Cooperative Multi-Agent Path Planning for Heterogeneous UAVs in Contested Environments

Grant Stagg[1] and Cameron K. Peterson[2]

***Abstract*—This paper addresses the challenge of navigating unmanned aerial vehicles (UAV) in contested environments by introducing a cooperative multi-agent framework that increases the likelihood of safe UAV traversal. The approach involves two types of UAVs: low-priority agents that explore and localize threats, and a high-priority agent that navigates safely to its target destination while minimizing the risk of detection by enemy radar systems. The low-priority agents employ a decentralized optimization algorithm to balance exploration, radar localization, and safe path identification for the high-priority agent. For the high-priority agent, two path-planning methods are proposed: one for deterministic scenarios using weighted Voronoi diagrams, and another for uncertain scenarios leveraging generalized Voronoi diagrams and probabilistic constraints. Both methods employ optimization techniques to refine the trajectories while accounting for kinematic constraints and radar detection probabilities. Numerical simulations demonstrate the effectiveness of our framework. This research advances UAV path planning methodologies by combining heterogeneous multi-agent cooperation, probabilistic modeling, and optimization to enhance mission success in adversarial environments.**

I. Introduction

Unmanned aerial vehicles (UAVs) are being increasingly deployed in complex and contested environments for missions that include reconnaissance, surveillance, and combat operations. A key challenge for the UAVs in these scenarios is ensuring their safe navigation while avoiding detection by enemy radar systems. Radar detection is inherently probabilistic, influenced by factors such as radar power, environmental conditions, and UAV positioning. This uncertainty poses a significant challenge for mission-critical UAVs that must traverse hostile regions while minimizing detection risks.

In this paper, we address this challenge by introducing a cooperative framework involving two classes of UAVs: a high-priority agent with a critical mission objective and multiple low-priority agents tasked with scouting the area. The scout agents explore the environment, intercept radar emissions, and estimate radar locations and capabilities, providing critical information for the high-priority UAV. The high-priority UAV then leverages this information to traverse a safe path from its initial location to a target destination.

The use of heterogeneous vehicles increases the likelihood of mission success by designating low-priority vehicles as expendable agents. To maximize their value, we optimize their paths using a three-part objective function that prioritizes uncertainty reduction and exploration while ensuring efficient information gathering along the shortest route. These vehicles integrate probabilistic radar modeling with cooperative multi-agent path planning strategies to gather information that increases the survivability and operational effectiveness of high-priority UAVs operating in adversarial environments.

In addition to our low-priority path planning algorithm, we introduce two high-priority path planning algorithms tailored to different levels of radar parameter certainty. The first algorithm addresses the deterministic case, where all radar parameters are known. First, we use a multiplicatively weighted Voronoi diagram together with an A-star search to produce an initial feasible path. That path is then refined via an interior-point optimizer to yield a minimum-time trajectory that meets both safety and kinematic constraints. The second algorithm is designed for scenarios with uncertain radar parameters. In this case, we use a generalized Voronoi diagram and the A-star algorithm to find an initial feasible trajectory, followed by an optimization process to refine the path.

The key contributions of our work are summarized as follows:

- **Cooperative Path Planning for Low-Priority Agents:** We develop an algorithm that optimally balances regional exploration and the reduction of uncertainty in detected radar locations.
- **Path Planning with Known Radar Parameters:** Utilizing weighted Voronoi diagrams, we design a path planning algorithm that minimizes radar detection risk when radar parameters are fully known.
- **Path Planning with Uncertain Radar Parameters:** For scenarios with uncertain radar parameters, we introduce a path planning algorithm based on generalized Voronoi diagrams to enhance the stealth and survivability of a high-value UAV.

This paper is organized as follows. Section II reviews relevant past research. In Section III, we define the problem and its path-planning applications. Section IV provides essential background information on radar localization, detection uncertainty quantification, Voronoi diagrams, and B-splines. Section VI introduces our multi-agent path planning algorithm for low-priority vehicles, focusing on exploration and uncertainty reduction. In Section VII, we present our high-priority path planning algorithm for radar detection avoidance in both deterministic and uncertain scenarios. Simulation results are presented in Section VIII, and finally, Section IX concludes the paper.

[1] is with the Department of Electrical and Computer Engineering, Brigham Young University, Provo, UT, USA ggs24@byu.edu

[2] is with the Department of Electrical and Computer Engineering, Brigham Young University, Provo, UT, USA cammy.peterson@byu.edu

II. Related Works

Our approach involves two different forms of path planning: one tailored for low-priority attritible agents, and the other for a high-priority agent. The design of our low-priority path planning algorithm is informed by previous research in emitter localization [1]–[7]. Our high priority path planning algorithm draws on previous studies in radar avoidance path planning [8]–[17] and obstacle avoidance techniques using Voronoi diagrams [18]–[24].

The objectives of the low-priority agents are to explore the environment, discover new radar stations, and reduce uncertainty in the estimated locations of the identified radars. Low-priority UAVs discover and localize enemy radar stations. Prior research in this area employed the extended Kalman filter (EKF) to track targets and radio frequency sources based on the angle of arrival and signal strength measurements [1], [2], [5], [6]. Similarly, we utilize the EKF to estimate the locations of enemy radar stations and their effective radiated power. To aid in localization, low-priority agents travel to measurement locations that give the best information about the environment.

Optimization techniques for measurement placement and trajectory planning are integral to these approaches, often leveraging either the Fisher information matrix (FIM) [1], [2], [6] or the EKF covariance matrix [1], [3]–[5], [25] in their objective functions. Since scalar representations are required for optimization, common FIM-based scalarizations include the determinant of FIM (D-optimality), the trace of FIM (A-optimality), and the largest eigenvalue of FIM (E-optimality) [1]. For EKF-based methods, scalarizations like the determinant [2] or trace [6] of the covariance matrix are frequently used in the objective functions. These works aim to optimize path planning by minimizing the scalarization of the EKF covariance matrix or maximizing the scalarization of the FIM. The primary goal is to reduce uncertainty in the estimates as efficiently as possible. However, these works assume there is already an estimate of where the emitter is located, which limits their applicability in scenarios where emitters are initially unknown. Our approach addresses this limitation by incorporating exploration strategies that allow agents to discover and localize previously unidentified radar sources, thus broadening the scope of potential applications.

Building upon these works we utilize reduction of the determinant of the EKF covariance in our objective function. We also include extra terms in the objective that incentivize exploration, creating a trade-off between reducing the uncertainty in the estimate of the already discovered radar, and exploring to discover undiscovered radar. We do this through a Bayesian method outlined in Section VI-A. We also add a third term in the objective function that promotes exploration towards the goal location of the high-priority agent, facilitating the identification of a safe trajectory for the high-priority agent. Additionally, this term helps prioritize efficient navigation by balancing exploration and safety requirements.

For high-priority agents, our research builds on prior work in path planning using Voronoi diagrams and radar detection avoidance strategies. The authors in [18]–[20] use Voronoi diagrams to generate initial trajectories, then use smoothing or optimization algorithms to find feasible minimum-time trajectories. Similarly, we employ Voronoi diagrams to generate an initial feasible trajectory and then refine the path using a smoothing and optimization algorithm.

Past research [21]–[24] has used standard Voronoi diagrams to plan the path around the threat points, such as radar, missile, or terrain. In these studies, threat levels were modeled by assigning weights to edges based on the type and parameters of each threat. The minimum threat path was then computed through the Voronoi diagram. However, these approaches do not account for varying threat levels, e.g., the ideal path should pass closer to a lesser threat. We address this limitation by employing weighted Voronoi diagrams, where the weighted Voronoi ridge naturally shifts closer to the radar with a smaller range, reflecting their reduced threat levels.

The approach in [20] utilizes weighted Voronoi diagrams to generate trajectories around obstacles. Extending this, we apply both weighted and generalized Voronoi diagrams to design initial trajectories that avoid radar. Instead of navigating around physical obstacles, we leverage weighted Voronoi diagrams to identify the ridge of minimum probability of detection between two radar stations with different capabilities. To our knowledge, this represents the first application of weighted Voronoi diagrams in this context.

Prior research in radar detection avoidance path planning has predominantly focused on planning routes through enemy radar under the assumption that the radar parameters, such as location and power, are fully known [8]–[16]. The authors in these works use similar formulations and minimize radar detection or detection probability using the known information. The methods vary significantly in their approaches. For instance, the authors in [8] train a deep reinforcement learning model to generate paths to minimize or avoid radar tracking. In [9], a genetic optimization algorithm is employed to plan safe paths, while leveraging terrain masking to reduce detection risks. The authors in [10] and [12] developed improved A-star algorithms to find paths through enemy radar, while simulated annealing optimization is applied in [11]. Other techniques include ant colony optimization [13] and calculus of variations for optimal path planning [16]. Despite their methodological diversity, all these approaches share a critical limitation: they assume perfect knowledge of radar parameters. This assumption neglects the inherent uncertainty in contested environments, where radar locations and capabilities are often unknown or probabilistic.

Recently, the authors in [17], [26], [27] introduced a linearization technique to account for uncertainty in radar parameters that is similar to the one presented in this paper. In [26], they provide a sensitivity analysis on the radar probability of detection equations with respect to the agents. This approach enables an approximation of the uncertainty in the probability of detection levels, effectively accounting for the variability in the agent's state. In [27] the authors extend their sensitivity analysis of the radar probability of detection, but instead with respect to the radar parameters. This approach approximates the uncertainty in the probability of detection arising from the inherent variability in the radar parameters.

The path planning approach in [17] uses a visibility graph

around radar-based avoidance polygons, with smoothing applied to generate a flyable trajectory. Because the visibility graph places the initial path close to the safety boundaries, smoothing often causes constraint violations, motivating their iterative approach of expanding polygons and replanning. In contrast, we opt to utilize Voronoi diagrams to discover an initial feasible trajectory (satisfying kinematic feasibility and maximum PD constraints), followed by refinement using an interior point optimization algorithm. Both the visibility graph and Voronoi-based paths require smoothing to ensure kinematic feasibility. However, since the Voronoi-based initial path is biased away from high-risk areas, it reduces the likelihood that smoothing will violate constraints. This allows the path to be directly refined with a continuous optimization method, without requiring iterative replanning.

III. Problem Statement

We consider a scenario where all the agents (UAVs) operate in a 2D region $D \subset \mathbb{R}^2$. The low-priority agents are located at $\boldsymbol{x}_{l,i} = [x_{l,i},\ y_{l,i}]^\top \forall i \in \{1, \ldots, N_l\}$, where N_l is the number of low-priority agents, $x_{l,i}$ is the distance East of the origin, and $y_{l,i}$ is the distance North of the origin. Additionally, we assume a single high-priority agent located at $\boldsymbol{x}_h = [x_h,\ y_h]^\top$. The high-priority agent's objective is to plan a path starting at its initial location $\boldsymbol{x}_{h_0}$ and ending at its goal location $\boldsymbol{x}_{h_f}$, while avoiding detection by enemy radar. Meanwhile, the low-priority agent's aim is to explore the region, detect and locate enemy radar, and identify a safe path for the high-priority agent.

