

Article

Decentralized Multi-Agent Search for Moving Targets Using Road Network Gaussian Process Regressions

Brady Moon ^{1,2,*} , Christine Akagi ²  and Cameron K. Peterson ^{2,*} ¹ Robotics Institute, Carnegie Mellon University, Pittsburgh, PA 15289, USA² Department of Electrical Engineering, Brigham Young University, Provo, UT 84602, USA; cmakagi@gmail.com

* Correspondence: bradygmoon@cmu.edu (B.M.); cammy.peterson@byu.edu (C.K.P.)

Abstract: Unmanned aerial vehicles (UAVs) can collaborate as teams to accomplish diverse mission objectives, such as target search and tracking. This paper introduces a method that leverages accumulated target-density information over the course of a UAV mission to adapt path-planning rewards, guiding UAVs toward areas with a higher likelihood of target presence. The target density is modeled using a Gaussian process, which is iteratively updated as the UAVs search the environment. Unlike conventional search algorithms that prioritize unexplored regions, this approach incentivizes revisiting target-rich areas. The target-density information is shared across UAVs using decentralized consensus filters, enabling cooperative path selection that balances the exploration of uncertain regions with the exploitation of known high-density areas. The framework presented in this paper provides an adaptive cooperative search method that can quickly develop an understanding of the region's target-dense areas, helping UAVs refine their search. Through Monte Carlo simulations, we demonstrate this method in both a 2D grid region and road networks, showing up to a 26% improvement in target density estimates.

Keywords: unmanned aerial vehicles; search and tracking; Gaussian process regression; mission planning



Citation: Moon, B.; Akagi, C.; Peterson, C.K. Decentralized Multi-Agent Search for Moving Targets Using Road Network Gaussian Process Regressions. *Drones* **2024**, *8*, 606. <https://doi.org/10.3390/drones8110606>

Academic Editor: Pablo Rodríguez-González

Received: 1 August 2024

Revised: 19 October 2024

Accepted: 20 October 2024

Published: 23 October 2024



Copyright: © 2024 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (<https://creativecommons.org/licenses/by/4.0/>).

1. Introduction

The demand for unmanned aerial vehicles (UAVs) continues to grow due to their wide range of practical applications and advantageous qualities. UAVs offer numerous operational benefits [1–6], are typically more cost-effective than manned alternatives, and protect pilots by keeping them out of dangerous situations [7]. Deploying multiple UAVs in a coordinated manner can significantly enhance their collective efficiency and impact, such as allowing them to gather information more quickly and comprehensively within a given area.

Patrolling, search, and surveillance operations face considerable challenges when there are not enough UAVs to cover the entire area simultaneously. In these situations, UAVs must coordinate their trajectories to maximize area coverage as efficiently as possible. While previous research has focused on developing cooperative search algorithms, it has often overlooked the impact of uneven target distributions in unknown environments. When targets are distributed unevenly across a region, it becomes crucial to adapt the search strategy accordingly. In areas with low target densities, the likelihood of finding valuable information is minimal, and thus, frequent visits to these regions would lead to an inefficient use of resources. Conversely, regions with a higher concentration of targets offer greater potential for discovery, and UAVs should prioritize these areas to maximize mission success. Without adapting to these variations, UAVs risk wasting time and energy on low-value areas, which could hinder overall mission efficiency. Therefore, an optimal search approach would allocate more time and effort to high-probability zones, dynamically adjusting based on learned information about the environment. This is especially relevant in scenarios where uneven target distributions are expected, such as when searching large regions that include both sparsely populated rural areas and heavily populated urban regions.

In this paper, we provide novel cooperative search algorithms that dynamically adapt the path-planning algorithm based on gathered information while it operates in areas with uneven target distributions. The search algorithm learns the target-density environment using Gaussian process (GP) regressions and uses this information to drive UAV behavior. A depiction of this concept is shown in Figure 1. In our approach, UAVs prioritize revisiting areas that are highly likely to provide information on new or existing targets while minimizing time spent in regions where targets are rarely observed. The balance between exploiting regions with a high likelihood of valuable information and exploring new or lower-priority areas is guided by the uncertainty in the GP model. This method is particularly effective in search regions with nonuniform target densities and frequencies, such as road networks with highways or areas that encompass both sparsely and densely populated regions.

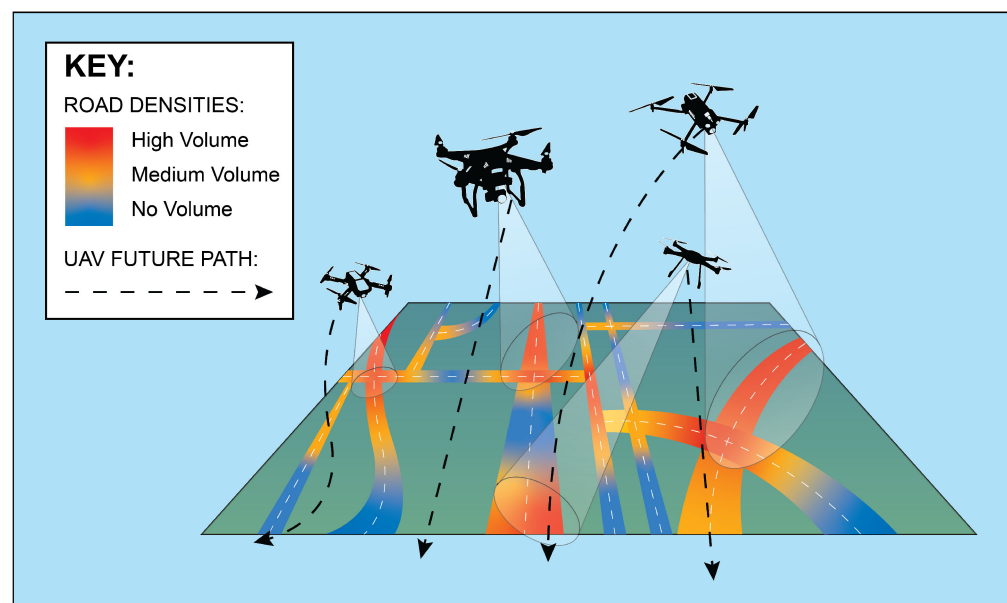


Figure 1. This figure depicts the research objective to efficiently search for targets using UAVs by predicting the target densities along the road networks based on their history of observations.

This paper extends the authors' prior published conference proceeding [8] which applied GP regressions in a centralized implementation to a search region by segmenting the area into grid cells. We expand this approach by also providing a decentralized road network algorithm. The road network approach provides several advantages over a grid cell approach, including the ability to (a) automatically learn the road network through open source information, (b) limit UAVs from searching areas where targets will not be found (i.e., off of roads), (c) provide a more accurate prediction of target locations, and (d) propagate target uncertainty along the road network through a novel kernel function applied in the GP regression.

Specifically, the contributions in this paper include:

- Creating search algorithms that balance the exploration and exploitation of a region in uneven target densities for both a grid region and road network.
- Automatic generation of a road network using open-source databases that may drive the search algorithm.
- Developing a novel kernel function that determines spatial closeness to points along the road network and thus provides a method to update the GP regression.
- Constructing a decentralized implementation of our search algorithm using consensus filters.

The paper is organized as follows. Section 2 presents a review of related works. In Section 3, we outline the overall path-planning approach and explain how cooperating vehicles optimize their search areas using a cost function. The core contributions are highlighted in Section 4, where we explain the method for calculating and updating road segment rewards based on learned information. In Section 5, we compare our approach with baseline methods across both grid-based and road network scenarios. Finally, Section 6 offers conclusions and a summary of the work.

2. Related Works

The performance of a search algorithm is generally assessed based on the mission's objectives, such as ensuring frequent and uniform coverage of an area or identifying and tracking dynamic targets within a specified region. Each objective comes with its own advantages and trade-offs. Frequency-driven methods aim to optimize the interval between visits to various locations in the search area. Some strategies prioritize reducing this interval, as highlighted in [9–11], while others aim for consistency in the elapsed time across the entire search grid, as discussed in [12,13]. Another approach emphasizes identifying the maximum number of high-interest locations in the shortest time, often employing probabilistic methods, such as in [14].

While these algorithms perform well according to their evaluation criteria, they often show limitation when applied to searching for and monitoring moving targets. Specifically, they may struggle to accurately map the targets or predict their behavior. In many scenarios, targets are not uniformly distributed across a search area, leading to UAVs spending disproportionate amounts of time searching regions with few or no targets.

In this work, we choose to model the target distributions using GP regressions, both for a grid region and road networks. GP regressions can model complex behavior without parameterized prior knowledge [15]. They also have the advantage of providing an uncertainty quantification along with a mean function defined over the entirety of the space [16]. This makes it an ideal tool for modeling target densities since targets exhibit complex, nonlinear patterns with sharp spatial boundaries (i.e., as occurs when they are confined to roads). GPs have been used previously to estimate wind fields for gliders and UAVs [17–19], discover patterns [20], and in model predictive control [21–23].

There are many approaches to multi-agent path routing, including both learned and analytical policies. For learned policies, some multi-agent path routing methods have used Reinforcement Learning (RL) for applications such as path finding [24–26], collaborative searching and tracking [27,28], air combat [29], network optimization [30,31], and area coverage [32,33]. Though these methods do show promise, in contrast to our method, they are often sample inefficient, require high computational power and memory, can lead to stochastic policies, and may need to be retrained when there are any modifications to the vehicle or environment models.

Other multi-agent path routing methods have used Mixed Integer Linear Programming to optimize UAV placement applications such as maximizing cellular coverage [34] or truck multi-drone routing over a directed graph [35]. The work by [36] chooses actions from a finite control set by evaluating all sets of actions up to a horizon and then uses a more efficient policy to evaluate the reward for future actions. This requires much less computation than an exhaustive search over sets of actions. Our multi-agent path routing method extends this work by combining an exhaustive search algorithm with a greedy-heuristic algorithm, while incorporating an adaptive threshold based on spatial diversity in the paths to determine when to switch between the two.

An important aspect of multi-agent path planning is providing algorithms capable of sharing information while increasing the number of cooperating vehicles. While many works rely on full communication between UAVs and centralized planning, for this work we assume that UAVs have a finite communication radius and are unable to connect to a centralized processing station. Assuming an undirected and connected communication topology, each UAV shares observation and target count numbers. We use a consensus filter

to calculate the average estimate of the shared information. Consensus filters are useful in driving vehicles to a common state [37]. In prior research, they have been used to estimate environmental information [38,39], track objects [40], determine the number of targets [41], and drive vehicles to formations [42]. Using a consensus filter enables our method to be decentralized and more robust to real-world communication networks.

3. Vehicle Path Planning

This section explains how our path planning algorithm operates and works to enable cooperative control between UAVs. Figure 2 illustrates the main components of the UAV framework, where each vehicle gathers information (target detections) from within its sensing field of view. Each UAV updates the parameters of its search-reward function by merging its local heat map data with those from peer UAVs, as described in Section 4. The vehicle's updated understanding of the environment is then used to determine its paths. This discrete process is repeated once all the vehicles have moved forward in time.

