) \in (0, +\infty) \times \partial\Omega \tag{10}$$

where $\mathbf{n}$ is the unit normal vector to $\partial\Omega$, and the initial conditions

$$\xi(0, \mathbf{x}) = \xi_0(\mathbf{x}), \quad v(0, \mathbf{x}) = v_0(\mathbf{x}), \quad x \in \Omega \tag{11}$$

The boundary conditions are referred to as zero-flux, which can be interpreted in our setting as (1) no heat can leave or enter the region through the boundary and (2) no fuel is supplied or lost through the boundary (as fuel is confined to the forested region). We believe these are reasonable as it ensures the fire can only spread within the region of interest.

An algorithm for numerically approximating solutions ξ, v to 9 is provided in [2]. Although we attempted to replicate this approach, we were ultimately unsuccessful. This remains a key area of interest for future work.

7. Conclusions and Next Steps

In this project, we explored the application of PMP in path planning for wildfire rescues. By modeling the dynamics of the fire and of the movemnt of the drone, we were able to derive the optimal paths to minimize time, energy and risk exposure. We modeled the fires with dynamic Gaussian distributions that moved in the direction of wind. While we got promising results with this initial model, we improved upon our techniques by dynamically scaling the distributions to prevent unrealistic cost decay, penalizing drone flight in the direction of the wind, and explored more realistic approaches to modeling wind. After incorporating these improvements, we developed a heuristic algorithm that plans an efficient path for visiting multiple targets and returning to the starting position.

Future work on this problem could explore the more realistic model of the fire as given in Section 6. While the HOAGY algorithm (1) produced strong results, alternative algorithms could be developed to prioritize different objectives or improve computational efficiency. Additionally, one could extend the model to three dimensions to better simulate a drone's flight through the air. Also, using principles of reinforcement learning, future work could incorporate uncertainty into the positions of the targets and fires. Lastly, it would be interesting to extend the framework to support multi-agent systems for coordination among multiple rescuers.

8. Code

All relevant code for replicating these experiments can be located on our shared GitHub page linked here.

References

[1] Raimund Bürger, Elvis Gavilán, Daniel Inzunza, Pep Mulet, and Luis Miguel Villada. Exploring a convection–diffusion–reaction model of the propagation of forest fires: Computation of risk maps for heterogeneous environments. *Mathematics*, 8(10):1674, 2020.

[2] Raimund Bürger, Elvis Gavilán, Daniel Inzunza, Pep Mulet, and Luis Miguel Villada. Implicit-explicit methods for a convection-diffusion-reaction model of the propagation of forest fires. *Mathematics*, 8(6):1034, 2020.