There are N_r enemy radar stations in D at locations, $\boldsymbol{x}_{r,j} = [x_{r,j}, y_{r,j}]^\top \ \forall j \in \{1, \ldots, N_r\}$. These locations are unknown to the agents. The low-priority agents are equipped with sensors capable of intercepting enemy radar emissions. Each agent is assumed to measure the angle of arrival $\phi_{i,j}$ of the enemy radar signals, along with the received power. The received power at the i^{th} agent from the j^{th} radar is given by

$$S_{E,i,j} = \frac{P_{T,j} G_{T,j} G_{I,i} \lambda^2}{(4\pi)^2 R_{i,j}^2 L_j}, \tag{1}$$

where $P_{T,j}$ is the power transmitted from the j^{th} enemy radar, $G_{T,j}$ is the transmit gain of the j^{th} enemy radar, $G_{I,i}$ is the gain of the i^{th} agent's intercept antenna, λ is the radar wavelength (we assume the wavelength for all radar is the same), $R_{i,j}$ is the distance between the i^{th} agent and the j^{th} radar, and L_j is the path loss.

We wish to estimate the location of the radar $\boldsymbol{x}_{r,j}$ and the effective radiated power (ERP) $P_{E,j}$ for the j^{th} radar, where the effective radiated power is

$$P_{E,j} = \frac{P_{T,j} G_{T,j}}{L_j}. \tag{2}$$

We combine the location and ERP for each radar into an augmented state that will be estimated. The combined state for the j^{th} radar station is denoted by $\bar{\boldsymbol{x}}_{r,j} = [x_{r,j},\ y_{r,j},\ P_{E,j}]^\top$, where $\left[x_{r,j},\ y_{r,j}\right]$ is the radar's location. Given this augmented state, the radar's measurement model is

$$h(\boldsymbol{x}_{l,i}, \bar{\boldsymbol{x}}_{r,j}) = \begin{bmatrix} S_{E,i,j} \\ \phi_{i,j} \end{bmatrix} = \begin{bmatrix} \frac{P_{E,j} G_{I,i} \lambda^2}{(4\pi)^2 ||\boldsymbol{x}_{a,i} - \boldsymbol{x}_{r,j}||_2^2} \\ \arctan\left(\frac{y_{r,j} - y_{l,i}}{x_{r,j} - x_{l,i}}\right) \end{bmatrix}. \tag{3}$$

We assume that the measurements are corrupted with zero-mean Gaussian noise $\boldsymbol{\delta}$. The k^{th} measurement of the j^{th} radar from the i^{th} agent is given by

$$\boldsymbol{z}_{i,j}^k = h(\boldsymbol{x}_{l,i}^k, \bar{\boldsymbol{x}}_{r,j}) + \boldsymbol{\delta},\ \boldsymbol{\delta} \sim \mathcal{N}(\boldsymbol{0}, \Sigma_z),\ \Sigma_z = \begin{bmatrix} \sigma_{S_E}^2 & 0 \\ 0 & \sigma_\phi^2 \end{bmatrix}, \tag{4}$$

where σ_{S_E} is the noise variance for the power measurement and σ_ϕ^2 is the noise variance for the angle of arrival measurement.

As agents traverse the environment, they gather measurements and store them in a set $\overline{Z}_i = \{\boldsymbol{z}_{i,j}^k\}_{\forall k \in \{1,\ldots,N_{z,i}\}}$, where $N_{z,i}$ is the number of measurements agent i has taken. The agents also store the locations where the measurements were taken, $\overline{X}_{z,i} = \{\boldsymbol{x}_{l,i}^k\}_{\forall k \in \{1,\ldots,N_{z,i}\}}$.

We assume known data association, meaning the agents can reliably identify the radar station from which each measurement originates. This assumption allows us to focus on the core contribution of our work in path planning, rather than delving into the complexities of measurement data association, which would require more advanced estimation algorithms and are beyond the scope of this paper. In real-world scenarios, agents can differentiate radar sources by leveraging signal characteristics such as frequency, pulse width, or pulse patterns, a topic explored in prior research [28].

IV. Background

In this section, we describe several background topics that contribute to our algorithms. We first provide an overview of Voronoi diagrams, weighted Voronoi diagrams, and generalized Voronoi diagrams. Next, we discuss B-splines, which we use to parameterize paths.

A. Voronoi Diagrams

An ordinary Voronoi diagram is defined using a set of generator points $\overline{X}_g = \{\boldsymbol{x}_{g,1}, \ldots, \boldsymbol{x}_{g,N_g}\}$, where N_g is the number of generator points and $\boldsymbol{x}_{g,i} \in \mathbb{R}^n$, with n being the dimensions of the space (in our case, we use a planar space with $n = 2$). The Voronoi cells are then defined as the region where the Euclidean distance between all points in the region and the generator point is less than the distance to any other generator point [29]:

$$V(\boldsymbol{x}_{g,i}) = \{\boldsymbol{x} \mid \|\boldsymbol{x} - \boldsymbol{x}_{g,i}\|_2 \leq \|\boldsymbol{x} - \boldsymbol{x}_{g,j}\|_2, \\ \forall j \neq i,\ j \in \{1, \ldots, N_g\}\}. \tag{5}$$

The Voronoi diagram generated by the points $\overline{X}_g$ is the set of all the Voronoi cells

$$\mathcal{V}(V(\boldsymbol{x}_{g,1}), \ldots, V(\boldsymbol{x}_{g,N})). \tag{6}$$

Voronoi edges are given as the intersection of two Voronoi regions if the intersection exists (e.g. $V(\boldsymbol{x}_{g,i}) \bigcap V(\boldsymbol{x}_{g,j}) \neq \emptyset$):

$$e(\boldsymbol{x}_{g,i}, \boldsymbol{x}_{g,j}) = \left(V(\boldsymbol{x}_{g,i}) \bigcap V(\boldsymbol{x}_{g,j})\right). \tag{7}$$

These edges can either be line segments, half lines (where one direction of the line starts at a point and extends to infinity), or infinite lines. The end points of the Voronoi edges are called Voronoi vertices. These can also be defined as the intersection of three Voronoi regions [29]. We call the set of Voronoi edges $\mathcal{E}(\overline{X}_g)$ and the set of Voronoi vertices $\mathcal{N}(\overline{X}_g)$.

Voronoi diagrams can be generated using different distance metrics. A multiplicatively weighted Voronoi diagram is generated using a weighted distance metric and is defined by the set of generator points $\overline{X}_g$ and a set of corresponding weights $\overline{W} = \{w_1, \ldots, w_N\}$ where w_i is the weight that corresponds to the i^{th} generator point. The multiplicatively weighted Voronoi regions are defined using the weighted distance

$$V(\boldsymbol{x}_{g,i}, w_i) = \left\{ \boldsymbol{x} \middle| \frac{\|\boldsymbol{x} - \boldsymbol{x}_{g,i}\|_2}{w_i} \leq \frac{\|\boldsymbol{x} - \boldsymbol{x}_{g,j}\|_2}{w_j}, \forall j \neq i, j \in \{1, \ldots, N_g\} \right\}. \tag{8}$$

Instead of line segments, Voronoi edges of a multiplicatively weighted Voronoi diagram are arcs of circles. If an edge exists, it can be found using the intersection of two weighted Voronoi regions

$$e(\boldsymbol{x}_{g,i}, \boldsymbol{x}_{g,j}) = \left(V(\boldsymbol{x}_{g,i}, w_i) \bigcap V(\boldsymbol{x}_{g,j}, w_j) \right). \tag{9}$$

The Voronoi vertices of the multiplicatively weighted Voronoi diagram are the intersection points of the Voronoi edges. We define the set of Voronoi edges as $\mathcal{E}(\overline{X}_g, \overline{W})$ and the set of Voronoi vertices as $\mathcal{N}(\overline{X}_g, \overline{W})$. The edges and vertices of a multiplicatively weighted Voronoi diagram can be found efficiently using the algorithm outlined in [30].

B. B-splines

B-splines are piecewise polynomial functions defined by control points $\overline{C} = \{\boldsymbol{c}_1, \ldots, \boldsymbol{c}_{N_c}\}, \boldsymbol{c}_i \in \mathbb{R}^2$ and knot points $\boldsymbol{t}_k = \{t_0 - p\Delta_t, \ldots, t_0 - \Delta_t, t_0, t_0 + \Delta_k, t_0 + 2\Delta_k, \ldots t_f, t_f + \Delta_k, \ldots t_f + p\Delta_k\}$ where $\Delta_k = (t_f - t_0)/(N_k - 2p)$ is the knot point spacing, $N_k = N_c + p + 1$ is the number of knot points and p is the degree of the B-spline for an unclamped uniform B-spline. The B-spline is defined on the interval $[t_0, t_f]$ as

$$\boldsymbol{p}(t) = \sum_{i=1}^{N_c} B_{i,p}(t)\boldsymbol{c}_i, \tag{10}$$

where the basis functions $B_{i,p}$ are defined using the Cox-de Boor recursive formula shown in [31]. B-splines are commonly used in path planning applications because of their local support property (sparse Jacobians) and the convex hull property (the trajectory must be within the convex hull of the control points) [19].

We assume that the agents follow unicycle kinematics

$$\begin{bmatrix} \dot{x}(t) \\ \dot{y}(t) \\ \dot{\theta}(t) \end{bmatrix} = \begin{bmatrix} v(t)\cos\theta(t) \\ v(t)\sin\theta(t) \\ u(t) \end{bmatrix}, \tag{11}$$

where $v(t)$ is the speed of the agent at time t and $u(t)$ is the turn rate. Using the property of differential flatness, we can define kinematic feasibility constraints as done in [32]. The velocity of the trajectory can be found as

$$v(t) = ||\dot{\boldsymbol{p}}(t)||_2, \tag{12}$$

and the turn rate $u(t)$ is

$$u(t) = \frac{\dot{\boldsymbol{p}}(t) \times \ddot{\boldsymbol{p}}(t)}{||\dot{\boldsymbol{p}}(t)||_2{}^2}. \tag{13}$$

We can compute the curvature of the trajectory as

$$\kappa(t) = \frac{u(t)}{v(t)}. \tag{14}$$

V. Radar Localization and Probability of Detection

In this section, we outline our method for localizing radar and estimating the ERP using angle-of-arrival and signal strength measurements. Our approach to radar localization utilizes a combination of nonlinear least squares and an EKF to track enemy radar. We also present a method similar to [26], [27] that accounts for uncertainty in the radar parameters to find the radar PD. This approach linearizes the radar PD equations and propagates the uncertainty through the linearized model.

A. Radar Localization

Our approach for localizing radar stations uses a non-linear least squares method to initialize the radar station models, leveraging available measurements to estimate their initial parameters. Once a model is initialized, the state estimates are refined through EKF correction updates as new measurements are received. The goal is to estimate each radar's combined state $\bar{\boldsymbol{x}}_{r,j}$; its location and ERP, and the associated uncertainty of that estimate, represented as a covariance matrix. The estimation algorithm outputs a list of mean value and covariance pairs, $\overline{R} = \{(\mu_{\bar{\boldsymbol{x}}_{r,j}}, \Sigma_{\bar{\boldsymbol{x}}_{r,j}})\}\ \forall j \in \{1, \ldots, N_{\hat{r}}\}$,where $N_{\hat{r}}$ is the total number of estimated radar stations. Algorithm 1 provides a detailed outline of the radar estimation process.