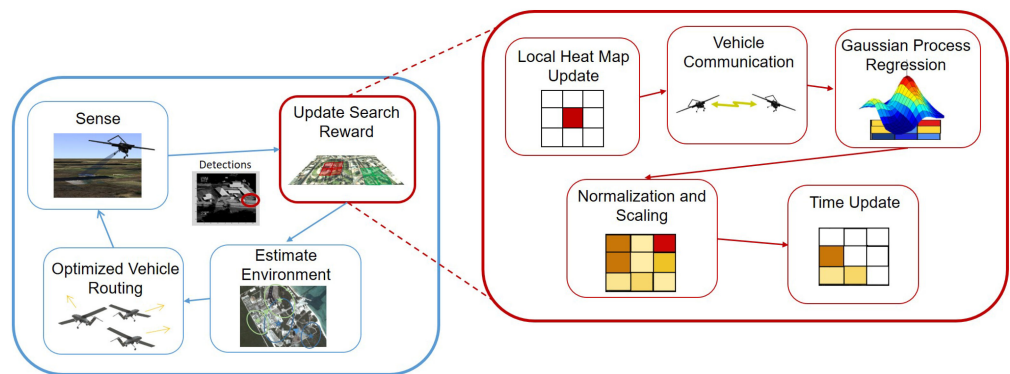


Figure 2. This figure provides the functional diagram of our cooperative search algorithm. This shows that once vehicles sense information from their environment and share their local common operating picture then this information is incorporated in the road segment rewards.

The rest of this section outlines the UAV model in Section 3.1 and the approach for determining vehicle paths using receding horizon control in Section 3.2.

3.1. Vehicle and Target Model

We model the UAVs as fixed-wing aircraft that travel at a fixed velocity and altitude. Each UAV is controlled with changes in heading achieved through a coordinated turn,

$$\dot{\psi}_a(k) = \frac{g \tan \phi_a(k)}{V_a(k)},$$

where g is the gravitational constant, $a \in [1, \dots, A]$ is the a th UAV, $\dot{\psi}_a(k)$ is the turn rate at time step k , $V_a(k)$ is the ground speed, and $\phi_a(k)$ is the roll angle. All aircraft have a saturation limit on their roll rate $|\dot{\phi}_a| \leq \dot{\phi}_{max}$.

The p th target's state at time step k is given by $x_p(k) = [x_p(k) \ y_p(k) \ \dot{x}_p(k) \ \dot{y}_p(k)]^T$, which represents the target's north-east-down (NED) Cartesian position and velocity. Targets are assumed to move in a fixed plane with the distinguishing relationship between them and the UAVs being the bearing and ground range. As such, UAV sensors measure the ground range and azimuth angle of targets in their sensor's field of view. These detections are converted from their range-azimuth measurement space to a Cartesian coordinate frame and used in an extended Kalman filter (EKF) to estimate the targets' positions.

The measurement noise $w_a \sim \mathcal{N}(0, R)$ of vehicle a is characterized by range and azimuth uncertainties, σ_r and σ_θ with noise covariance matrix given by $R = \text{diag}[\sigma_r^2, \sigma_\theta^2]$.

The measurement matrix that relates measurement space to state space for UAV a and target p is

$$H_{p,a}(k) = \begin{bmatrix} \frac{x_p(k) - x_a(k)}{r_{p,a}(k)} & \frac{y_p(k) - y_a(k)}{r_{p,a}(k)} & 0 & 0 \\ \frac{y_p(k) - y_a(k)}{r_{p,a}(k)} & -\frac{x_p(k) - x_a(k)}{r_{p,a}(k)} & 0 & 0 \end{bmatrix},$$

where $(x_a(k), y_a(k))$ is the planar position UAV a at time step k . Target p 's position is $(x_p(k), y_p(k))$. The ground range between target p and UAV a is $r_{p,a}(k) = \sqrt{(x_p(k) - x_a(k))^2 + (y_p(k) - y_a(k))^2}$.

A sensor measurement is

$$z_{p,a}(k) = H_{p,a}(k)\mathbf{x}_p(k) + \mathbf{w}_a.$$

Estimates of the target's state $\hat{\mathbf{x}}_{p,a}(k)$ and error covariance $P_{p,a}(k)$ are updated using an EKF as described in [43]. Sensed target information is assumed to be shared with all peer UAVs.

3.2. Cooperative Search

Our search mission is carried out by rewarding vehicles for sensing specific portions of the operating area, $\mathcal{M} \in \mathbb{R}^2$. The operating area may be subdivided in two different ways, either using a road map or dividing the area into equally spaced grid cells. The first method is advantageous if the road network is known and targets travel along it (i.e., no off-road maneuvers). The second method places no assumptions on target movement nor requires prior knowledge of the area. In this subsection, we describe the vehicle rewards and routing for each of these two approaches.

If not otherwise known, a road network may be obtained by exporting data from OpenStreetMap [44], which contains worldwide information. Major roads may be extracted from the data, while discarding trails, paths, and other lower-interest road segments to create a road-network graph $\mathcal{G} = \{V, S\}$ with vertices V and road segments S connecting the vertices. $D = \text{card}\{S\}$ equal to the total number of road segments. Each road segment $S_d \in S$ with $d \in [1, \dots, D]$ is described by two endpoints $S_d = \{V_i, V_j\}$ which have positions (x_i, y_i) and (x_j, y_j) respectively. To ensure all road segments are of similar length, a maximum and minimum threshold is chosen, with the maximum road segment length chosen to be less than the sensing diameter of the UAVs. Road segments greater than the specified maximum length are divided into equal sections of length smaller than the maximum. These newly created segments are appended to the set S . Road segments in $S_d \in S$ whose lengths are less than the specified minimum are grouped with adjacent road segments to create a grouped segment G_g , where g is an index of the road segment group, and G_g is made up of subsets of S . Let G be the total set of road segment groups. Every $G_g \in G$ contains one or more road segments and every segment in S belongs to exactly one G_g . The total length of road segments contained in group G_g , $\text{len}(G_g)$, is greater than the minimum road segment length and less than the maximum road segment length.

In a grid-based approach, $\mathcal{M}$ is divided into equally spaced grid cells S_g each identified by a grid number g and center location (x_g, y_g) in the east-north frame.

In both search approaches, the reward function for UAV a is

$$J_{\text{search}}(k) = \sum_g J_g(k), \quad \forall g \in \Gamma_a(k),$$

where $\Gamma_a(k)$ is either the union of grid cells or road segment groups G that are contained within the vehicle's sensing radius at time step k and

$$J_g(k) = J_{g,\text{max}}(k) - (J_{g,\text{max}}(k) - J_g(k-1))e^{-\Delta t/\zeta} \quad (1)$$

is the reward of the g th road-segment group (or grid cell) with growth rate ζ , time step Δt between sensor measurements, and the maximum grid cell or group reward $J_{g,max}(k)$. The reward $J_g(k)$ is time-varying and may differ for each group or cell.

Equation (1), modified from [45], rewards visiting areas that have not been recently viewed. Making $J_{g,max}(k)$ time-varying is critical to modifying UAV behavior based on information learned from the environment. The definition of $J_{g,max}(k)$ is discussed in Section 4.

Grid cells are sensed, and belong to the set $\Gamma_a(k)$, when their centers (x_g, y_g) fall inside the UAV's sensing radius r_s , in other words, the distance $r_{g,a}(k)$ between the UAV and grid cell center is less than r_s , which is calculated as

$$r_{g,a}(k) = \sqrt{(x_g - x_a(k))^2 + (y_g - y_a(k))^2}.$$

Road segment groups require that the endpoints of each segment be contained within the sensing radius. The distances between the UAV and the two endpoints V_i and V_j of segment S_d are defined as

$$\begin{aligned} r_{i,a}(k) &= \sqrt{(x_i - x_a(k))^2 + (y_i - y_a(k))^2} \\ r_{j,a}(k) &= \sqrt{(x_j - x_a(k))^2 + (y_j - y_a(k))^2} \end{aligned}$$

and therefore $r_{d,a}(k) = \max(r_{i,a}(k), r_{j,a}(k))$ and $r_{g,a}(k) = \max\{r_{d,a}(k) : \forall S_d \in G_g\}$. When $r_{g,a}(k)$ is less than r_s , then $g \in \Gamma_a(k)$.

We use a receding horizon controller (RHC) to simultaneously plan the paths for all UAVs. RHC operates by looking at future rewards over an event horizon and selecting the path that maximizes the UAV's objective function. The vehicles then execute one decision command in their chosen path policy, before repeating the process of re-optimizing the entire path for the next horizon. At each time step, the vehicle assesses all possible control commands. For this work, each UAV controls its roll angle ϕ_c and can choose to bank left, go straight, or bank right, with $\phi_c \in [-\phi_{max}, 0, \phi_{max}]$.

With cooperating vehicles, a joint-reward RHC is used to determine paths for all the vehicles that will maximize the aggregate reward. The combined reward for A vehicles of an RHC with event horizon L is computed using [45]

$$J_{search}^A = \sum_k \sum_g^{k+L} J_g(k), \quad \forall g \in \Gamma^A(k), \quad (2)$$

where $\Gamma^A(k) = \left\{ \bigcup_a (r_{g,a}(k) < r_s) \right\}$ is the set of road segments or grid cells that are contained within any of the UAV's sensing radius. The value $J_g(k)$ is given if it lies within the field of view of any vehicle. Therefore, multiple vehicles viewing the same location simultaneously do not yield any additional reward.

The goal of our vehicle path planning objective can thus be summarized as follows:

$$\begin{aligned} &\text{maximize}_{\Phi_c^A} \quad J_{search}^A \\ &\text{subject to} \quad \forall \phi_c \in \Phi_c^A, \phi_c \in [-\phi_{max}, 0, \phi_{max}] \end{aligned} \quad (3)$$

where Φ_c^A is the sequence of commands for all UAVs over the horizon.

3.3. Vehicle Path Planning

Jointly planning the paths of cooperating UAVs is computationally expensive. Using a standard exhaustive search requires evaluating the objective function of every path combination of all the UAVs. This becomes impractical even with a modest number of UAVs employing a short event horizon. We address this challenge with two strategies: (a) a Rollout policy that lessens the number of paths evaluated, and (b) a block coordinate

ascent (BCA) optimization, which plans each vehicle's path individually while assuming the paths of all other vehicles remain fixed.

This paper builds on the Rollout policy introduced in [36], which combines exhaustive search with a greedy heuristic. We augment the Rollout policy by introducing an adaptive threshold that decides when to transition from an exhaustive search to our greedy-heuristic method. The threshold is triggered when there is sufficient spatial diversity between the UAV paths being evaluated.

Figure 3 shows our adaptive Rollout policy, where initially, every possible combination of D discrete command options is evaluated. Once a spatial distance between UAV paths exceeds a predefined threshold, the policy switches to greedily selecting the immediate best reward, significantly reducing the number of UAV path combinations that need to be evaluated. The exhaustive search guarantees an optimal solution for the initial planning steps, and after achieving sufficient spatial diversity, the Rollout policy completes the path. In practice, this method can produce near-optimal results [36]. The initial exhaustive search remains computationally feasible because it is limited to a small portion of the overall planned path before switching to the greedy-heuristic policy. This approach aligns well with our discrete optimization problem that involves both a nonlinear objective function and constraints.