The input to the algorithm is a stream of measurements $\boldsymbol{z}_{i,j}^k$ and measurement locations $\boldsymbol{x}_{i,j}^k$ from all agents. The Algorithm outputs a mean and covariance estimate for each radar station that has been discovered $\overline{R}$. On line 3, several lists are initialized; $\overline{R}$ will store the mean and covariance pairs for each radar station, then for each radar station a list to store the measurements $\overline{Z}_j$ and measurement locations $\overline{X}_j$ are created. When a new measurement $\boldsymbol{z}_{i,j}^k$ and measurement location $\boldsymbol{x}_{i,j}^k$ is received, it is added to the list of measurements $\overline{Z}_j$ and measurement locations $\overline{X}_j$ for each radar on line 5.

If a model (mean and covariance estimate) exists for the j^{th} radar, the algorithm uses the EKF measurement correction step to incorporate the measurement into the existing model (lines 6-9). This is done by computing the Kalman gain using the current estimate covariance for the j^{th} radar $\Sigma_{\bar{\boldsymbol{x}}_{r,j}}$, the measurement covariance Σ_z, and the Jacobian of the measurement model

$$J_h(\boldsymbol{x}_i^k, \mu_{\bar{\boldsymbol{x}}_{r,j}}) = \left. \frac{\partial h}{\partial \bar{\boldsymbol{x}}_{r,j}} \right|_{(\boldsymbol{x}_{l,i}^k, \bar{\boldsymbol{x}}_{r,j}) = (\boldsymbol{x}_i^k, \mu_{\bar{\boldsymbol{x}}_{r,j}})} \tag{15}$$

evaluated at the measurement location $\boldsymbol{x}_i^k$ and the current mean value $\mu_{\bar{\boldsymbol{x}}_{r,j}}$. After the Kalman gain is computed, the mean and covariance values are updated.

If a model does not exist for a specific radar, the algorithm will initialize a new one if there are at least $N_{z,\text{min}}$ (lines 10-13). A new track's mean value $\mu_{\bar{\boldsymbol{x}}_{r,j}}$ is calculated using a non-linear least squares optimization algorithm where the objective function is to minimize the sum of the squared Mahalonobis distances between the measurements and the measurement model. The covariance is approximated by finding the inverse of the Fisher information matrix. To do this, we find the Jacobian of each measurement model with respect to its state $\bar{\boldsymbol{x}}_{r,j}$ and stack them as

$$\boldsymbol{J}_h(\overline{X}_j) = \begin{bmatrix} J_h(\boldsymbol{x}_1^k, \mu_{\bar{\boldsymbol{x}}_{r,j}}) \\ \vdots \\ J_h(\boldsymbol{x}_i^k, \mu_{\bar{\boldsymbol{x}}_{r,j}}) \\ \vdots \\ J_h(\boldsymbol{x}_{N_{r,min}}^k, \mu_{\bar{\boldsymbol{x}}_{r,j}}) \end{bmatrix}. \tag{16}$$

Similarly, the measurement covariance is stacked in a block diagonally form as

$$\boldsymbol{\Sigma}_z = \begin{bmatrix} \Sigma_z & \mathbf{0} & \mathbf{0} & \cdots & 0 \\ \mathbf{0} & \Sigma_z & \mathbf{0} & \cdots & 0 \\ \mathbf{0} & \mathbf{0} & \Sigma_z & \cdots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \cdots & \Sigma_z \end{bmatrix}. \tag{17}$$

Equations (16) and (17) are then used to approximate the covariance of the estimate of the radar's location and ERP (line 12). The new model $(\mu_{\bar{\boldsymbol{x}}_{r,j}}, \Sigma_{\bar{\boldsymbol{x}}_{r,j}})$ is added to the list of models $\overline{R}$ in line 13.

Algorithm 1 EKF for radar model estimation

1: **Input:** stream of measurement $\boldsymbol{z}_{i,j}^k$ and measurement locations $\boldsymbol{x}_i^k$ from all agents
2: **Output:** $\overline{R}$
3: **Initialize:** $\overline{R} \leftarrow \emptyset, \overline{Z}_j \leftarrow \emptyset, \overline{X}_j \leftarrow \emptyset, N_{z,j} = 0 \ \forall j \in \{1, \ldots, N_r\}$
4: **for** $\boldsymbol{z}_{i,j}^k, \boldsymbol{x}_i^k$ in stream **do**
5: $\overline{X}_j = \overline{X}_j \bigcup \{\boldsymbol{x}_i^k\}, \overline{Z}_j = \overline{Z}_j \bigcup \{\boldsymbol{z}_{i,j}^k\}$
6: **if** Model j is initialized **then**
7: $K = \Sigma_{\bar{\boldsymbol{x}}_{r,j}} J_h(\boldsymbol{x}_i^k, \mu_{\bar{\boldsymbol{x}}_{r,j}})^\top (J_h(\boldsymbol{x}_i^k, \mu_{\bar{\boldsymbol{x}}_{r,j}}) \Sigma_{\bar{\boldsymbol{x}}_{r,j}} J_h(\boldsymbol{x}_i^k, \mu_{\bar{\boldsymbol{x}}_{r,j}})^\top + \Sigma_z)^{-1}$
8: $\mu_{\bar{\boldsymbol{x}}_{r,j}} = \mu_{\bar{\boldsymbol{x}}_{r,j}} + K(\boldsymbol{z}_i^k - h(\boldsymbol{x}_l^k, \mu_{\bar{\boldsymbol{x}}_{r,j}}))$
9: $\Sigma_{\bar{\boldsymbol{x}}_{r,j}} = (I - KJ_h)\Sigma_{\bar{\boldsymbol{x}}_{r,j}}$
10: **else if** $|\overline{Z}_j| = N_{r,min}$ **then**
11: $\mu_{\bar{\boldsymbol{x}}_{r,j}} = \underset{\mu_{\bar{\boldsymbol{x}}_{r,j}}}{\text{argmin}} \sum_{\boldsymbol{x}_l^k \in \overline{X}_j, \boldsymbol{z}^k \in \overline{Z}_j} ||h(\boldsymbol{x}_l^k, \mu_{\bar{\boldsymbol{x}}_{r,j}}) - \boldsymbol{z}^k||^2_{\Sigma_z}$
12: $\Sigma_{\bar{\boldsymbol{x}}_{r,j}} = (\boldsymbol{J}_h \boldsymbol{\Sigma}_z^{-1} \boldsymbol{J}_h^\top)^{-1}$
13: $\overline{R} \leftarrow \overline{R} \bigcup (\mu_{\bar{\boldsymbol{x}}_{r,j}}, \Sigma_{\bar{\boldsymbol{x}}_{r,j}})$
14: **else**
15: Pass
16: **end if**
17: **end for**

B. Radar Probability of Detection

We now present the method used to calculate PD and its associated uncertainty, employing a linearization technique similar to the approach described in [27].

The signal-to-noise ratio (SNR) from i^{th} agent to the j^{th} radar is

$$\mathcal{S}_{i,j}(\Theta_{k,i}, \Theta_{u,j}, \Theta_{e,j}) = \frac{P_{E,j} G_{R,j} \lambda^2 \sigma_i \tau_{p,j}}{(4\pi)^3 R_{i,j}^4 \kappa T_{s,j} L_j}, \tag{18}$$

where $G_{R,j}$ is the gain of the receive antennae of the radar, σ_i is the radar cross section (RCS) of the agent, $\tau_{p,j}$ is the radar pulse length, κ is the Boltzman constant, and $T_{s,j}$ is the radar system noise. Uing the SNR the PD of the i^{th} agent from the j^{th} radar is

$$P_{D,i,j}(\Theta_{k,i}, \Theta_{u,j}, \Theta_{e,j}) = \exp\left\{ \frac{\ln P_{fa,j}}{\overline{S}_{i,j}(\Theta_{k,i}, \Theta_{u,j}, \Theta_{e,j}) + 1} \right\}, \tag{19}$$

where $P_{fa,j}$ is the probability of a false alarm (determined by radar operator) and $\mathcal{S}_{i,j}$. The SNR and PD are functions of known parameters of the agent $\Theta_{k,i} = [\sigma_i, \boldsymbol{x}_i]$, unknown parameters of the radar $\Theta_{u,j} = [P_{fa,j}, G_{R,j}, \lambda, \tau_{p,j}, T_{s,j}]^\top$ and estimated parameters of the radar $\Theta_{e,j} = \bar{\boldsymbol{x}}_{r,j}$.

Using the current estimate of the radar parameters, the overall probability of detection of the i^{th} agent is given as

$$P_{D,i}(\Theta_{k,i}, \overline{\Theta}_u, \overline{\Theta}_e) = 1 - \prod_{j=1}^{N_{\hat{r}}} (1 - P_{D,i,j}(\Theta_{k,i}, \Theta_{u,j}, \Theta_{e,j})), \tag{20}$$

where $N_{\hat{r}}$ is the current number of discovered radar, $\overline{\Theta}_u = \{\Theta_{u,j}\}_{\forall j \in \{1,\ldots,N_r\}}$ is the set of unknown radar parameters for radar systems within range of the i^{th} agent, and $\overline{\Theta}_e = \{\Theta_{e,j}\}_{\forall j \in \{1,\ldots,N_r\}}$ is the set of estimated radar parameters also for radar systems within range of the i^{th} agent.

Using the known, unknown, and estimated parameters, we wish to compute the PD and provide a covariance for that estimate. We do this by creating a first-order approximation of the PD (Equation (20)),

$$\begin{aligned} P_{D,i}(\Theta_{k,i} + \boldsymbol{\delta}_k, \overline{\Theta}_u + \boldsymbol{\delta}_u, \overline{\Theta}_e + \boldsymbol{\delta}_e) \approx \\ P_{D,i}(\Theta_{k,i}, \overline{\Theta}_u, \overline{\Theta}_e) + \boldsymbol{\delta}_k \frac{\partial P_{D,i}(\Theta_{k,i}, \overline{\Theta}_u, \overline{\Theta}_e)}{\partial \Theta_{k,i}} + \\ \boldsymbol{\delta}_u \frac{\partial P_{D,i}(\Theta_{k,i}, \overline{\Theta}_u, \overline{\Theta}_e)}{\partial \overline{\Theta}_u} + \boldsymbol{\delta}_e \frac{\partial P_{D,i}(\Theta_{k,i}, \overline{\Theta}_u, \overline{\Theta}_e)}{\partial \overline{\Theta}_e}, \end{aligned} \tag{21}$$

where $\boldsymbol{\delta}_k \in \mathbb{R}^3, \boldsymbol{\delta}_u \in \mathbb{R}^{5N_{\hat{r}}}$, and $\boldsymbol{\delta}_e \in \mathbb{R}^{3N_{\hat{r}}}$, are perturbations in known, unknown, and estimated parameters respectively. The Jacobian of Equation (20) with respect to the agent's parameters is $\partial P_{D,i}(\Theta_{k,i}, \overline{\Theta}_u, \overline{\Theta}_e)/\partial \Theta_{k,i}$. And similarly, $\Theta_{k,i}$, $\partial P_{D,i}(\Theta_{k,i}, \overline{\Theta}_u, \overline{\Theta}_e)/\partial \overline{\Theta}_u$ is the Jacobian of Equation (20) with respect to the unknown radar parameters $\overline{\Theta}_u$, and $\partial P_{D,i}(\Theta_{k,i}, \overline{\Theta}_u, \overline{\Theta}_e)/\partial \overline{\Theta}_e$ is the Jacobian of Equation (20) with respect to the estimated radar parameters $\overline{\Theta}_e$. These Jacobians can be found analytically or through automatic differentiation. In this work, we use the JAX automatic differentiation library [33].