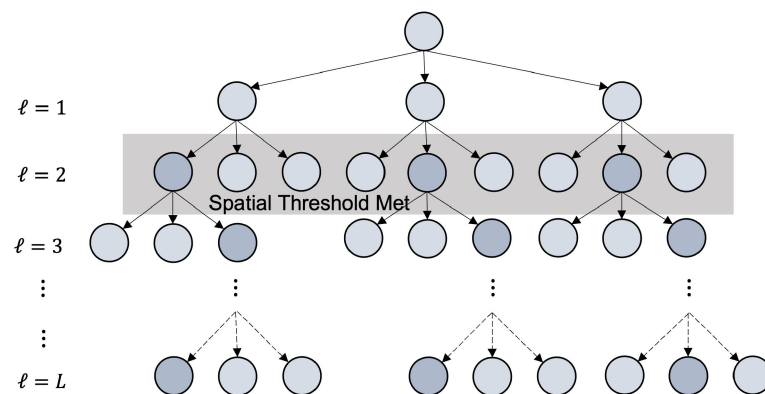


Figure 3. This figure illustrates the Rollout policy for path planning, which initially employs an exhaustive search before transitioning to a greedy heuristic policy once sufficient spatial diversity is achieved among the potential paths. The darker gray circles represent the states selected for expansion by the greedy heuristic policy.

Other methods may offer more computationally efficient path planning and should be explored. In particular, approaches such as Monte Carlo Tree Search [46] would align well with the discrete and nonlinear aspects of our objective function and should be evaluated to determine whether it provides a better path-planning solution. However, since this was not the primary focus of our paper, we have not included that work here. Instead, we suggest that future research explore alternative optimization techniques that may further reduce computational costs.

The second method we use to reduce computational complexity is the BCA algorithm. This is a derivative-free optimization approach that allows us to optimize the path of a single vehicle conditioned on assuming the policies of all other vehicles are fixed [47,48]. Each vehicle iteratively updates its policy, sharing the best policy with its neighbors. This cyclic process repeats until a convergence criterion is met. In this work, convergence occurs when either no vehicle changes its policy after a complete cycle of all UAVs updating their paths, or a predefined number of cycles is reached. In practice, we found that convergence generally occurs after 2–3 cycles.

The full path planning algorithm is provided in Algorithm 1. The potential paths are propagated forward for L lookahead steps, initially exploring all possible paths. If the spatial threshold is met, the planner switches to a greedy policy, selecting the next UAV state with the highest immediate reward. Once all UAV paths are propagated forward

to the L th lookahead step, the path with the highest reward is chosen and shared with all neighboring UAVs. The next UAV then plans its path, assuming all other UAVs will follow the most up-to-date paths shared by their neighbors. These cycles continue until the convergence criterion is satisfied. With the RHC, the UAVs then execute a step of its planned path, prior to using the information learned to re-plan a new path.

Algorithm 1 Vehicle Path Planning.

```

1:  Input: Current UAV states  $s_i \forall i \in [1, \dots, A]$  and current path policies  $\pi$ .
2:  Output: Updated path policies  $\Phi_c^A$  for all vehicles
3:  while converging do
4:    for each UAV  $a \in [1, \dots, A]$  do
5:      Set  $P = \{s_a\}$ 
6:      for each lookahead step  $\ell \in [1, \dots, L]$  do
7:        for each parent node state  $p \in P$  do
8:          Set  $P = P \setminus \{p\}$   $\triangleright$  Remove  $p$  from our set of parent nodes.
9:          for each discrete control command  $\phi_c \in [-\phi_{max}, 0, \phi_{max}]$  do
10:           Propagate forward UAV  $a$ 's state using  $\phi_c$  to get  $s_{a,\ell,c}$ 
11:           Evaluate the reward  $J_{search,d}^A$  for the full path up to this state:
12:             (a) Using Equation (2), and
13:             (b) the cumulative path from state  $s_a$  to  $s_{a,\ell,c}$ , while
14:             (c) keeping the policies of other vehicles are fixed.
15:           Calculate the spatial difference  $\tau_{(i,j)}$  between each child
16:           state  $s_{a,\ell,c} \forall \phi_c \in [-\phi_{max}, 0, \phi_{max}]$ 
17:           if  $\max\{\tau_{(i,j)}\} > \text{spatial threshold}$  then
18:             Pick the child node  $s_{a,\ell,c}$  with the highest path reward:  $\max\{J_{search}^A\}$ .
19:             Add the highest reward state into set  $P = P \cup \{s_{a,\ell,c}\}$ 
20:           else
21:             Add all child states as the parent states for the next lookahead
22:             step:  $P = P \cup \{s_{a,\ell,c}\} \forall \phi_c \in [-\phi_{max}, 0, \phi_{max}]$ 
23:           Communicate UAV  $a$ 's policy  $\Phi_{a,c}^A$  with neighbors

```

4. Learned Search Rewards

This section outlines the key contribution of our paper: a novel approach for dynamically adapting UAV search behaviors based on real-time environmental data. As described in the prior section, the UAVs determine their optimal path using the objective function in Equation (3). However, as they collect sensor data during flight, we use that sensed information to evolve their search behavior through modification of the search rewards. This ongoing data collection is used to continuously adjust the maximum reward value, $J_{g,max}(k)$, which informs and refines the vehicle's future planning.

Our method ensures that UAVs not only prioritize high-density target regions but also maintain essential coverage of lower-density areas. By continuously updating the reward values based on the UAV's learned target densities, this approach optimizes search efficiency while ensuring thorough exploration of the entire operational region.

The steps for determining the reward are shown on the right side of Figure 2. We will discuss how each of these steps is implemented for both the road network and the grid cell approach in the following two subsections.

4.1. Road Network

In this subsection, we explain how our search rewards are calculated when operating with the UAVs following a road network. Algorithm 2 provides an outline of the road segment reward method and will be explained below. This iterative algorithm runs at each time step, incorporating the latest information on target positions. The steps for the road segment reward method are as follows:

Algorithm 2 Method for Multi-Agent Search using Road Network GPR.

```

1: Initialize arrays  $C_a(k)$  and  $T_a(k)$ , with length equal to the number of road segments  $D$ 
2: Compute covariance matrices  $K$ ,  $K_s$ ,  $K_{ss}$  (Equation (12))
3: For each timestep  $k$ 
4:   For each UAV  $a \in [1, \dots, A]$ 
5:     Local Heat Map Update:
6:     For all road segments within sensing radii of UAVs
7:       Set  $T_d(k) = \text{true}$ 
8:     end for
9:     For all detected targets
10:      Update estimated target state  $\hat{x}$  using EKF
11:      Find target association likelihoods to each road segment (Equation (4))
12:      Set  $C_d(k)$  to the normalize likelihood values
13:    end for
14:    Consensus:
15:    Compute road segment group target densities,  $\rho_{g,a}(k)$  (Equation (5))
16:    Compute average target density  $\bar{X}_{d,a}$  and variance  $s_{d,a}^2$  for each segment
17:    GP Regression:
18:    Find estimations  $f_a$  and variances  $\sigma^2$  (Equations (7) and (8))
19:     $Y_a \leftarrow f_a + \sigma_g$ 
20:    Normalization and Scaling:
21:    Normalize  $Y_a$  (Equation (11)) to find  $J_{d,a,max}$ 
22:    Time Update:
23:    Find road segment reward  $J_d$  at time  $k$  (Equation (1))
24:  end for
25:  Choose Action:
26:  Choose the best action for the UAV using RHC
27: end for

```

1. **Local Heat Map Update:** In this step, each UAV uses its observations to locally create a heat map of the target densities. A probabilistic approach is used to associate the noisy target observations with specific road segments. This approach enables us to distribute the target's probability of associating across multiple road segment groups. The heat map is stored as a list of target counts and a number of road segment observations.
2. **Consensus:** Local target counts are shared among peer UAVs using a consensus algorithm. This decentralized approach eliminates the need for a common ground station, allowing the UAVs to share information directly with those within their communication radius. Through the consensus algorithm, each UAV agrees on the target counts and number of observations for the road segments. This enables them to act on the collective knowledge of all cooperating UAVs, providing a more complete view of the environment, including areas they may not have directly observed.
3. **Gaussian Process Regression:** We use GPs to model the mean and variance of target densities across the entire operational area, allowing us to predict densities in areas without direct or recent measurements. Applying GPs to a road network required developing a custom kernel function, where similarity is propagated along connecting roads and through intersections. Updated target densities and variances are recursively calculated through the GP regression anytime new information is received.
4. **Normalization and Scaling:** The GP-derived predicted target densities and variances are used to calculate a maximum reward value for each road segment. To do this, we first increase the GP estimates by their standard deviations. This incentivizes exploration; areas with high uncertainty will yield higher reward values. Next, these adjusted estimates are normalized to a predetermined maximum value, J_{max} , and assigned to each road segment group as their upper reward limit, denoted as $J_{g,max}(k)$.
5. **Time Update:** The upper reward limit, $J_{g,max}(k)$, for a road segment group g , is used to calculate the reward a UAV receives for observing that group. This value is updated

at each time step according to Equation (1). The update follows an exponential curve that starts at the last road segment observation, incentivizing the monitoring of areas that have not been viewed recently.

Each of these steps is described in detail in the following subsections.

4.1.1. Local Heat Map Update

This subsection explains how each vehicle creates a local heat map of the observed target densities. At each time step, each vehicle uses its estimated target tracks $\hat{x}_{p,a}$ to update the local target counts. Only tracks that were detected during the current time step are included. The counts are stored for vehicle a in a one-dimensional array $C_a(k)$ of length D . Each entry in the array $C_{d,a}(k)$, where $d \in [1, \dots, D]$, contains the number of targets assigned to each road segment for time step k . Since measurements are noisy, target estimates may not perfectly align with the road segment on which they are traveling. Therefore, we probabilistically associate each target with roads using its likelihood value.

To find this association, we first project $\hat{x}_{p,a}$ onto each line segment to get the closest point on the road segment to the target. Then using the target's error uncertainty $P_{p,a}(k)$, we compute the Mahalanobis distance $\mu_{p,c}$ between the target p 's estimated position and the closest point on each road segment. Road segments that fall within the 95% Chi-squared bound, $\mu_{p,c} < \sqrt{5.99}$, are selected. For selected segments, the likelihood value is determined as

$$\Lambda = be^{(-\frac{1}{2}\mu_{p,c}^2 - \kappa)}, \quad (4)$$

where $b = \text{card}\{r_{g,a} < r_s\}$ is equal to the number of road segments within the UAV's sensing radius and $\kappa = -0.5 \text{Log}(|P_{p,a}(k)|) - \text{Log}(2\pi)$ is the constant value for the log-likelihood. The likelihood values are normalized and stored in $C_a(k)$ for each road segment at that time step.

Each UAV also keeps a Boolean array T_a where the d^{th} entry is $T_{d,a}$ with $d \in [1, \dots, D]$. The entry's Boolean value indicates if the road segments fell within the vehicle's sensor range during that time step. This vector helps track negative information (i.e., when a road segment contains no targets) and ensures that target densities are computed correctly.

At the start of a simulation, the vectors $C_a(0)$ and $T_a(0)$ are initialized with non-zero target counts and observations. This encourages an initial exploration of all the road segments.

4.1.2. Consensus

This subsection describes the decentralized algorithm used to share target density information among a group of cooperating UAVs. UAVs can only communicate with peer vehicles that fall within their communication range. Although the communication topology is time-varying, we assume that it is strongly connected at each time step. Decentralized sharing of target information is accomplished using a proportional-integral (PI) consensus filter. PI consensus filters have been utilized in past research when communicating spatiotemporal environmental information across strongly connected communication topologies [38,39].

We use the PI consensus filter to converge to the average number of target counts and times that each road segment was sensed using the local C_a and T_a values. Let $\eta_{0,a}$ be UAV a 's input to the estimated value (either $C_{d,a}$ or $T_{d,a}$). The PI consensus filter is [39]

$$\begin{aligned} \dot{\eta}_a &= \gamma(\eta_{0,a} - \eta_a) - K_p \sum_{j \in N_a} (\eta_a - \eta_j) + K_I \sum_{j \in N_a} (\zeta_a - \zeta_j) \\ \dot{\zeta}_a &= -K_I \sum_{j \in N_a} (\eta_a - \eta_j), \end{aligned} \quad (5)$$

where $\gamma > 0$ is a gain indicating its reliance on its own input relative to the input of others, K_p is the proportion gain, K_I is the integral gain, η_a is the consensus variable of vehicle a

(i.e., the average of $C_{d,a}$ or $T_{d,a}$), ζ_a is an integrator variable used internally to the filter, and N_a are the peer agents in the neighborhood of UAV a .

From the resulting consensus outputs, we calculate the target counts for each group $C_{g,a}(k)$ as the sum of counts for all road segments in its group at each timestep k . The density for the group at each timestep is then defined as

$$\rho_{g,a}(k) = \frac{C_{g,a}(k)}{\text{len}(G_g)}.$$

We define $\bar{X}_{g,a}$ to be the average of $\rho_{g,a}$, where we use $T_{d,a}$ from one of the segments in the group (all $T_{d,a}$ in the same group would be equal) to determine the number of measurements of the group. The average target density for each road segment $\bar{X}_{d,a}$ in the group are all set to the group average. The combined vector of average target densities for all road segments is defined as $\bar{X}_a \in \mathbb{R}^{D \times 1}$ where the total number of road segments is D . We let $s_{g,a}^2$ be the variance of the target densities for the road segments in group g .

A target density heat map is created using $\bar{X}_{d,a}$, which provides UAV a 's best information of the area. The heat map continuously updates as targets are observed by the UAVs.

4.1.3. Gaussian Process Regression

This subsection describes how we utilize a GP regression to predict the target densities across the entire operational area. GPs allow us to predict target information in areas without direct observations given a spatial correlation between two locations. We spatially correlate our target densities using a unique kernel function that propagates target densities along the road network and intersections, assuming an equal probability of movement in any direction.

We will outline the GP regression notation that is relevant to our research and show how it is used to create a global target density map. A more comprehensive treatment of GPs may be found in [15].

Using information from the target density heat map, a GP regression is applied to all road segments, utilizing their mean values and variances, $\bar{X}_{d,a}$ and $s_{d,a}^2$. This approach allows us to predict target densities in regions without direct observations. Gaussian processes are defined as collections of random variables at input points X that are characterized by a mean function $m(X)$ and covariance function $k(X, X')$. The distribution over an arbitrary function $f(X)$ is defined as

$$f(X) \sim GP(m(X), k(X, X')).$$

GP regression employs Bayes' rule to compute a posterior distribution over functions using both training and test points. For this work, the training data consists of measurements from the road segments, and the test points include all road segments within the search area. The resulting posterior distribution enables inference of the target density across the entire search region.

We assume that the mean $m(X) \equiv \mathbf{0}$, and that the input X is all road segments in S . The prior covariance K is an $n \times n$ matrix, where n is the number of road segments that have at least one measurement. The matrix K is constructed using a custom kernel which accounts for the minimum along-road distance $\text{dist}(i, j)$ between road segments i and j , and the number of intersections $h(i, j)$ along this shortest path. The custom kernel function is designed to be

$$k(S_i, S_j) = \sigma_f e^{-\lambda_1 \text{dist}(i, j)} e^{-\lambda_2 h(i, j)} \quad (6)$$

where λ_1 , λ_2 , and σ_f are the tuned hyperparameters. The along road distances $\text{dist}(i, j)$ are computed using Dijkstra's algorithm with special care taken to find the shortest distance between either of the two road segment's endpoints. Road segments $i, j \in \xi$, where $\xi \subset S$ contains road segments that were measured at least one time.

This kernel function provides a similarity measure between road segments and operates under the assumption that ground vehicles travel by the shortest path from one location to another and are equally likely to choose any direction at an intersection. Note our prior covariance matrix is only dependent on the structure of the road network. In practice, we only need to compute this kernel function once, offline, when the road network is defined. Then in-flight UAVs take a subset of this matrix (for the road segments that have been observed) to use in the GP regression.

A joint covariance matrix of current data (training points) and desired estimates (test points), K_s , is constructed and used to help predictive variance and mean target density. The matrix K_s is of size $D \times n$, where D is the total number of road segments, and n is the number of road segments with one or more measurements. The matrix is constructed using Equation (6), where $K_s(i, j) = k(S_i, S_j)$, $\forall i \in S$ and $\forall j \in \xi$.

New road segment predictions are calculated using

$$f_a = K_s(K + E)^{-1}\bar{X}_a, \quad (7)$$

where E is a matrix with diagonal elements $E(i, i) = s_{i,a}^2$, $\forall i \in \xi$, [49,50]. The prediction vector f_a is of length $D \times 1$.

The variance of f_a is calculated by

$$\sigma_a^2 = K_{ss} - K_s(K + E)^{-1}K_s^T, \quad (8)$$

where $K_{ss}(i, j) = k(S_i, S_j)$, $\forall i, j \in S$.

Tuning the hyper-parameters of the kernel function leads to optimized performance of the Gaussian process. To illustrate the importance of proper hyper-parameter tuning consider a simple 1-D case that uses the Laplacian kernel function

$$k(i, j) = \sigma_f^2 e^{-\lambda|x_i - x_j|}, \quad (9)$$

where λ is the length scale, σ_f^2 is the signal variance, and x_i, x_j are the 1-D positions. The hyper-parameters λ and σ_f^2 are the length scale and signal variance respectively.

As demonstrated in [15], Figure 4 illustrates the impact of the length factor and signal variance on the GP regression, highlighting how variations in these parameters can lead to good-fit, under-fit, or over-fit models. Data points sampled from a noisy sine wave are shown as red “+” symbols, with the GP prediction represented by the blue line and the 95% confidence region shaded in gray. In Figure 4a, a GP regression with high signal variance σ_f^2 and a length scale set to cause underfitting is depicted. Figure 4b displays a GP regression with lower σ_f^2 and a chosen length scale that achieves a good fit. Figure 4c shows an over-fitted GP regression, where the prediction follows the training data too closely, and the uncertainty region expands rapidly, even at small distances from the training data points.

In Equation (6), the hyper-parameters are λ_1 , λ_2 , and σ_f^2 (two length scales and signal variance). The length scales affect the distance traveled in the input space before the output changes significantly [15].

A common method of tuning the hyperparameters of a kernel function is to minimize the log marginal likelihood of that kernel function using gradient descent methods [51,52]. We likewise tune the hyperparameters one time off-line by minimizing the negative log marginal likelihood of Equation (6), which is

$$\log p(C_a|X) = -\frac{1}{2}C_a^T(K + \sigma_n^2 I)^{-1}C_a - \frac{1}{2}\log|K + \sigma_n^2 I| - \frac{n}{2}\log 2\pi. \quad (10)$$

The log marginal likelihood was computed using Algorithm 2.1 in [15], which uses Cholesky decomposition to find the matrix inverses.

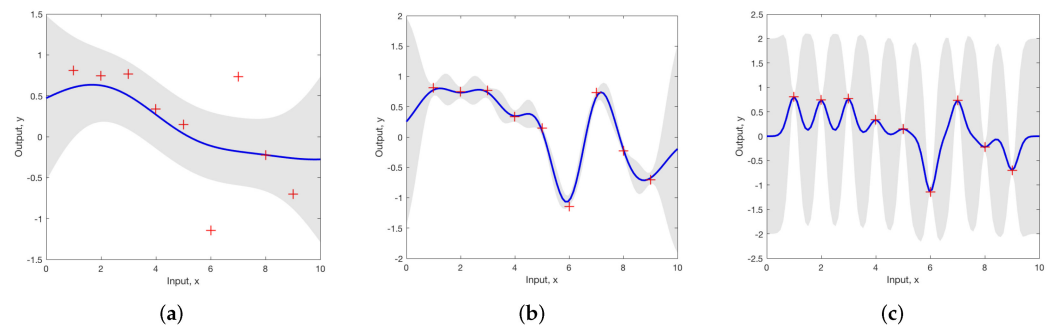


Figure 4. An illustration of how the length scale (λ) and signal variance (σ_f^2) influence the GP regression fit using test data generated from a noisy sine wave. (a) Under-fit GP regression; (b) Good-fit GP regression; (c) Over-fit GP Regression.

Equation (10) was minimized using a gradient descent algorithm. As noted in [15], σ_n^2 is included to model noise variance when tuning. This variable was kept constant throughout the tuning process. In practice, we found that the starting values must be relatively close to the order of magnitude of the ideal value. Optimal tuning also required a training data set that was large enough to seed the Gaussian process but small enough so as not to over-fit the model. We used the observations from a single simulation run with a sub-sample of only 10% of the road segments.

4.1.4. Normalization and Scaling

This subsection outlines the normalization step in our road segment reward method, which ensures a consistent distribution of rewards. The GP regression outputs the predicted target densities for each road segment along with their associated uncertainties. To promote exploration, we augment the predicted target density by adding one standard deviation to the predicted target density according to $Y_a = f_a + \sigma_a$, where $Y_a = [Y_{1,a}, Y_{2,a}, \dots, Y_{D,a}]^\top$. By initializing $C_a(k)$ and $T_a(k)$ with non-zero values, we ensure Y_a does not start at zero, incentivizing the UAVs to explore a road segment multiple times before the variance is decreased.

The array Y_a needs to be normalized to maintain a balanced reward function when integrating with other mission objectives, such as target tracking or collision avoidance. Each road segment maximum reward is computed as

$$J_{d,a,max} = \frac{Y_{d,a} J_{max}}{\max(Y_a)} \quad (11)$$

where J_{max} is the set maximum reward for any road segment.

4.1.5. Time Update

The final step in our reward method is to update the current reward based on the elapsed time. The reward for viewing a road segment group grows exponentially starting from the time it was last observed (as described in Equation (1)). These reward values are used in the RHC controller to evaluate high-value paths and drive the UAV to choose paths that lead to target-rich locations.

4.2. Grid Based

In this section, we describe how search rewards are adapted when operating with the operational space divided into a grid. This approach is useful in scenarios where the road network is either unknown, cannot be automatically generated from open-source databases, or when vehicles are expected to travel off-road or along smaller, unmapped roads. The process for creating the grid cell rewards follows the same steps as those for road networks with only minor modifications. In the next paragraphs, we highlight these differences when compared to the road-network approach.

1. **Local Heat Map Update:** Under the grid-cell implementation, when targets are detected they are associated with the grid cell in which their estimated position is located. Because grid cells have uniform spatial dimensions, the resulting counts are not normalized like they were with the road network implementation.
2. **Consensus:** The consensus algorithm works identically to that presented in Section 4.1.2. However, the values over which consensus is performed now includes a target and observation count for each grid cell (as opposed to road segment).
3. **Gaussian Process Regression:** In the grid-based approach we use a Laplacian kernel function instead of the custom kernel function. The Laplacian kernel function associates the i th and j th grid points based upon the Euclidean distance,

$$k(S_i, S_j) = \sigma_f^2 e^{-\lambda \sqrt{(x_i - x_j)^2 + (y_i - y_j)^2}}, \quad (12)$$

where σ_f^2 is the variance of the signal, λ is the length scale, and $i, j \in \xi$, where $\xi \subset S$ and only contains the grid points which have been measured at least once. The hyper-parameters, σ_f^2 and λ , are optimized using the same method that the road segment hyper-parameters were chosen, as is described in Section 4.1.3.