We assume that we have a known Gaussian distribution $\Theta_{k,i} \sim \mathcal{N}(\mu_{\Theta_{k,i}}, \Sigma_{\Theta_{k,i}})$ that quantifies the uncertainty in the

agent's known parameters. To get a similar distribution for the radar's estimated parameters, we use the estimator described in Section V-A to obtain mean value and covariance estimates for each discovered radar $\overline{R}$. We then combine these into a mean vector

$$\mu_{\overline{\Theta}_e} = \begin{bmatrix} \mu_{\Theta_{e,1}} \\ \mu_{\Theta_{e,2}} \\ \vdots \\ \mu_{\Theta_{e,N_{\hat{r}}}} \end{bmatrix} \tag{22}$$

and covariance matrix

$$\Sigma_{\overline{\Theta}_e} = \begin{bmatrix} \Sigma_{\Theta_{e,1}} & \mathbf{0} & \dots & \mathbf{0} \\ \mathbf{0} & \Sigma_{\Theta_{e,2}} & \dots & \mathbf{0} \\ \vdots & \dots & \ddots & \vdots \\ \mathbf{0} & \dots & \dots & \Sigma_{\Theta_{e,N_{\hat{r}}}} \end{bmatrix}, \tag{23}$$

where the individual radar estimates are uncorrelated and the combined radar parameters are normally distributed $\overline{\Theta}_e \sim \mathcal{N}(\mu_{\overline{\Theta}_e}, \Sigma_{\overline{\Theta}_e})$.

Because we cannot estimate the unknown parameters, we must use prior knowledge of the system to create reasonable prior beliefs of these parameters. We assume a normal distribution and represent the mean value of this prior belief as

$$\mu_{\overline{\Theta}_u} = \begin{bmatrix} \mu_{\Theta_{u,1}} \\ \mu_{\Theta_{u,2}} \\ \vdots \\ \mu_{\Theta_{u,N_{\hat{r}}}} \end{bmatrix} \tag{24}$$

and covariance matrix as

$$\Sigma_{\overline{\Theta}_u} = \begin{bmatrix} \Sigma_{\Theta_{u,1}} & \mathbf{0} & \dots & \mathbf{0} \\ \mathbf{0} & \Sigma_{\Theta_{u,2}} & \dots & \mathbf{0} \\ \vdots & \dots & \ddots & \vdots \\ \mathbf{0} & \dots & \dots & \Sigma_{\Theta_{u,N_{\hat{r}}}} \end{bmatrix}, \tag{25}$$

where $\mu_{\Theta_{u,j}}$ and $\Sigma_{\Theta_{u,j}}$ are the mean value and covariance for the unknown parameters of the j^{th} radar station and the combined unknown parameters are normally distributed $\overline{\Theta}_u \sim \mathcal{N}(\mu_{\overline{\Theta}_u}, \Sigma_{\overline{\Theta}_u})$. In this work, we use the same prior distribution for each radar's unknown parameters ($\mu_{\Theta_{u,j}} = \mu_{\Theta_{u,i}}, \Sigma\Theta_{u,j} = \Sigma\Theta_{u,i}, \forall (i,j) \in \{1, \dots, N_{\hat{r}}\}$).

Using these distributions and the linearized model of PD we can find the mean value of the PD as

$$\mu_{P_{D,i}}(\Theta_{k,i}, \overline{\Theta}_u, \overline{\Theta}_e) = P_{D,i}(\mu_{\Theta_{k,i}}, \mu_{\overline{\Theta}_u}, \mu_{\overline{\Theta}_e}). \tag{26}$$

and the variance

$$\sigma^2_{P_{D,i}}(\Theta_{k,i}, \overline{\Theta}_u, \overline{\Theta}_e) = \tag{27}$$

$$\frac{\partial P_{D,i}(\Theta_{k,i}, \overline{\Theta}_u, \overline{\Theta}_e)}{\partial \Theta_{k,i}} \Sigma_{\Theta_{k,i}} \frac{\partial P_{D,i}(\Theta_{k,i}, \overline{\Theta}_u, \overline{\Theta}_e)}{\partial \Theta_{k,i}}^\top + \frac{\partial P_{D,i}(\Theta_{k,i}, \overline{\Theta}_u, \overline{\Theta}_e)}{\partial \overline{\Theta}_u} \Sigma_{\overline{\Theta}_u} \frac{\partial P_{D,i}(\Theta_{k,i}, \overline{\Theta}_u, \overline{\Theta}_e)}{\partial \overline{\Theta}_u}^\top + \frac{\partial P_{D,i}(\Theta_{k,i}, \overline{\Theta}_u, \overline{\Theta}_e)}{\partial \overline{\Theta}_e} \Sigma_{\overline{\Theta}_e} \frac{\partial P_{D,i}(\Theta_{k,i}, \overline{\Theta}_u, \overline{\Theta}_e)}{\partial \overline{\Theta}_e}^\top. \tag{28}$$

Using this mean and covariance, we can create an approximate distribution for PD $P_{D,i} \sim \mathcal{N}(\mu_{P_{D,i}}, \sigma^2_{P_{D,i}})$. We can then use this distribution to approximate the probability of the true PD being below a threshold:

$$P(P_D \leq P_{D,t}) = \Phi\left(P_{D,t}, \mu_{P_D}(\Theta_{k,i}, \overline{\Theta}_u, \overline{\Theta}_e), \sigma^2_{P_D}(\Theta_{k,i}, \overline{\Theta}_u, \overline{\Theta}_e)\right), \tag{29}$$

where

$$\Phi(x; \mu, \sigma^2) = \frac{1}{2}\left[1 + \text{erf}\left(\frac{x-\mu}{\sqrt{2}\sigma}\right)\right] \tag{30}$$

is the normal cumulative distribution function and

$$\text{erf}(z) = \frac{2}{\sqrt{\pi}} \int_0^z e^{-t^2}\, dt. \tag{31}$$

VI. Low-Priority Agent Path Planning

The objective of low priority agents is to explore the operational area, discover unknown enemy radar, and find a safe path for the high priority agent. We employ a decentralized optimization strategy. Each agent uses the current information it has received about the environment to plan its own path. This path information is transmitted to other agents, who then plan their paths. The paths are planned in a round-robin fashion, where the agent closest to the most uncertain radar station plans their path first.

A. Objective Function

The agents' goals of discovering enemy radar, better localizing discovered radar, and finding the optimal path for the high-priority agent lead to a three-part objective function. First, agents should explore unexplored regions. We encourage this through a Bayesian approach described later in Equation (38). Second, agents need to reduce the uncertainty of the previously discovered radar. Agents accomplish this through the use of an objective that incentivizes choosing waypoints that will reduce the uncertainty in the EKF estimate of the radar parameters. This objective will be quantified below in Equation (40). Finally, agents are searching for the optimal (minimum-time) trajectory for the high-priority agent. This optimal trajectory is most likely found near the strait line path between the starting location, $\boldsymbol{x}_{h_0}$, and the goal location, $\boldsymbol{x}_{h_f}$, of the high priority agent. We penalize the objective function by the distance the agent is from the straight-line path encourage exploration near the strait line trajectory, as shown in Equation (41).

We use a non-linear interior point optimization algorithm called IPOPT [34], to find waypoints for the low-priority agents. We constructed an objective function that applies the trade-off between exploiting known information and exploring for new information. Our objective function has three parts: exploration, radar parameter uncertainty reduction, and goal-directed search prioritization for the high-priority agent.

The overall objective function is the sum of the three part:

$$\Gamma(\boldsymbol{x}, \overline{X}_e, \overline{R}) = -\alpha_e \Gamma_e(\boldsymbol{x}, \overline{X}_e) + \alpha_u \Gamma_u(\boldsymbol{x}, \overline{R}) + \alpha_s \Gamma_s(\boldsymbol{x}), \tag{32}$$

where α_e, α_u, and α_s are the weights for the exploration, uncertainty reduction, and goal objective functions, respectively. Vector $\overline{X}_e$ is the combined path history of all agents, and $\overline{R}$ is the current output of the EKF (described in Section V-A) that

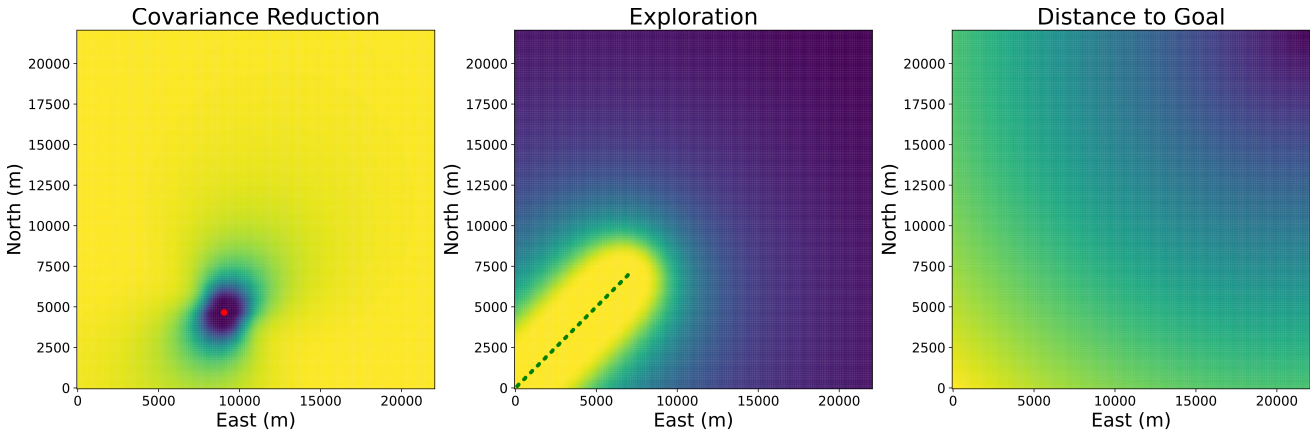

Fig. 1. This figure shows the three different parts of the objective function. The left figure shows the covariance reduction objective function $\Gamma_u(\boldsymbol{x})$. The estimated radar location is plotted in red. From this figure we see that the best measurement locations to reduce a radar's covariance uncertainty, according to this objective function, come from being closer to the radar, and from going perpendicular to the current measurements. The middle figure illustrates the exploration objective function $\Gamma_e(\boldsymbol{x})$, where the green points show locations the agents have already explored. Going further from the explored area provides better measurements. The final plot shows the distance to goal objective, where the goal is in the upper right corner of the plot.

is a list of means and covariances for each discovered radar. Each part of the objective function is shown in Figure 1.

Every agent keeps a list of locations that they have explored: $\overline{X}_{e,i} = \{\boldsymbol{x}_{l,i}(t_0), \boldsymbol{x}_{l,i}(t_0+\Delta_{t_e}), \boldsymbol{x}_{l,i}(t_0+2\Delta_{t_e}), \ldots, \boldsymbol{x}_{l,i}(t_0+N_e\Delta_{t_e})\}$, where $\boldsymbol{x}_{l,i}(t)$ is the state of i^{th} low priority agent at time t, t_0 is the time when the agent started its mission, Δ_{t_e} is the time interval between path history points, and N_e is the current number of stored path history points ($N_e = \text{int}(t_c/\Delta_{t_e})$). The individual path history list of every agent is combined into a single list $\overline{X}_e = \bigcup_{i=1}^{N_l} \overline{X}_{e,i}$.