4. **Normalization and Scaling:** As with the road-network GP, we apply a normalization factor to determine the maximum possible reward value, J_{max} , for each individual grid cell. Each grid cell is then assigned a relative maximum value, $J_{g,max}(k)$, which (as in Section 4.1.4) is proportional to the ratio of its mean target density plus one standard deviation to the overall maximum of the mean target densities plus one standard deviation across all grid cells.
5. **Time Update:** As before, the reward of a grid cell at any time step is calculated using Equation (1).

5. Results

In this section, we evaluate the GP regression methodology for calculating search rewards by testing within a simulated environment. The scenario includes both urban and rural areas and illustrates the algorithm's ability to map and explore mixed environments. The grid-based map and an imported road network method are both tested.

We test the GP regression method against two baseline search reward methods: uniform maximum search rewards (UMSR) and heat map maximum search rewards (HMMSR). The results demonstrate that the GP regression method improves the ability to accurately model the target environment.

The first comparison method, UMSR, applies Equations (1) and (11) with a fixed J_{max} rather than adapting dynamically based on learned information. In this approach, road segment rewards depend solely on the time elapsed since their last update.

The second comparison method, HMMSR, relies on the learned heat map to set each cell's maximum road segment reward, $J_{g,max}$. It calculates the normalized reward using the mean target value for each cell. However, this method lacks the incentive to search regions initially found to have a low mean target value and is unable to predict mean target values in cells that have not yet been searched.

To evaluate the effectiveness of each approach, we create a true heat map of the search area that is updated using the true positions of the targets at every time step. The heat map is computed by associating each target with the correct road segment or grid cell, keeping a running sum of targets, and then dividing by the elapsed time since the simulation started. Because we use true target locations, it provides the exact heat map at each simulation time step. This enables us to compare the heat maps generated from the three different search methods to the true map.

We use two different metrics to evaluate our three search reward algorithms. The first metric computes the fraction of targets that were observed and is calculated by taking the number of targets the UAVs are tracking and dividing it by the total number of targets

that are currently in the search area. A high value for this metric indicates that the search method is effectively directing the UAVs to regions with higher target densities.

To prevent the UAVs from becoming fixated on local maxima and ensure their learned heat map accurately reflects the true target distribution, the search strategy must balance exploiting target-dense regions with exploring the operational area. The second metric is the root mean squared (RMS) error between the true heat map and the learned heat maps. This is computed at each time step and quantifies the difference in perceived target densities. In computing the learned heat maps, we remove the initial values provided to all grid cells or road segments that were used to artificially encourage exploration at the beginning of each simulation. This ensures that we are only evaluating the information that was directly learned by the UAVs. The target averages are computed at each time step and set to $\bar{X}_{d,t}$. A lower RMS value reflects a more accurate understanding of the environment and target densities.

5.1. Chatsworth Grid Cell Search Simulation

Our simulation environment is illustrated in Figure 5. Figure 5a shows the 5000-m square search area in Chatsworth, California, with red lines indicating the targets' paths throughout the full simulation. We generated these target paths using the Simulation of Urban Mobility (SUMO) software package [53] v1.1. SUMO simulates vehicle movement based on actual road networks and standard driving patterns, such as stopping at intersections. The simulation creates a higher concentration of targets on larger roads, such as freeways, which gradually disperse into the rural areas. The map also includes large low-density traffic areas in the upper and lower-left regions of the map. In this search region, a freeway runs horizontally across the middle of the region. It has two exits positioned in the middle and right section that filter into a set of smaller urban roads. The top-left portion of the search region has no vehicular roads. Figure 5b provides the true heat map for this search region given at simulation time $t = 600$ s.

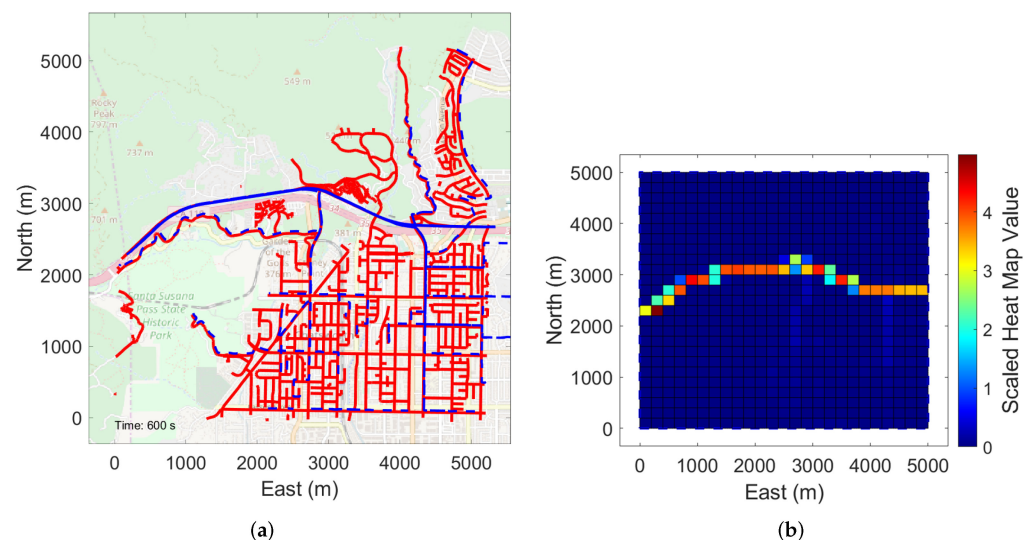


Figure 5. This figure shows the Chatsworth simulation environment and true heat map at time $t = 600$ s. (a) Target paths and road network; (b) True heat map.

The Chatsworth simulation is used to test our GP search method along with the two baseline search comparisons. We tested these algorithms using 100 Monte Carlo (MC) runs, with 120 moving targets, two UAVs, and an event horizon of ten steps for each UAV's path planner. Figure 6 provides an example result of one of the MC simulation runs. Using the method described in Section 4.1.3, σ_f^2 and λ were optimized for the grid cell and road network methods. Table 1 lists the parameters we used in these simulations.

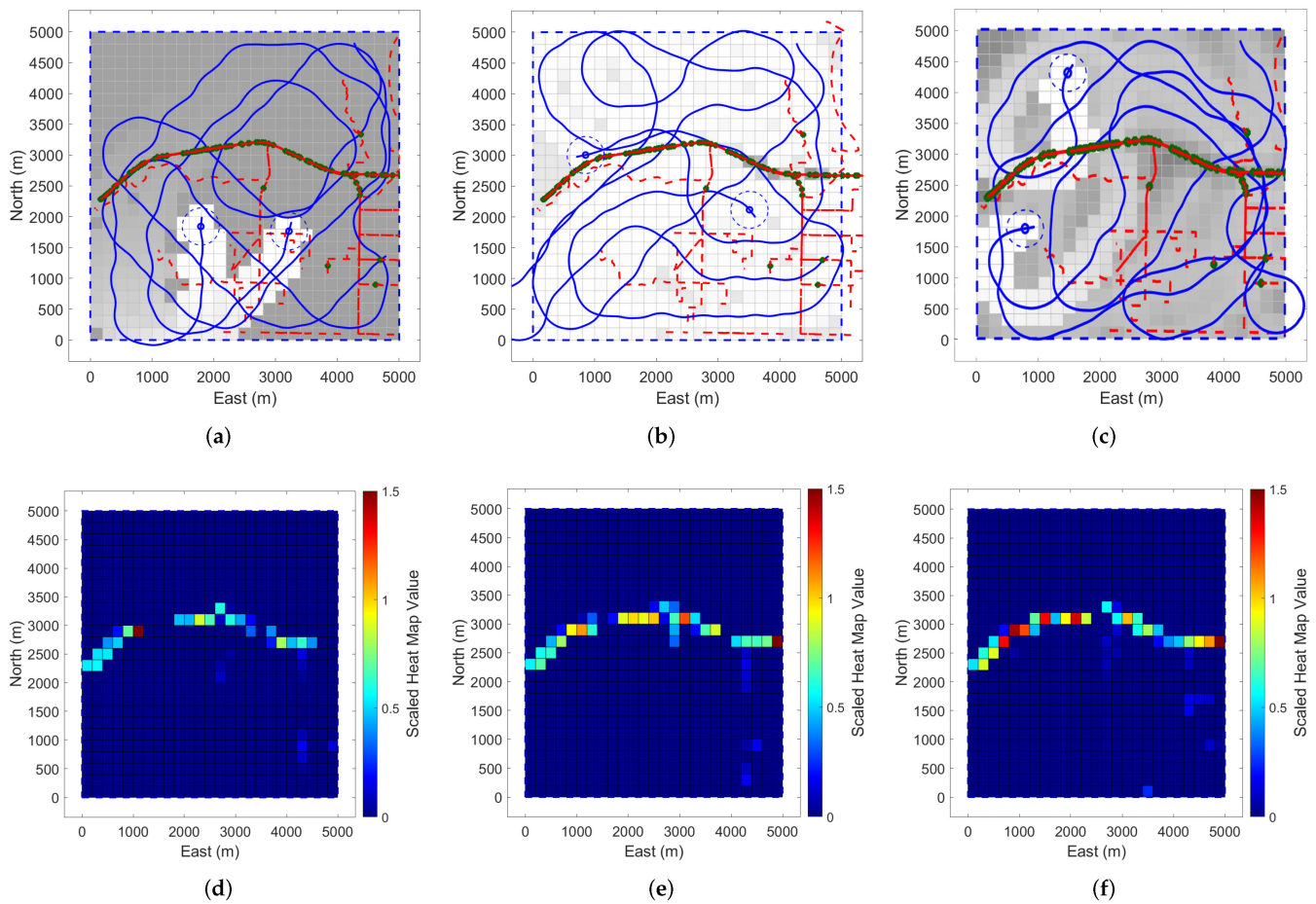


Figure 6. These figures display the heat maps and UAV paths for a single simulation run across all three search methods at time $t = 600$ s, with two UAVs conducting searches in the Chatsworth region. (a) UMSR Method; (b) HMMSR Method; (c) GP Regression Method; (d) Heat map for UMSR; (e) Heat map for HMMSR; (f) Heat map for GP regression.

Table 1. Parameters Used in Simulations.

Simulation Type	Grid Based	Road Network
Sensing Radius (m)	300	300
Airspeed (m/s)	35	50
Target Speed (m/s)	10	10
Maximum Road Segment Length (m)	N/A	75
Minimum Road Segment Length (m)	N/A	2

The simulation for the UMSR method is shown in Figure 6a. As expected, the UAVs cover the search area in an uniform manner regardless of the target densities. The decision to revisit previously search areas is based purely on the elapsed time since they were last observed, as reflected in the grid cell rewards. Cells that were recently searched appear in light gray, while those that have not been observed for a longer period are depicted in dark gray. Figure 6b displays the learned heat map from the UMSR method. When compared to Figure 5b, the outline of the road is visible, although it lacks clarity due to the absence of repeated measurements.

The simulation for the HMMSR method is shown in Figure 6b, where the grid cell rewards show that the learned heat map influences those values. The darker grid cells appear in areas where target observations have been made, indicating that the UAVs have learned these regions have higher target concentrations. The high-density roads, such as

the freeway and bigger streets, consistently get set to the highest reward values due to the increased max rewards, $J_{g,max}$, which encourages more frequent searches of those areas. As a result, the UAVs tend to focus on regions with a higher density of targets. Using the HMMSR method leads to a greater number of target detections and a more accurate heat map, as shown in Figure 6e, compared to the UMSR method. The HMMSR search method offers a better approximation of the true map, yielding a reduced RMS error when compared against the UMSR method.

The top set of panels in Figure 6, Figure 6a–c show each search algorithm’s final target and vehicle trajectories at $t = 600$ s. The target paths are shown with the red lines, with green “x” marking their positions at the final time step ($t = 600$ s). The positions of the two UAVs are given by the blue circles, with the sensing radii depicted by the large dotted blue circles. Each of the UAV’s velocities are given by the blue arrows, while the solid blue lines trace their trajectories throughout the simulation. The current reward value for searching a grid cell is depicted using gray shading, with darker cells indicating higher rewards. Cells that the UAV is actively sensing are shown in white, indicating that the reward has already been collected.

Figure 6c depicts the simulation using our GP regression search method. The UAV trajectories indicate that, in comparison to the other methods, more time is spent observing the road networks in the southwest portion of the search area, reflecting how the UAVs gravitate toward regions with higher target densities. However, the UAVs still engage in exploration, as evidenced by the comparison of their heat maps. The heat map for the single MC run of the GP regression search method is shown in Figure 6f. It is evident that this heat map provides more detail than the heat maps which result from the UMSR and HMMSR methods. The heat map clearly shows the freeway and major city roads.

One advantage of GP regression is its capability of predicting values into unmeasured regions. In our scenario, this means we can predict the target densities of regions that have not been searched. This is shown Figure 6c, where the middle freeway section has a high reward in an area that the UAVs have not searched. This suggests that the GP regression, using data from neighboring areas, has predicted a high average number of targets in those grid cells. As a result of this prediction, the grid cell is given a high maximum reward value, prompting the UAVs to prioritize searching these cells over others. In this simulation, the GP regression’s prediction would likely lead a UAV to encounter a significant number of targets in those areas.

The RMS heat map errors averaged across all the MC runs are shown in Figure 7. The error is displayed for all three of the search methods from time interval $t = [0, 600]$ s. Initially, each method starts with the same error and the error remains close for the first part of the simulation (until about 30 s). However, following this initial period, the GP regression error decreases below that of the other methods and remains significantly lower for the rest of the simulation. The UMSR method performs worse than the HMMSR method because the HMMSR method fully leverages existing knowledge, allowing it to learn the high-density regions more accurately and quickly.

The mean RMS heat map error and standard deviation are shown in Table 2 for each of the three search methods. This table also provides the fraction of targets detected. As anticipated, the GP regression method outperformed the other two methods in terms of heat map RMS error, with an average error of 8.45, compared to 11.41 for UMSR and 9.95 for HMMSR. This represents a 15% and 26% improvement in mapping target densities using our method over the comparative approaches. A two-sample t-test between the error of the GP regression method and UMSR resulted in a p -value of 5.21×10^{-31} , while the comparison between the GP regression method and HMMSR resulted in a p -value of 2.54×10^{-11} . These results indicate that the differences in RMS errors between the methods are statistically significant.

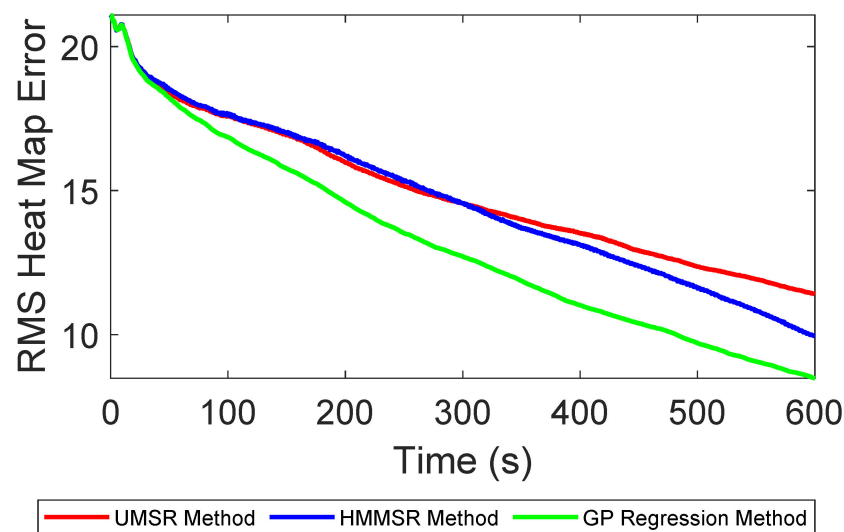


Figure 7. RMS errors for the heat map averaged over 100 MC runs for 600 s using the grid cell method in the Chatsworth region.

Table 2. RMS error, standard deviation, and average fraction of targets seen for the Chatsworth simulations at time $t = 600$ s.

Approach	Final Heat Map Error	Average % Targets Seen
UMSR	11.41 ± 6.70	$3.78 \pm 1.57 \times 10^{-5}$
HMMSR	9.95 ± 6.86	$5.12 \pm 2.2 \times 10^{-4}$
GP Regression	8.45 ± 6.17	$5.60 \pm 4.80 \times 10^{-5}$

Similarly, Table 2 shows the GP regression method also achieves superior performance in terms of the average fraction of targets detected, with an average value of 5.60. This result surpasses that of the HMMSR method, which achieved an average of 5.12, and the UMSR method, which had an average of 3.78. A two sample t-test comparing the GP regression method to UMSR produced a p -value of 7.5×10^{-242} , while the comparison between the GP regression method and HMMSR resulted in a p -value of 1.06×10^{-15} . These low p -values indicate that the differences in the fraction of targets detected by the GP regression method, compared to both UMSR and HMMSR, are highly statistically significant. These results suggest that the GP regression method not only improves target heat map accuracy but also significantly enhances the system's ability to maximize the amount of targets detected within the search area.

5.2. Chatsworth Road Network Search Simulation

The Chatsworth simulation is now used to test the road-network-based search algorithm. The environment is identical to that presented in Section 5.1 with the exception that the search area is segmented into road segments as part of a road network rather than being divided into grid cells. The simulation used OpenStreetMaps to import road segment information.

Figure 8 shows Chatsworth as viewed as a road network. Note that using a road network drastically reduces the search space when compared with the equivalent grid-based approach. As a result, we would expect all three algorithms to have drastically reduced heat map RMS errors. To create the density inequalities in the search space (for which this method was designed), we ensure that the main freeway has a much higher traffic density than the non-freeway roads. The balance of traffic between the main freeway and the non-freeway roads is split so that approximately 80% of the cars travel from one

end of the freeway to the other and 20% of the cars exit the freeway and drive through the residential or non-freeway roads.

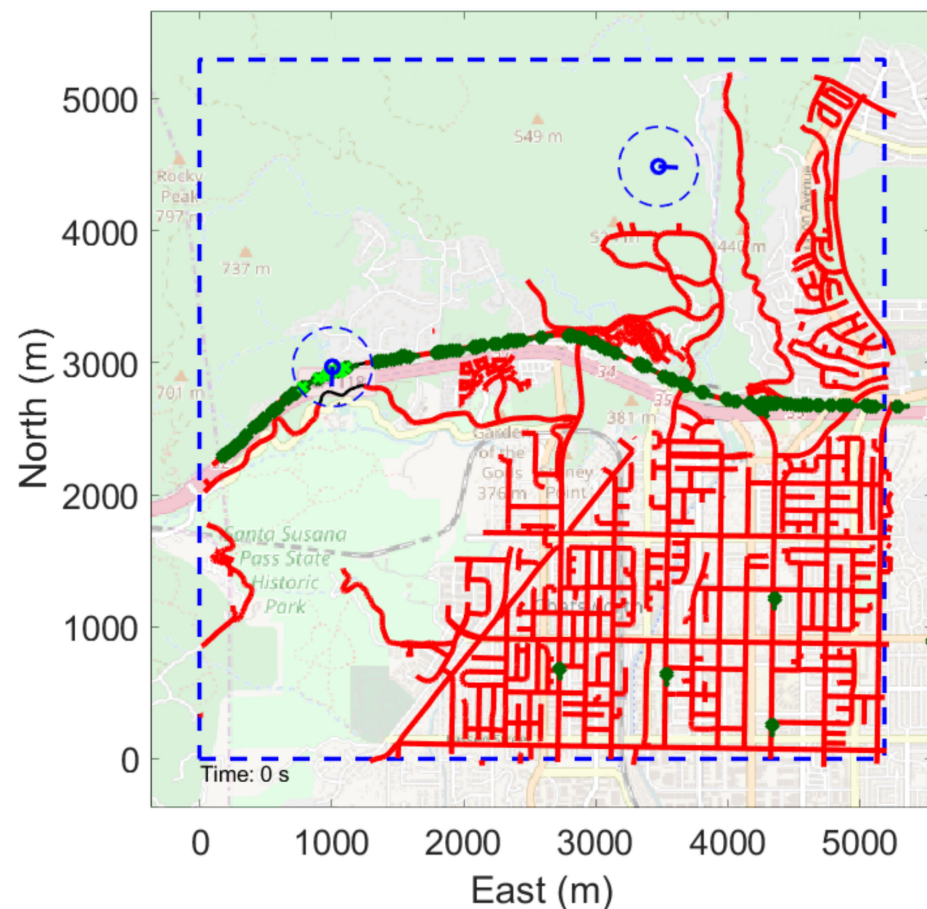


Figure 8. Chatsworth as a road network.

Throughout the simulations, there were approximately 120 active targets. This resulted in a greater dispersion of targets throughout the residential areas and provided more incentive for the UAVs to explore the residential areas. While this also increased the number of targets on the freeway, these road segments had a maximum reward value of J_{max} . Since there was already a high number of targets on these roads the reward values of these roads were not increased as more targets were added to the simulation. The overall outcome was that by increasing the number of targets in the simulation, vehicles that explored the residential areas obtained more information and created a more accurate heat map than vehicles that focused on searching only the highest-density areas along the main freeway.

This second scenario also varies from the first in that a road network, not a grid, was implemented as the basis for the search area. The greatest implication of this change is that the vehicles have a much smaller area to search. While they can still fly off the roads in order to get to other areas of the map, there is no longer any incentive to spend time searching areas with no roads. As the results below will show, the change from grid-based to road networks led to a significant decrease in overall RMS error for all the search methods.

For this simulation, Figure 9 shows the results from time $t = [0, 600]$ s, and Table 3 shows the results at the final simulation time. All three search methods are compared with over 100 Monte Carlo runs performed at ten look-ahead steps with 120 total targets and two UAVs. The results show that GP Regression had the lowest final RMS error of 0.2260 as compared to UMSR and HMMSR. This gives a 2.4% and 1.8% improvement in mapping the target densities when using GP regression over the comparative approaches. When

GP regression was compared to UMSR and HMMSR in a two-sample t-test the p -values were 3.11×10^{-6} and 1.85×10^{-4} respectively, showing a statistically significant lower RMS error.