From this combined path history, we will approximate the probability that an undiscovered radar is present at a location $\boldsymbol{x}$ given the locations that the agents have visited. We do this using Bayes rule and approximating the probability that an agent intercepts an enemy radar signal. Given an agent was at location $\boldsymbol{x}_{l,i}(t_j)$, and that a radar is at location $\boldsymbol{x}$, the probability the agent would have intercepted a signal is given by:

$$P(\textit{intercept at } \boldsymbol{x}_{l,i}(t_j) | \textit{radar at } \boldsymbol{x}) = \exp \frac{\ln(P_{fa})}{\mathcal{S}\left(\boldsymbol{x}_{l,i}(t_j), \boldsymbol{x}\right) + 1}, \tag{33}$$

where P_{fa} is the probability of false alarm, and $\mathcal{S}(\boldsymbol{x}_i, \boldsymbol{x})$ is the signal to noise ratio (SNR) between the agent and the radar. The SNR is given by:

$$\mathcal{S}(\boldsymbol{x}_{l,i}(t_j), \boldsymbol{x}) = \frac{P_T G_T G_{I,i} \lambda^2 \tau_p}{(4\pi)^2 ||\boldsymbol{x}_{l,i}(t_j) - \boldsymbol{x}||_2^2 \kappa T_s L \delta_i}. \tag{34}$$

Since the parameters of the potentially undiscovered radar are unknown, we assume default values for the transmitted power, P_T, the transmit gain, G_T, the pulse width, τ_p, the system temperature, T_s, and the loss factor, L. However, we have prior knowledge of the agent's intercept antenna gain, $G_{I,i}$ and the signal wavelength, λ. In addition, we introduce a discount term, δ_l, to account for signal degradation caused by factors such as misaligned antennae, mismatched filters, and other unknown losses. The probability of failing to intercept a signal at $\boldsymbol{x}_i$, given that a radar is located at $\boldsymbol{x}$, is related to the probability of intercepting by $P(\textit{no intercept at } \boldsymbol{x}_i | \textit{radar at } \boldsymbol{x}) = 1 - P(\textit{intercept at } \boldsymbol{x}_i | \textit{radar at } \boldsymbol{x})$.

We wish to approximate the probability that a radar could be found at location $\boldsymbol{x}$ given that the agents did not intercept signals at the locations $\overline{X}_e$. Using Bayes rule, this is given by:

$$P(\textit{radar at } \boldsymbol{x} | \bigcup \textit{no intercept at } \boldsymbol{x}_i, \boldsymbol{x}_i \in \overline{X}_e) = \frac{P(\bigcup \textit{no intercept at } \boldsymbol{x}_i, \boldsymbol{x}_i \in \overline{X}_e | \textit{radar at } \boldsymbol{x}) P(\textit{radar at } \boldsymbol{x})}{P(\bigcup \textit{no intercept at } \boldsymbol{x}_i, \boldsymbol{x}_i \in \overline{X}_e)}. \tag{35}$$

If we assume that intercepting a signal at $\boldsymbol{x}_i$ is independent of intercepting a signal at a different location, given that there is a radar at $\boldsymbol{x}$, then

$$P(\bigcup \textit{no intercept at } \boldsymbol{x}_i, \boldsymbol{x}_i \in \overline{X}_e | \textit{radar at } \boldsymbol{x}) = \prod_{\boldsymbol{x}_i \in \overline{X}_e} P(\textit{no intercept at } \boldsymbol{x}_i | \textit{radar at } \boldsymbol{x}). \tag{36}$$

We assume $P(\textit{radar at } \boldsymbol{x}) = 0.5$, meaning there are equal chances that a radar is present at $\boldsymbol{x}$ or not. If additional information about radar locations in the region were available, it could be used to refine this prior probability. To find the denominator, we use a partition:

$$\begin{aligned} &P(\bigcup \textit{no intercept at } \boldsymbol{x}_i, \boldsymbol{x}_i \in \overline{X}_e) = \\ &P(\bigcup \textit{no intercept at } \boldsymbol{x}_i, \boldsymbol{x}_i \in \overline{X}_e | \textit{radar at } \boldsymbol{x}) P(\textit{radar at } \boldsymbol{x}) + \\ &P(\bigcup \textit{no intercept at } \boldsymbol{x}_i, \boldsymbol{x}_i \in \overline{X}_e | \textit{no radar at } \boldsymbol{x}) P(\textit{no radar at } \boldsymbol{x}). \end{aligned} \tag{37}$$

We know $P(\textit{no intercept at } \boldsymbol{x}_i | \textit{no radar at } \boldsymbol{x}) = 1 - P_{f_a}$. Assuming that not intercepting a signal at one location is independent of not intercepting a signal at another location when no radar is present, we can write $P(\bigcup \textit{no intercept at } \boldsymbol{x}_i, \boldsymbol{x}_i \in \overline{X}_e | \textit{no radar at } \boldsymbol{x}) = (1 - P_{f_a})^{|\overline{X}_e|}$, where $|\overline{X}_e|$ is the cardinality of $\overline{X}_e$.

Combing all this, for our exploration objective, we wish for the low priority agents to maximize the likelihood that an undiscovered radar exists at a location $\boldsymbol{x}$

$$\Gamma_e(\boldsymbol{x}, \overline{X}_e) = \frac{\prod_{\boldsymbol{x}_i \in \overline{X}_e} (1 - \exp \frac{\ln(P_{fa})}{\mathcal{S}(\boldsymbol{x}_i, \boldsymbol{x})+1}) \Phi(\boldsymbol{x})}{\prod_{\boldsymbol{x}_i \in \overline{X}_e} (1 - \exp \frac{\ln(P_{fa})}{\mathcal{S}(\boldsymbol{x}_i, \boldsymbol{x})+1}) \Phi(\boldsymbol{x}) + (1 - P_{fa})^{|\overline{X}_e|} (1 - \Phi(\boldsymbol{x}))}, \tag{38}$$

where $\Phi(\boldsymbol{x})$ is a function that computes the prior probability $P(\textit{radar at } \boldsymbol{x})$ given a location $\boldsymbol{x}$. For this paper, we assume $\Phi(\boldsymbol{x}) = 0.5$ for all $\boldsymbol{x}$, however, if more prior information is available, it could be incorporated into $\Phi(\boldsymbol{x})$. This part of the objective function is shown in the left image of Figure 1.

The second part of the objective function, $\Gamma_u(\boldsymbol{x})$, guides agents toward areas where radar measurements are more informative and will improve localization. We do this by noting that when a measurement is incorporated into an EKF, the covariance does not depend on the value that is measured, only on the location at which the measurement was taken. This part of the objective function incentives reducing the determinant of the covariance of the radar parameter estimate that would result from taking a measurement were taken at a given location. This part of the objective function is the same as was used in [2].

From the estimator, we have a list of the current estimated mean and covariance values for the radar parameters $\overline{R}$. The equation for the updated covariance when incorporating an additional measurement is:

$$\text{cov}^+(\boldsymbol{x}, \mu_{\boldsymbol{x}_{r,j}}, \Sigma_{\boldsymbol{x}_{r,j}}) = (I - K J_h(\boldsymbol{x}, \mu_{\boldsymbol{x}_{r,j}})) \Sigma_{\bar{\boldsymbol{x}}_{r,j}}, \tag{39}$$

where K is the Kalman gain defined in Algorithm 7, I is the identity matrix, and J_h is the Jacobian of the measurement model. Given a list of models–each defined by their mean and covariance–for all the discovered radars, we aim to find the measurement location that would most improve all the models. To do this, we compute the mean of the determinant of all the updated covariances that would result if a measurement were taken at location $\boldsymbol{x}$. This yields,

$$\Gamma_u(\boldsymbol{x}, \overline{R}) = \frac{1}{N_{\hat{r}}} \sum_{(\mu_{\boldsymbol{x}_{r,j}}, \Sigma_{\boldsymbol{x}_{r,j}}) \in \overline{R}} \frac{\det(\text{cov}^+(\boldsymbol{x}, \mu_{\boldsymbol{x}_{r,j}}, \Sigma_{\boldsymbol{x}_{r,j}}))}{d_{cov}}, \tag{40}$$

where d_{cov} is a normalizing term that controls how much the objective function prioritizes reducing the covariance. Once the mean value of the determinant of the future covariances is much less than d_{cov}, the covariance reduction objective will no longer affect the objective function. This allows agents to ignore radar stations that are localized well enough, where well enough is defined by d_{cov}. At location $\boldsymbol{x}$, it is not likely that a measurement will be received for each model; however, this objective is a good heuristic to see how traveling to a certain location will improve the current estimates of the enemy radar. This part of the objective function is visualized in the center image of Figure 1.

The next part of the objective function $\Gamma_s(\boldsymbol{x})$ incentives the low-priority agents to explore in the direction of the high-priority agent's goal. The objective is the normalized distance to the high-priority goal,

$$\Gamma_s(\boldsymbol{x}) = \frac{\|\boldsymbol{x} - \boldsymbol{x}_{h_f}\|_2}{d_{max}}, \tag{41}$$

where d_{max} is the maximum distance in the region $\mathcal{D}$ from the goal $\boldsymbol{x}_{h_f}$. This portion of the objective function is shown in the right image of Figure 1.

Equation (32) shows how the three parts of the objective function are combined using the weights α_u, α_e, and α_s. Because we minimize Equation (32), we subtract the exploration objective function, Equation (38). This results in maximizing the likelihood of discovering an undiscovered radar.

B. Multi-Agent Optimization

Given multiple low-priority agents, we require a coordination strategy that prevents all agents from converging on the same high-reward locations. To address this, we employ a decentralized strategy in which each agent selects its optimal waypoint $\boldsymbol{x}^w_{l,i}$ (for the i^{th} low-priority agent). The agent's travel towards their waypoint in a straight-line (or minimum-path?) trajectory. This approach is only effective if we deconflict the waypoints each agent chooses.

The order of waypoint selection is determined by each agent's distance to the radar with the highest uncertainty (as measured by the determinant of the covariance). Waypoint optimization is triggered either when an agent reaches its current waypoint or through a receding horizon (RH) scheme. Additionally, if a fixed time interval T_h has elapsed since an agent has re-planned its path, waypoint optimization is re-initialized. Each agent picks its next waypoint $\boldsymbol{x}^w_{l,i}$ by minimizing Equation (32). We employ a decentralized method by allowing agents to pick their waypoint based on the information they have received, and then computing their future path, and sharing that with other agents.

Algorithm 2 shows our low-priority path planning scheme. The input to our algorithm is the current estimate of the radar parameters and locations $\overline{R}$ and the current combined path history of all agents $\overline{X}_e$. The output is a waypoint $\boldsymbol{x}^w_{l,i}$ for each low-priority agent. We use a receding horizon scheme; If the time since an agent has last planned its path $t_{h,i}$ is greater than the receding horizon time T_h, a new waypoint optimization is triggered. The most uncertain radar (based on the determinant of its covariance) is found in line 6. The distance between this radar and all agents is then found in line 7. The agents are then sorted by distance to the most uncertain radar in line 8. Agents optimize waypoints in order of which is closest to the most uncertain radar to ensure that the closest agents are tasked with reducing the uncertainty of the most uncertain radar. Then each agent is looped through in this order. The i^{th} agent then optimizes its waypoint $\boldsymbol{x}^w_{l,i}$ based on Equation (32). To enable decentralized path planning, each agent plans its waypoint based on the information it has received. After picking a waypoint, the agent samples the path between its current location $\boldsymbol{x}_{l,i}(t)$ and the waypoint to simulate where it will travel in the future (line 12). This future path information is combined with the agent's current

Algorithm 2 Multi-agent Path Planning for Low Priority Agents

1: **Input:** $\overline{R}, \overline{X}_e$
2: **Output:** $\boldsymbol{x}^w_{l,i}\ \forall i \in \{1,\dots,N_l\}$
3: $t_{h,i} = T_h\ \forall i \in \{1,\dots,N_l\}$
4: **while** High priority path not found **do**
5: **if** $t_{h,i} > T_h$ **for any** $i \in \{1,\dots,N_l\}$ **then**
6: $j = \texttt{argmax}_{(\mu_{\boldsymbol{x}_{r,j}},\Sigma_{\boldsymbol{x}_{r,j}})\in\overline{R}} \det(\Sigma_{\boldsymbol{x}_{r,j}})$
7: $\overline{D}_{i,j} = \{||\boldsymbol{x}_{l,i}(t) - \mu_{\boldsymbol{x}_{r,j}}||_2\}_{\forall i\in\{1,\dots,N_l\}}$
8: $\overline{I} = \texttt{argsort}(\overline{D}_{i,j})$
9: $\overline{R}^+ \leftarrow \overline{R},\ \overline{X}^+_e \leftarrow \overline{X}_e$
10: **for** $i \in \overline{I}$ **do**
11: $\boldsymbol{x}^w_{l,i} = \texttt{argmin}_{\boldsymbol{x}\in\mathcal{D}}(\Gamma(\boldsymbol{x}, \overline{X}^+_e, \overline{R}^+))$
12: $\overline{X}^+_{e,i} = \left\{\boldsymbol{x} | \boldsymbol{x} = q\frac{\boldsymbol{x}^w_{l,i} - \boldsymbol{x}_{l,i}(t)}{||\boldsymbol{x}^w_{l,i} - \boldsymbol{x}_{l,i}(t)||_2}\Delta_{t_e} v_l\ \forall q \in \{1\dots,\texttt{int}\left(\frac{||\boldsymbol{x}^w_{l,i} - \boldsymbol{x}_{l,i}(t)||_2}{\Delta_{t_e} v_l}\right)\}\right\}$ {Sample future path}
13: $\overline{X}^+_e = \overline{X}^+_e \bigcup \overline{X}^+_{e,i}$
14: $\overline{R}^+ = \{(\mu_{\boldsymbol{x}_{r,k}}, \text{cov}^+(\boldsymbol{x}^w_{l,i}, \mu_{\boldsymbol{x}_{r,k}}, \Sigma_{\boldsymbol{x}_{r,k}})\forall k \in \{1,\dots,N_{\hat{r}}\}\}$
15: $\texttt{transmit}(\overline{R}^+, \overline{X}^+_e)$
16: $t_{h,i} = 0$
17: **end for**
18: **end if**
19: **end while**

knowledge of its past path and the past paths of other agents in line 13. To approximate the effect that the potential future measurement taken at $\boldsymbol{x}^w_{l,i}$ will have on the estimated radar parameter covariances, we compute what the future covariance would be for each estimated radar if a measurement were taken at the waypoint in line 14. The future path $\overline{X}^+_e$ and the future covariance information $\overline{R}^+$ are then transmitted to all agents in range. To reduce communication bandwidth, measures could be taken, such as sending only information that has not already been transmitted, such as the communication protocols described in [35]. This process is repeated until all agents have optimized their waypoints.

VII. High-Priority Path Planning

The goal of the high priority path planner is to find a safe path that starts at $\boldsymbol{x}_{h_0}$ and ends at $\boldsymbol{x}_{h_f}$, while keeping the PD along the trajectory less than a threshold $P_{D,t}$. The high priority agent does not have perfect knowledge of the region, it only has estimates of the radar parameters $\overline{R}$ that the low priority agents have found. The high-priority agent must account for this uncertainty when planning trajectories. We do this by approximating the probability that the true probability of detection (Equation (20)) is less than a certain threshold (Equation (29)). In this work, we present two path-planning methods for the high-priority agent, one for the deterministic case, where the radar parameters are all known, and the other for the uncertain case, where the radar parameter estimates and prior beliefs are used. Both rely on the use of Voronoi diagrams to find initial feasible trajectories, then using interior point optimization algorithms (IPOPT [34]) to optimize a B-spline trajectory which accounts for the path safety (PD) and kinematic feasibility constraints (velocity, curvature, turn rate).

A. Deterministic High-Priority Path Planner

For the deterministic case, we assume that the high-priority agent has knowledge of all the radar's parameters, $\boldsymbol{x}_{r,j}, P_{T,j}, G_{T,j}, G_{R,j}, \lambda, \tau_{p,j}, T_{s,j}$ and L_j. Using this information, we wish to find a feasible trajectory where the probability of detection along the entire trajectory is below a threshold $P_{D,t}$, e.g., $P_D \leq P_{D,t}$,. To do this, we first use a multiplicatively weighted Vornoi diagram along with the A-star graph search algorithm [36]. We then fit a B-spline trajectory to the path and use a heuristic to ensure that velocity constraints are met. Finally, we use that B-spline trajectory as a seed trajectory to an interior point optimization algorithm, which finds the minimum time trajectory with PD, velocity, turn rate, and curvature constraints.

To find an initial feasible trajectory, we use a multiplicatively weighted Voronoi diagram. We show that the minimum PD boundary between two radars is equivalent to finding the edges of a multiplicatively weighted Voronoi cell. We use this to find the boundary between two radar stations where the PD from each radar station is equal. For the j^{th} and k^{th} radars, this yields

$$\exp\frac{\ln P_{fa,j}}{\frac{P_{E,j}G_{R,j}\lambda^2\sigma_i\tau_{p,j}}{(4\pi)^3R^4_{i,j}\kappa T_{s,j}L_j}+1} = \exp\frac{\ln P_{fa,k}}{\frac{P_{E,k}G_{R,k}\lambda^2\sigma_i\tau_{p,k}}{(4\pi)^3R^4_{i,k}\kappa T_{s,k}L_k}+1}. \quad (42)$$

If we assume that $P_{fa,j} = P_{fa,k}$, finding the boundary where the PD is equal for both radars is the same as finding the region where the SNR for both radars is equal:

$$\frac{P_{E,j}G_{R,j}\lambda^2\sigma_i\tau_{p,j}}{(4\pi)^3R^4_{i,j}\kappa T_{s,j}L_j} = \frac{P_{E,k}G_{R,k}\lambda^2\sigma_i\tau_{p,k}}{(4\pi)^3R^4_{i,k}\kappa T_{s,k}L_k}. \quad (43)$$

Rearranging this equation to match the multiplicatively weighted Voronoi diagram, we get:

$$\frac{R_{i,j}}{(\frac{\kappa T_{s,j}L_j}{P_{E,j}G_{R,j}\lambda^2\sigma_i\tau_{p,j}})^{\frac{1}{4}}} = \frac{R_{i,k}}{(\frac{\kappa T_{s,k}L_k}{P_{E,k}G_{R,k}\lambda^2\sigma_i\tau_{p,k}})^{\frac{1}{4}}}. \quad (44)$$

This means the boundary (path) between two radars with minimum PD is equivalent to the multiplicatively weighted Voronoi edge generated using the radar locations as generator points where the weight for the j^{th} radar is

$$w_j = \left(\frac{\kappa T_{s,j}L_j}{P_{E,j}G_{R,j}\lambda^2\sigma_i\tau_{p,j}}\right)^{\frac{1}{4}}. \quad (45)$$

We denote the set of Voronoi edges generated using the radar locations $\overline{X}_r = \{\boldsymbol{x}_{r,1},\dots,\boldsymbol{x}_{r,N_r}\}$ and the weights $\overline{W}_r$ found using Equation (45) as $\mathcal{E}(\overline{X}_r, \overline{W}_r)$.

Algorithm 3 presents our high-priority path planning method. It takes as input the radar locations and their associated weights, which are computed using Equation (45). The algorithm outputs a set of control points, $\overline{C}_{opt}$, along with knot points (determined using the final time t_f) that define an optimized B-spline trajectory.

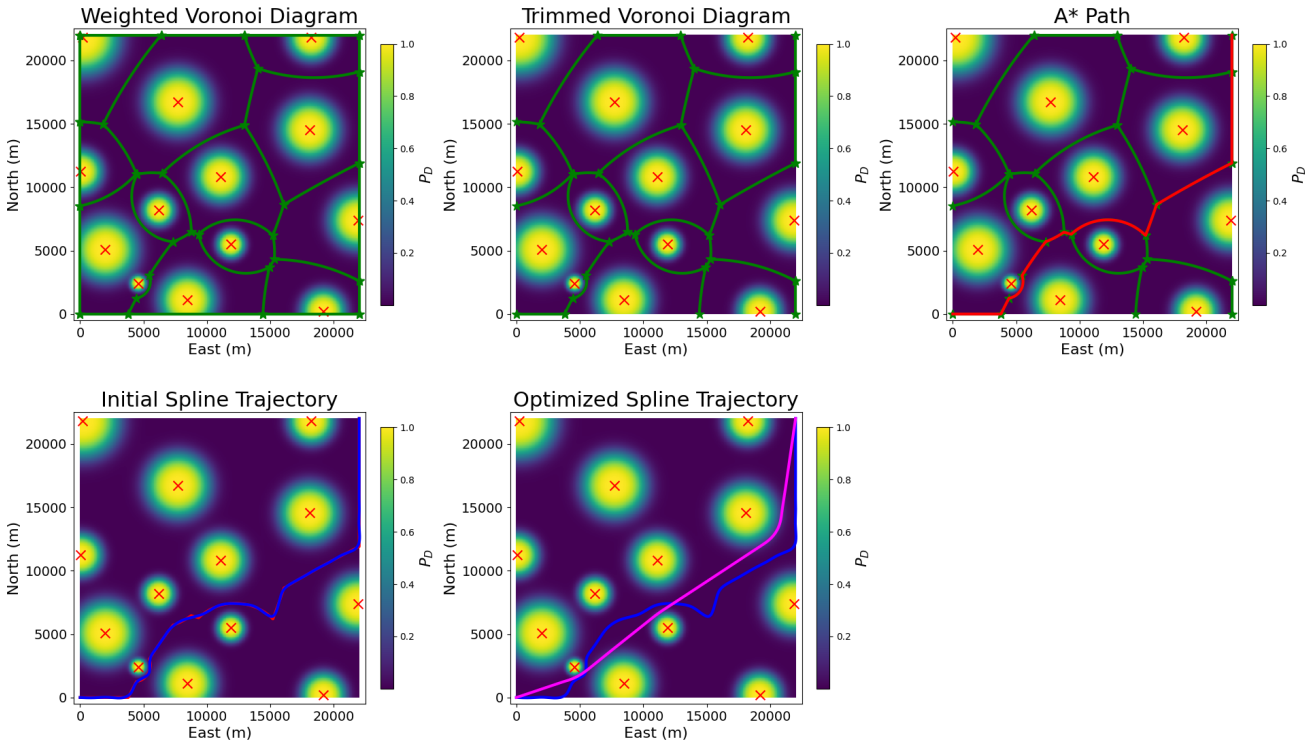

Fig. 2. This figure shows the steps of our deterministic high-priority path planning algorithm (Algorithm 3). In the top left figure, the weighted Voronoi diagram is found. Then all edges where $P_D \leq P_{D,t}$ for any point along the edge are removed. In the top right figure, A-star is used to find the shortest path through the graph. In the bottom left, a B-spline is fit to the A-star path. Then, in the bottom center, the optimized B-spline is shown. All figures show the locations of the radar as red x's with the PD at every location depicted by the heatmap.