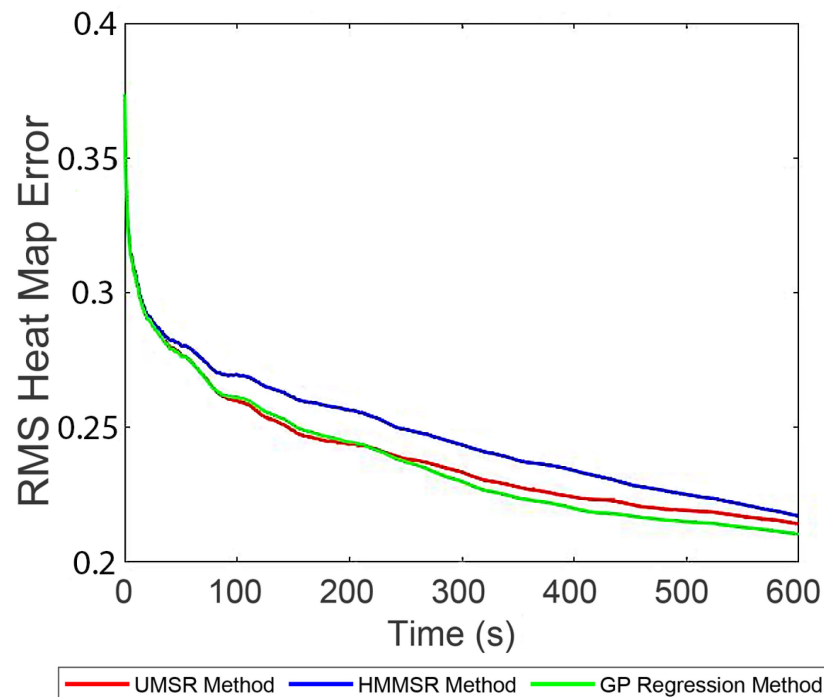


Figure 9. RMS errors for the heat map averaged over 100 MC runs for 600 s using the Chatsworth road network.

Table 3. Road network simulations average and standard deviation of the RMS error and fraction of targets seen.

Approach	Final Heat Map Error	Average % Targets Seen
UMSR	0.2316 ± 0.000129	$5.67 \pm 7.39 \times 10^{-5}$
HMMSR	0.2302 ± 0.000154	$6.23 \pm 9.62 \times 10^{-5}$
GP Regression	$0.2260 \pm 9.72 \times 10^{-5}$	$6.17 \pm 9.35 \times 10^{-5}$

However, when comparing the average number of targets seen, HMMSR had the highest average. A two-sample t-test between HMMSR and GP regression resulted in a p -value of 0.091 which shows that there was not a statistically significant difference in the average targets seen. When comparing the average targets seen by UMSR and GPR the p -value was 9.56×10^{-48} , which is a significant difference. The closeness of these values is unsurprising since the objective function is rewarding searching areas rather than specifically finding targets. The RMS error provides a more suitable measurement relating to how the vehicles are rewarded. Since GP regression had a significantly lower RMS error than the other methods, it can be concluded that GP regression was more accurate in learning and predicting target densities across the whole map.

5.3. Multiple Vehicles

All of the simulation results presented thus far have used only two UAVs. In this subsection, we explore how increasing the number of vehicles changes the results.

In Table 4 we see the resulting RMS error for the UMSR, HMMSR, and GPR methods using 2, 3, and 4 vehicles and a grid-based search. Although this shows that GPR performs better, the difference between the methods decreases as the number of vehicles increases. In terms of a percentage increase, this shows as GP regression performs 16% better than

HMMSR with 2 UAVs, but this changes to only a 3.6% improvement given 4 UAVs. This intuitively makes sense, since additional vehicles mean that the search locations are revisited more frequently and there is less incentive to reward visits to high-density cells. As is also expected, the overall RMS value for each method decreases as the number of vehicles increases. More vehicles enable faster learning of the map.

Table 4. Chatsworth grid simulations final RMS error with multiple vehicles.

Approach	2 UAV	3 UAV	4 UAV
UMSR	10.87 $\pm$ 7.01	8.46 $\pm$ 5.08	6.71 $\pm$ 2.90
HMMSR	9.32 $\pm$ 9.32	6.28 $\pm$ 3.38	4.99 $\pm$ 1.48
GPR	7.78 $\pm$ 5.20	5.63 $\pm$ 2.19	4.81 $\pm$ 1.21

In Table 5 we see the resulting RMS error for the UMSR, HMMSR, and GPR methods using 2, 3, and 4 vehicles and a road-network-based search. This table shows similar trends as were found in the grid-based search method, albeit with lower RMS errors due to the restricted search space. The percentage decrease in RMS error when comparing the GP regression method with HMMSR drops from 2.5% to less than 1% as the number of UAVs increases from two to four. The scenario used for these results was specifically chosen to highlight the utility of the GPR method since this method is most useful when there is both (a) a disparity in the regions with high and low densities of targets and (b) there may be long intervals when the locations are unseen (i.e., the search space is not close to being saturated).

Table 5. Chatsworth road network simulations final RMS error with multiple vehicles.

Approach	2 UAV	3 UAV	4 UAV
UMSR	0.242 $\pm$ 3.410 ^{−4}	0.247 $\pm$ 6.910 ^{−5}	0.254 $\pm$ 3.010 ^{−5}
HMMSR	0.242 $\pm$ 3.0 $\times$ 10 ^{−4}	0.249 $\pm$ 8.1 $\times$ 10 ^{−5}	0.255 $\pm$ 3.5 $\times$ 10 ^{−5}
GPR	0.236 $\pm$ 2.3 $\times$ 10^{−4}	0.244 $\pm$ 3.8 $\times$ 10^{−5}	0.253 $\pm$ 1.1 $\times$ 10^{−5}

6. Conclusions

In this paper, we presented a novel algorithm for searching areas with uneven target densities. Using GP regression, our method continuously integrates new information about target locations to guide cooperative UAV path planning and coordination. Higher rewards are given to regions with denser target populations over time. The results show that GP regression effectively estimates and predicts target densities, even in unobserved parts of the search area. Additionally, simulation results highlight the numerical advantages of GP regression over baseline search methods. Our approach strikes a balance between exploration and exploitation, achieving the lowest error in the learned heat map of target densities while maintaining a high target observation rate.

Future work in this area will address the limitations of the current work and focus on several key directions: (1) Rigorously determining the diversity of target density distributions and the number of UAVs for which this approach is most beneficial. (2) Investigating the use of alternative kernel functions and time-varying Gaussian processes to predict dynamic patterns in the target behavior, overcoming the current assumption of temporally consistent target densities. (3) Developing more computationally efficient path planning policies to improve upon the Rollout policy used in this work.

Supplementary Materials: The following supporting information can be downloaded at: <https://www.mdpi.com/article/10.3390/drones8110606/s1>, Data Explanatory Text File: README.md; Target Data File: target_data.txt; Road Network Matlab Data File: roadDataChat.mat.

Author Contributions: Conceptualization, B.M. and C.K.P.; methodology, B.M.; software, C.A. and B.M.; validation, C.A. and B.M.; formal analysis, B.M., C.K.P. and C.A.; investigation, B.M.; resources,

C.K.P.; data curation, C.A.; writing—original draft preparation, B.M., C.A. and C.K.P.; writing—review and editing, B.M. and C.K.P.; visualization, B.M., C.A. and C.K.P.; supervision, C.K.P.; project administration, C.K.P.; funding acquisition, C.K.P. All authors have read and agreed to the published version of the manuscript.

Funding: This work has been funded by the Center for Unmanned Aircraft Systems (C-UAS), a National Science Foundation Industry/University Cooperative Research Center (I/UCRC) under NSF award No. IIP-1650547 along with significant contributions from C-UAS industry members. This material is based upon work supported by the National Science Foundation Graduate Research Fellowship under Grant No. DGE1745016.

Data Availability Statement: The road networks and target trajectories used in these studies are included in the Supplementary Material; further inquiries can be directed to the corresponding authors.

Conflicts of Interest: The authors declare that there are no conflicts of interest. The funders had no role in the design of the study; in the collection, analyses, or interpretation of data; in the writing of the manuscript; or in the decision to publish the results.

References

1. Wenguang, L.; Zhiming, Z. Intelligent surveillance and reconnaissance mode of police UAV based on grid. In Proceedings of the 7th International Symposium on Mechatronics and Industrial Informatics, ISMII 2021, Zhuhai, China, 22–24 January 2021; pp. 292–295.
2. Wang, S.; Jiang, F.; Zhang, B.; Ma, R.; Hao, Q. Development of UAV-Based Target Tracking and Recognition Systems. *IEEE Trans. Intell. Transp. Syst.* **2020**, *21*, 3409–3422. [\[CrossRef\]](#)
3. Santos, L.; Moon, B.; Scherer, S.; Nguyen, H.V. *UniSaT*: Unified-Objective Belief Model and Planner to Search for and Track Multiple Objects. *arXiv* **2024**, arXiv:2405.15997.
4. Valsan, A.; Parvathy, B.; Vismaya Dev, G.; Unnikrishnan, R.; Reddy, P.K.; Vivek, A. Unmanned Aerial Vehicle for Search and Rescue Mission. In Proceedings of the 2020 4th International Conference on Trends in Electronics and Informatics (ICOEI) (48184), Tirunelveli, India, 15–17 June 2020; pp. 684–687.
5. Sudhakar, S.; Vijayakumar, V.; Sathiya Kumar, C.; Priya, V.; Ravi, L.; Subramaniaswamy, V. Unmanned Aerial Vehicle (UAV) based Forest Fire Detection and monitoring for reducing false alarms in forest-fires. *Comput. Commun.* **2020**, *149*, 1–16. [\[CrossRef\]](#)
6. Outay, F.; Mengash, H.A.; Adnan, M. Applications of unmanned aerial vehicle (UAV) in road safety, traffic and highway infrastructure management: Recent advances and challenges. *Transp. Res. Part A Policy Pract.* **2020**, *141*, 116–129. [\[CrossRef\]](#)
7. Zeng, Y.; Wu, Q.; Zhang, R. Accessing from the Sky: A Tutorial on UAV Communications for 5G and beyond. *Proc. IEEE* **2019**, *107*, 2327–2375. [\[CrossRef\]](#)
8. Moon, B.G.; Peterson, C.K. Learned Search Parameters For Cooperating Vehicles using Gaussian Process Regressions. In Proceedings of the 2018 International Conference on Unmanned Aircraft Systems (ICUAS), Dallas, TX, USA, 12–15 June 2018; pp. 493–502.
9. Caraballo, L.E.; Acevedo, J.J.; Díaz-Báñez, J.M.; Arrue, B.C.; Maza, I.; Ollero, A. The block-sharing strategy for area monitoring missions using a decentralized multi-UAV system. In Proceedings of the 2014 International Conference on Unmanned Aircraft Systems (ICUAS), Orlando, FL, USA, 27–30 May 2014; pp. 602–610. [\[CrossRef\]](#)
10. Nigam, N. The Multiple Unmanned Air Vehicle Persistent Surveillance Problem: A Review. *Machines* **2014**, *2*, 13–72. [\[CrossRef\]](#)
11. Acevedo, J.J.; Maza, I.; Ollero, A.; Arrue, B.C. An efficient distributed area division method for cooperative monitoring applications with multiple uavs. *Sensors* **2020**, *20*, 3448. [\[CrossRef\]](#)
12. Elmaliach, Y.; Shiloni, A.; Kaminka, G. A realistic model of frequency-based multi-robot polyline patrolling. In Proceedings of the 7th International Joint Conference on Autonomous Agents and Multiagent Systems, Estoril, Portugal, 12–16 May 2008; International Foundation for Autonomous Agents and Multiagent Systems: Richland, SC, USA, 2008; Volume 1, pp. 63–70.
13. Kawamura, A.; Soejima, M. Simple strategies versus optimal schedules in multi-agent patrolling. *Theor. Comput. Sci.* **2020**, *839*, 195–206. [\[CrossRef\]](#)
14. Zargar, R.R.; Sohrabi, M.; Afsharchi, M.; Amani, S. Decentralized area patrolling for teams of UAVs. In Proceedings of the 2016 4th International Conference on Control, Instrumentation, and Automation (ICCIA), Qazvin, Iran, 27–28 January 2016; pp. 475–480. [\[CrossRef\]](#)
15. Rasmussen, C.E.; Williams, C.K.I. *Gaussian Processes for Machine Learning (Adaptive Computation and Machine Learning)*; The MIT Press: Cambridge, MA, USA, 2005.
16. Hadji, A.; Szabo, B. Can we trust Bayesian uncertainty quantification from Gaussian process priors with squared exponential covariance kernel? *SIAM-ASA J. Uncertain. Quantif.* **2021**, *9*, 185–230. [\[CrossRef\]](#)
17. Yang, S.; Wei, N.; Jeon, S.; Bencatel, R.; Girard, A. Real-time optimal path planning and wind estimation using Gaussian process regression for precision airdrop. In Proceedings of the 2017 American Control Conference (ACC), Seattle, WA, USA, 24–26 May 2017; pp. 2582–2587. [\[CrossRef\]](#)