Algorithm 3 Deterministic High-Priority Path Planning

1: **Inputs:** $\overline{X}_r, \overline{W}_r, \boldsymbol{x}_{h_0}, \boldsymbol{x}_{h_f}$
2: **Output:** $\overline{C}, \boldsymbol{t}_k$
3: $\mathcal{E} = \texttt{computeWeightedVoronoiEdges}(\overline{X}_r, \overline{W}_r)$
4: $\mathcal{V} = \texttt{computeWeightedVoronoiVertices}(\overline{X}_r, \overline{W}_r)$
5: $\mathcal{V}, \mathcal{E} = \texttt{intersectVoronoiEdgesWithBoundary}(\mathcal{V}, \mathcal{E})$
6: $\mathcal{E} = \texttt{trimInfeasibleEdges}(\mathcal{E})$
7: $\mathcal{V}_{opt}, \mathcal{E}_{opt} = \texttt{shortestPathThroughGraph}(\mathcal{V}, \mathcal{E}, \boldsymbol{x}_{h_0}, \boldsymbol{x}_{h_f})$
8: $\overline{C}_0 = \texttt{fitSplineToPath}(\mathcal{V}_{opt}, \mathcal{E}_{opt}, N_c, p)$
9: $t_f = \texttt{checkVelocityConstraint}(\overline{C}_0, v_{lb}, v_{ub})$
10: $\overline{C}_{opt}, t_f = \texttt{optimizeTrajectory}(\overline{C}_0, t_f, \boldsymbol{x}_{h_0}, \boldsymbol{x}_{h_f})$

Computing Weighted Voronoi Diagrams. The algorithm's first step is to compute the weighted Voronoi diagram, specifically its edges $\mathcal{E}$ and vertices $\mathcal{V}$, using the method outlined in [30]. These diagrams represent proximity relationships among the radar nodes, factoring in their weights.

Intersecting Voronoi Edges with the Boundary. In line 5, we intersect the Voronoi edges with the boundary of the operating region $\mathcal{D}$. For a rectangular region defined by lower and upper bounds $\boldsymbol{x}_{lb}$ and $\boldsymbol{x}_{ub}$, we check each edge in $\mathcal{E}$ for intersections with the four sides of the region. Intersections result in new vertices added to $\mathcal{V}$, and the corresponding edges are trimmed to lie within the region. Additionally, edges between boundary-intersecting vertices are added to $\mathcal{E}$ to preserve connectivity.

Trimming Edges Based on PD Constraints. In line 6, we remove the edges of $\mathcal{E}$ that have a maximum PD along the path above that of the threshold $P_{D,t}$. For each edge in $\mathcal{E}$, we find the point closest to the generator points corresponding to the edge. Each edge has two corresponding generator points, and the closest point on the edge to both points will be the same. Each edge is an arc of a circle defined with a center $\boldsymbol{x}_{e,i}$, radius $r_{e,i}$, start angle $\theta_{0,e,i}$ and stop angle $\theta_{f,e,i}$. The closest point on the arc to the generator point $\boldsymbol{x}_{r,j}$ is

$$\theta_{i,j} = \arctan\left(\frac{y_{r,j} - y_{e,i}}{x_{r,j} - x_{e,i}}\right)$$

$$\boldsymbol{x}_{i,j} = \begin{cases} \boldsymbol{x}_{e,i} + r_{e,i}\begin{bmatrix}\cos\theta_{i,j}\\ \sin\theta_{i,j}\end{bmatrix}, \theta_{0,e,i} \leq \theta_{i,j} \leq \theta_{f,e,i} \\ \boldsymbol{x}_{e,i} + r_{e,i}\begin{bmatrix}\cos\theta_{0,e,i}\\ \sin\theta_{0,e,i}\end{bmatrix}, |\theta_{0,e,i} - \theta_{i,j}| \leq |\theta_{f,e,i} - \theta_{i,j}| \\ \boldsymbol{x}_{e,i} + r_{e,i}\begin{bmatrix}\cos\theta_{f,e,i}\\ \sin\theta_{f,e,i}\end{bmatrix}, |\theta_{f,e,i} - \theta_{i,j}| \leq |\theta_{f,e,i} - \theta_{i,j}|. \end{cases} \tag{46}$$

The PD at this closest point is found using Equation (20). If the PD is greater than the threshold $P_{D,t}$, that edge is removed from $\mathcal{E}$.

Graph Construction and Path Search. Using the trimmed set of vertices $\mathcal{V}$ and edges $\mathcal{E}$ a graph is created. We then use the A-star algorithm [36] to find the shortest path through the graph, where the distance between the nodes is the arc distance of the edge connecting the nodes, starting at the node corresponding to $\boldsymbol{x}_{h_0}$ and ending at $\boldsymbol{x}_{h_f}$. This path is defined as an ordered list of vertices $\mathcal{V}_{opt}$ and edges $\mathcal{E}_{opt}$.

Spline Fitting and Time Adjustment. We then fit a B-spline to the shortest path (line 8). The path is first sampled, and a B-spline is fitted with N_c control points, degree p, and internal knots spaced over the interval $[0, 1]$. To satisfy velocity constraints, we use a heuristic approach: we sample the spline's velocity using Equation (12) and check its maximum value. If the constraint is violated, t_f is increased, knot points are recomputed over $[0, t_f]$, and the process is repeated until the velocity constraint is satisfied.

Trajectory Optimization. Finally, an interior-point optimization algorithm (IPOPT [34]) refines the trajectory to minimize travel time, while conforming to physical feasibility constraints. The optimization problem is:

$$\overline{C}_{opt} = \underset{\overline{C}, t_f}{\operatorname{argmin}}\, t_f \tag{47a}$$

$$\text{subject to } \boldsymbol{p}(0) = \boldsymbol{x}_{h_0} \tag{47b}$$

$$\boldsymbol{p}(t_f) = \boldsymbol{x}_{h_f} \tag{47c}$$

$$\boldsymbol{p}(\boldsymbol{t}_s) \in \mathcal{D} \tag{47d}$$

$$v_{lb} \leq v(\boldsymbol{t}_s) \leq v_{ub} \tag{47e}$$

$$u_{lb} \leq u_A(\boldsymbol{t}_s) \leq u_{ub} \tag{47f}$$

$$-\kappa_{ub} \leq \kappa_A(\boldsymbol{t}_s) \leq \kappa_{ub} \tag{47g}$$

$$P_D(\boldsymbol{p}(\boldsymbol{t}_s)) \leq P_{D,t}, \tag{47h}$$

where $\boldsymbol{t}_s = \{0, \Delta_s, 2\Delta_s, \ldots, t_f\}$ are the discrete points in time where the constraints are evaluated, $\Delta_s = t_f / N_s$ is the time spacing between constraint samples and N_s is the number of constraint samples. We discretely sample the constraints, which means that the constraints could be violated between samples. However, the likelihood of this is reduced with an increasing number of samples.

The objective of Equation (47a), is to find the minimum time trajectory, starting at $\boldsymbol{x}_{h_0}$ and ending at $\boldsymbol{x}_{h_f}$. The constraint in Equation (47d) ensures that the agent stays within the operating region $\mathcal{D}$. The next three constraints, Equations (47e) through Equation (47g), show the constraints of kinematic feasibility, velocity, turn rate, and curvature. These ensure that the agent can physically follow the planned trajectory. The final constraint, Equation (47h), ensures that the PD of the agent (found using Equation (20)) is below the threshold. The steps of this algorithm are shown in Figure 2.

B. Uncertain High-Priority Path Planner

The previous section handled the case where the radar parameters are known by the agent. In this section, we present a trajectory planning algorithm that handles the case where the agent has uncertain estimates and prior beliefs of the radar parameters. Our approach parallels the deterministic case, but instead of forcing a threshold on PD, we have impose a constraint on the likelihood of the true PD falls below the threshold $(P(P_D \leq P_{D,t}) \geq \epsilon)$, which is found using Equation (29)).

The key difference between our deterministic and uncertain planners lies in the generation of the initial feasible trajectory. While we still use Voronoi diagrams, we use a generalized Voronoi diagram rather than a multiplicatively weighted one. The metric that generates the diagram is the likelihood that the true PD is below the threshold $P(P_D \leq P_{D,t})$ found from the approximate PD distributions created using the mean and variance from Equations (26) and (27).

The remainder of the algorithm proceeds the same as in the deterministic case, with the exception of the path safety constraint used.

Algorithm 4 Uncertain High-Priority Path Planning

1: **Inputs:** $\overline{R}_t, \overline{X}_{e,t}, \boldsymbol{x}_{h_0}, \boldsymbol{x}_{h_f}$
2: **Output:** $\overline{C}, \boldsymbol{t}_k$
3: $\mathcal{E}, \mathcal{V}$=`computeGeneralizedVoronoiEdgesAndVertices`$(\overline{R})$
4: $\mathcal{E} =$ `trimInfeasibleEdges`$(\mathcal{E}, \overline{R})$
5: $\mathcal{V}_{opt}, \mathcal{E}_{opt} =$ `shortestPathThroughGraph`$(\mathcal{V}, \mathcal{E}, \boldsymbol{x}_{h_0}, \boldsymbol{x}_{h_f})$
6: $\overline{C}_0 =$ `fitSplineToPath`$(\mathcal{V}_{opt}, \mathcal{E}_{opt}, N_c, p)$
7: $t_f =$ `checkVelocityConstraint`$(\overline{C}_0, v_{lb}, v_{ub})$
8: $\overline{C}_{opt}, t_f =$ `optimizeTrajectory`$(\overline{C}_0, t_f, \boldsymbol{x}_{h_0}, \boldsymbol{x}_{h_f}, \overline{R})$
9: $P_{max} = \texttt{max}_{t_s \in \boldsymbol{t}_s}(\Gamma_e(\boldsymbol{p}(t_s), \overline{X}_{e,t}))$
10: **if** $P_{max} \leq P_t$ **then**
11: Dispatch high-proirty agent
12: **end if**

Algorithm 4 outlines our method. The input of the algorithm is the current estimate of the radar parameters and the covariances $\overline{R}_t$, the current combined agent path history list $\overline{X}_{e,t}$, at time t, and the start $\boldsymbol{x}_{h_0}$ and destination $\boldsymbol{x}_{h_f}$ locations of the high priority agent.