18. Lawrance, N.R.; Sukkarieh, S. Autonomous Exploration of a Wind Field with a Gliding Aircraft. *J. Guid. Control. Dyn.* **2011**, *34*, 719–733. [\[CrossRef\]](#)
19. Patrikar, J.; Moon, B.G.; Scherer, S. Wind and the City: Utilizing UAV-Based In-Situ Measurements for Estimating Urban Wind Fields. In Proceedings of the 2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), Las Vegas, NV, USA, 24 October 2020–24 January 2021; pp. 1254–1260. [\[CrossRef\]](#)
20. Wilson, A.G.; Adams, R.P. Gaussian Process Kernels for Pattern Discovery and Extrapolation. In Proceedings of the 30th International Conference on Machine Learning, Atlanta, GA, USA, 17–19 June 2013; pp. 1067–1075.
21. Zhou, M.; Guo, Z.Q.; Li, X. Design of model predictive control for time-varying nonlinear system based on gaussian process regression modeling. In Proceedings of the 2016 IEEE 21st International Conference on Emerging Technologies and Factory Automation (ETFA), Berlin, Germany, 6–9 September 2016; pp. 1–6.
22. Cao, G.; Lai, E.M.K.; Alam, F. Gaussian Process Model Predictive Control of an Unmanned Quadrotor. *J. Intell. Robot. Syst.* **2017**, *88*, 147–162. [\[CrossRef\]](#)
23. Li, F.; Li, H.; Wu, C. Gaussian Process-Based Learning Model Predictive Control With Application to USV. *IEEE Trans. Ind. Electron.* **2024**, *71*, 16388–16397. [\[CrossRef\]](#)
24. Wang, Y.; Duhan, T.; Li, J.; Sartoretti, G. LNS2 + RL: Combining Multi-agent Reinforcement Learning with Large Neighborhood Search in Multi-agent Path Finding. *arXiv* **2024**, arXiv:2405.17794. [\[CrossRef\]](#)
25. Brotee, S.; Kabir, F.; Razzaque, M.A.; Roy, P.; Mamun-Or-Rashid, M.; Hassan, M.R.; Hassan, M.M. Optimizing UAV-UGV coalition operations: A hybrid clustering and multi-agent reinforcement learning approach for path planning in obstructed environment. *Ad Hoc Netw.* **2024**, *160*, 103519. [\[CrossRef\]](#)
26. Zhao, C.; Zhuang, L.; Liu, H.; Huang, Y.; Yang, J. Multi-Agent Path Finding via Reinforcement Learning with Hybrid Reward. In *AAMAS '23: Proceedings of the 2023 International Conference on Autonomous Agents and Multiagent Systems*; International Foundation for Autonomous Agents and Multiagent Systems: Richland, SC, USA, 2023; pp. 2577–2579.
27. Zhao, B.; Huo, M.; Li, Z.; Feng, W.; Yu, Z.; Qi, N.; Wang, S. Graph-based multi-agent reinforcement learning for collaborative search and tracking of multiple UAVs. *Chin. J. Aeronaut.* **2024**. [\[CrossRef\]](#)
28. Su, K.; Qian, F. Multi-UAV Cooperative Searching and Tracking for Moving Targets Based on Multi-Agent Reinforcement Learning. *Appl. Sci.* **2023**, *13*, 11905. [\[CrossRef\]](#)
29. Wang, B.; Gao, X.; Xie, T. An evolutionary multi-agent reinforcement learning algorithm for multi-UAV air combat. *Knowl.-Based Syst.* **2024**, *299*, 112000. [\[CrossRef\]](#)
30. Ning, Z.; Yang, Y.; Wang, X.; Song, Q.; Guo, L.; Jamalipour, A. Multi-Agent Deep Reinforcement Learning Based UAV Trajectory Optimization for Differentiated Services. *IEEE Trans. Mob. Comput.* **2024**, *23*, 5818–5834. [\[CrossRef\]](#)
31. Cui, J.; Liu, Y.; Nallanathan, A. The Application of Multi-Agent Reinforcement Learning in UAV Networks. In Proceedings of the 2019 IEEE International Conference on Communications Workshops (ICC Workshops), Shanghai, China, 20–24 May 2019; pp. 1–6. [\[CrossRef\]](#)
32. Hu, Z.; Zhao, D. Reinforcement learning for multi-agent patrol policy. In Proceedings of the 9th IEEE International Conference on Cognitive Informatics (ICCI), Beijing, China, 7–9 July 2010; pp. 530–535. [\[CrossRef\]](#)
33. Panait, L.; Luke, S. Cooperative multi-agent learning: The state of the art. *Auton. Agents Multi-Agent Syst.* **2005**, *11*, 387–434. [\[CrossRef\]](#)
34. Chiaraviglio, L.; D'Andreagiovanni, F.; Liu, W.; Gutierrez, J.A.; Blefari-Melazzi, N.; Choo, K.K.R.; Alouini, M.S. Multi-Area Throughput and Energy Optimization of UAV-Aided Cellular Networks Powered by Solar Panels and Grid. *IEEE Trans. Mob. Comput.* **2021**, *20*, 2427–2444. [\[CrossRef\]](#)
35. Leon-Blanco, J.M.; Gonzalez-R, P.; Andrade-Pineda, J.L.; Canca, D.; Calle, M. A multi-agent approach to the truck multi-drone routing problem. *Expert Syst. Appl.* **2022**, *195*, 116604. [\[CrossRef\]](#)
36. Tian, X.; Bar-Shalom, Y.; Pattipati, K.R. Multi-step look-ahead policy for autonomous cooperative surveillance by UAVs in hostile environments. In Proceedings of the 2008 47th IEEE Conference on Decision and Control, Cancun, Mexico, 9–11 December 2008; pp. 2438–2443. [\[CrossRef\]](#)
37. Zhou, R.; Feng, Y.; Di, B.; Zhao, J.; Hu, Y. Multi-UAV cooperative target tracking with bounded noise for connectivity preservation. *Front. Inf. Technol. Electron. Eng.* **2020**, *21*, 1494–1503. [\[CrossRef\]](#)
38. Peterson, C.K.; Paley, D.A. Distributed Estimation for Motion Coordination in an Unknown Spatiotemporal Flowfield. *J. Guid. Control. Dyn.* **2013**, *36*, 894–898. [\[CrossRef\]](#)
39. Lynch, K.M.; Schwartz, I.B.; Yang, P.; Freeman, R.A. Decentralized Environmental Modeling by Mobile Sensor Networks. *IEEE Trans. Robot.* **2008**, *24*, 710–724. [\[CrossRef\]](#)
40. Jia, B.; Pham, K.D.; Blasch, E.; Shen, D.; Wang, Z.; Chen, G. Cooperative space object tracking using space-based optical sensors via consensus-based filters. *IEEE Trans. Aerosp. Electron. Syst.* **2016**, *52*, 1908–1936. [\[CrossRef\]](#)
41. Li, T.; Hlawatsch, F.; Djuri, P.M. Cardinality-Consensus-Based PHD Filtering for Distributed Multitarget Tracking. *IEEE Signal Process. Lett.* **2019**, *26*, 49–53. [\[CrossRef\]](#)
42. Ren, W.; Beard, R.W.; Atkins, E.M.A. A Survey of Consensus Problems in Multi-Agent Coordination. In Proceedings of the American Control Conference, 2005, Portland, Oregon, 8–10 June 2005; pp. 1859–1864.

43. Peterson, C.K. Dynamic Grouping of Cooperating Vehicles using a Receding Horizon Controller for Ground Target Search and Track Missions. In Proceedings of the 2017 IEEE Conference on Control Technology and Applications (CCTA), Maui, HI, USA, 27–30 August 2017; pp. 1855–1860.
44. OpenStreetMap Contributors. Planet Dump. 2017. Available online: <https://planet.osm.org> (accessed on 15 October 2024).
45. Newman, A.J.; Martin, S.R.; DeSena, J.T.; Clarke, J.C.; McDerment, J.W.; Preissler, W.O.; Peterson, C.K. Receding Horizon Controller using Particle Swarm Optimization for Closed Loop Ground Target Surveillance and Tracking. *Signal Process. Sens. Fusion Target Recogniton* **2009**, 7336, 73360M-1–73360M-12. [[CrossRef](#)]
46. Akagi, J.; Morris, T.D.; Moon, B.; Chen, X.; Peterson, C.K. Gesture commands for controlling high-level UAV behavior. *SN Appl. Sci.* **2021**, 3, 603. [[CrossRef](#)]
47. Best, G.; Cliff, O.M.; Patten, T.; Mettu, R.R.; Fitch, R. Dec-MCTS: Decentralized planning for multi-robot active perception. *Int. J. Robot. Res.* **2019**, 38, 316–337. [[CrossRef](#)]
48. Peterson, C.K.; Casbeer, D.W.; Manyam, S.G.; Rasmussen, S. Persistent intelligence, surveillance, and reconnaissance using multiple autonomous vehicles with asynchronous route updates. *IEEE Robot. Autom. Lett.* **2020**, 5, 5550–5557. [[CrossRef](#)]
49. Mchutchon, A.; Rasmussen, C.E. Gaussian Process Training with Input Noise. In *Advances in Neural Information Processing Systems*; The MIT Press: Cambridge, MA, USA, 2011; pp. 1341–1349.
50. Quinonero-Candela, J.; Girard, A.; Rasmussen, C.E. *Prediction at an Uncertain Input for Gaussian Processes and Relevance Vector Machines—Application to Multiple-Step Ahead Time-Series Forecasting*; Technical Report; IMM, Danish Technical University: Copenhagen, Denmark, 2002.
51. Schulz, E.; Speekenbrink, M.; Krause, A. A tutorial on Gaussian process regression: Modelling, exploring, and exploiting functions. *J. Math. Psychol.* **2018**, 85, 1–16. [[CrossRef](#)]
52. Ulapane, N.; Thiyagarajan, K.; Kodagoda, S. Hyper-Parameter Initialization for Squared Exponential Kernel-based Gaussian Process Regression. In Proceedings of the 2020 15th IEEE Conference on Industrial Electronics and Applications (ICIEA), Kristiansand, Norway, 9–13 November 2020; pp. 1154–1159. [[CrossRef](#)]
53. Krajzewicz, D.; Erdmann, J.; Behrisch, M.; Bieker, L. Recent Development and Applications of SUMO—Simulation of Urban MObility. *Int. J. Adv. Syst. Meas.* **2012**, 5, 128–138.

Disclaimer/Publisher’s Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.