The first step in the algorithm is to find the generalized Voronoi edges and vertices using $P(P_D \leq P_{D,t})$ as the metric. This is accomplished using the grid-based algorithm shown in Algorithm 5. We begin by generating an evenly spaced set of

Algorithm 5 Grid Based Generalized Voronoi Diagram

1: **Inputs:** $\overline{R}$
2: **Output:** $\mathcal{E}, \mathcal{V}$
3: $\overline{X}_t =$ `evenlySpaceSampeles`$(\mathcal{D})$
4: $\overline{A} = \{\texttt{argmin}_j(P(P_D(\boldsymbol{x}_{t,i}, \theta_{u,j}, \theta_{e,j}) \leq P_{D,t})\} \forall \boldsymbol{x}_{t,i} \in \overline{X}_t$
5: $\overline{N} =$ `findAdjecentCellsUsingVoronoi`$(\overline{R})$
6: $\mathcal{E}, \mathcal{V} =$ `findGeneralizedVoronoiEdges`$(\overline{N}, \overline{A})$

test points in the region $\overline{X}_t$. At each test point, we evaluate the likelihood that the true PD falls below the threshold for each radar. Each point is then assigned to the Voronoi cell corresponding to the radar that minimizes this likelihood, as shown in line 4.

This process provides the generalized Voronoi cells; however, we need to find the edges and vertices between these cells. To find the edges we look at the contours around each cell. For each pair of neighboring cells, we find the portion of each cell's contours that overlap. This overlapped portion of the cell boundary boundaries is the generalized Voronoi edge between the cells. We then fit a third-order B-spline to this boundary, resulting in each edge being defined using control points, knot points $e_{i,j} = (\overline{C}_{ij}, \boldsymbol{t}_{i,j})$. The Voronoi vertices are the points where the generalized Voronoi edges intersect.

After finding the generalized Voronoi edges and vertices, our Algorithm 4 proceeds similarly to Algorithm 3. Infeasible

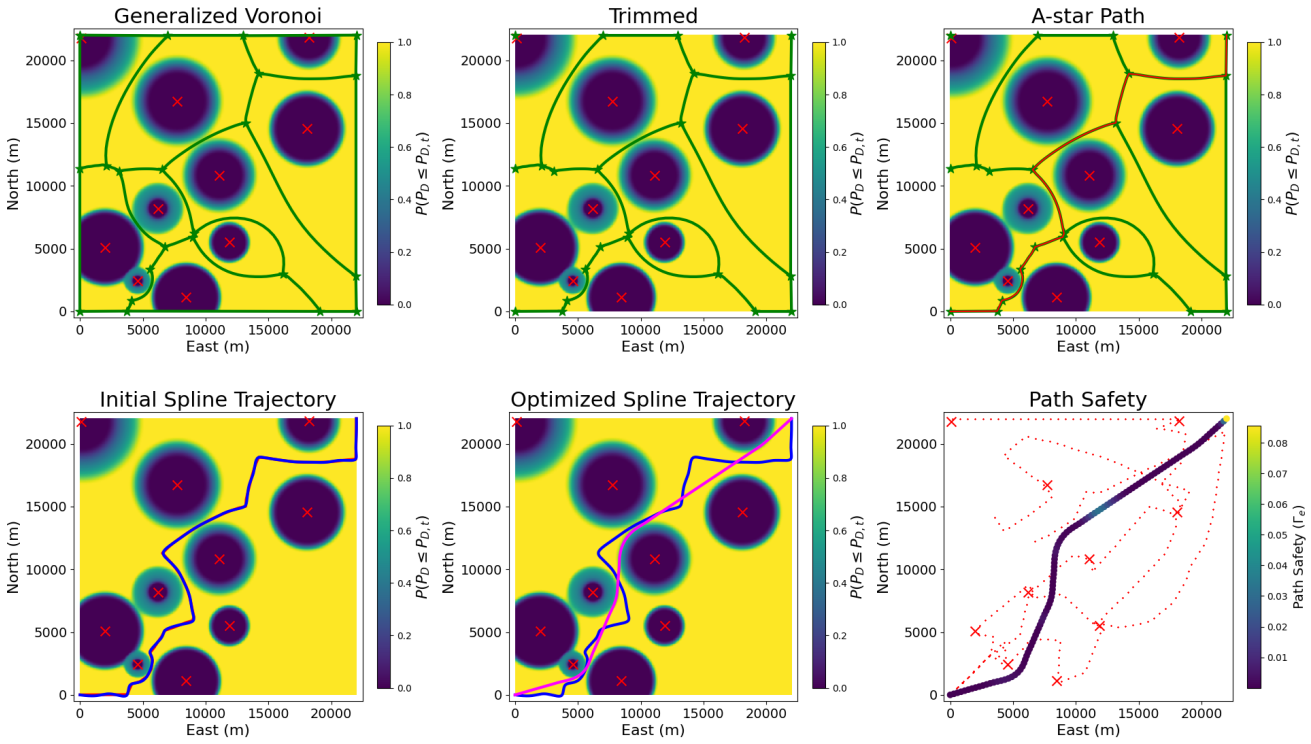

Fig. 3. This figure shows the steps of our uncertain high-priority path planning algorithm (Algorithm 4). In the top left figure, the generalized Voronoi diagram is found using $P(P_D \leq P_{D,t})$ as the metric. Then all edges where $P(P_D \leq P_{D,t}) < \epsilon$ are removed. In the top right figure, A-star is used to find the shortest path through the graph. In the bottom left, a B-spline is fit to the A-star path. Then, in the bottom center, the optimized B-spline is shown. Finally, the bottom left image shows the path safety metric. In this figure, the low-priority agents' paths are shown in red. All figures show the current estimated location of the radar as red x's. The first five show $P(P_D \leq P_{D,t})$ as the heatmap.

edges are trimmed in Line 4. Because there is no closed-form solution to the location of the lowest $P(P_D \leq P_{D,t})$, we sample the edge and check many different locations. If the probability that the PD is below the PD threshold is below ϵ for any of the sampled points on the edge, that edge is removed. After infeasible edges are removed, we use the A-Star algorithm to find the shortest path through the graph created using the generalized Voronoi vertices and trimmed edges. We then sample the path and fit a B-spline to the path using N_c control points and internal knot points defined on $[0, 1]$. We use the same huerisitc method described for the deterministic path planner to ensure the velocity constraint is met. After a feasible initial trajectory is found we use IPOPT to refine the trajectory according to the following optimization problem:

$$\overline{C}_{opt} = \underset{\overline{C}, t_f}{\operatorname{argmin}}\, t_f \tag{48a}$$

$$\text{subject to } \boldsymbol{p}(0) = \boldsymbol{x}_{h_0} \tag{48b}$$

$$\boldsymbol{p}(t_f) = \boldsymbol{x}_{h_f} \tag{48c}$$

$$\boldsymbol{p}(\boldsymbol{t}_s) \in \mathcal{D} \tag{48d}$$

$$v_{lb} \leq v(\boldsymbol{t}_s) \leq v_{ub} \tag{48e}$$

$$u_{lb} \leq u_A(\boldsymbol{t}_s) \leq u_{ub} \tag{48f}$$

$$-\kappa_{ub} \leq \kappa_A(\boldsymbol{t}_s) \leq \kappa_{ub} \tag{48g}$$

$$P(P_D(\boldsymbol{p}(\boldsymbol{t}_s), \overline{\theta}_u, \overline{\theta}_e) \leq P_{D,t}) \geq \epsilon, \tag{48h}$$

This optimization problem is the same as the deterministic case (Equation (47)) except for the final constraint (48h). Instead of using the ground-truth PD evaluated at points along the trajectory, we use the approximate probability that the ground-truth PD is greater than ϵ. This ensures the probability of the path being safe ($P_D \leq P_{D,t}$) is greater than ϵ accounting for the uncertainty from the estimates and prior beliefs.

The final step of the algorithm is to test the path to see if it is likely that there are undiscovered radar stations near the path. So far, the algorithm has only accounted for discovered radar, and it has no sense of the danger from undiscovered radar. Using Equation (38) we find the probability that an undiscovered radar is on the optimized trajectory. If this probability is greater than a threshold at any point along the trajectory, we determine that the path is not safe enough for the high-priority agent to traverse. If the probability is lower than the threshold at all points on the trajectory, the path is determined to be safe, and the high-priority agent is dispatched. Each step of your algorithm is outlined in Figure 3.

VIII. Results

In this section, we present simulation results for our low-priority and high-priority path planning algorithms. We first present results for the low-priority path planner, showing how it performs with various parameter values and how it compares to a baseline, "lawnmower" path planner. We next show results for the high-priority path planner. To test the algorithm in varying situations, we randomly placed 13 different radars in the region with random parameters. We used 50 different

random configurations of the radar. The parameters we used to randomly generate are shown in Table I. For each random configuration, we sample the radar parameters from a uniform distribution defined by the range provided in the table. For example, the output power is sampled from a uniform distribution from $P_{t,l}$ to $P_{t,u}$. All other parameters used in the simulation experiments are shown in Table I.

Parameter	Symbol	Algorithm	Value
Operational Region Bounds	-	-	$(22000, 22000)$
Radar Output Power Upper	$P_{T,u}$	-	20000 Watts
Radar Output Power Lower	$P_{T,l}$	-	0 Watts
Radar Transmit Gain Upper	$G_{T,u}$	-	20dB
Radar Transmit Gain Lower	$G_{T,l}$	-	0dB
Radar Receive Gain	G_R	-	10dB
Path Loss	L	-	0dB
Radar Wavelength	λ	-	99.9 millimeters
Radar Pulse Width	τ_p	-	0.000011 seconds
Radar System Temperature	T_s	-	745 Kelvin
Radar Probability of False Alarm	P_{fa}	-	745 Kelvin
Agent Intercept Antenna Gain	G_I	-	1dB
Agent Radar Cross Section	G_I	-	0.1 meters squared
Angle of Arrival Measurement Noise Variance	σ^2_ϕ	-	2 Degrees
Received Power Measurement Noise Variance	$\sigma^2_{S_E}$	-	0.1 milliwatts?
Probability of Intercept Loss	δ	Low Priority	1000
Covariance Objective Function Multiple	d_{cov}	Low Priority	$1e14$
Re-plan Horizon	T_h	Low Priority	20 Seconds
Agent Path History Period	Δ_{t_e}	Low Priority	5 Seconds
Probability of Detection Threshold	$P_{D,t}$	High Priority	15%
B-spline Degree	ρ	High Priority	3
Number of Control Points	N_c	High Priority	40
Start Location	$\boldsymbol{x}_{h_0}$	High Priority	$(0, 0)$
Goal Location	$\boldsymbol{x}_{h_f}$	High Priority	$(22000, 22000)$
Velocity Bounds	v_{lb}, v_{ub}	High Priority	$100, 134$ m/s
Turn Rate Bounds	u_{lb}, u_{ub}	High Priority	$-5, 5$ rad/s
Maximum Curvature	κ_{ub}	High Priority	0.1 rad/m
Probability of Detection Probability Threshold	ϵ	High Priority	0.9

TABLE I
SIMULATION PARAMETERS

We first show how varying the parameter of adjustment of the low priority path planning algorithm $(\alpha_e, \alpha_u, \alpha_s)$ affects performance. We constrain the sum of the tuning parameters to be one $\alpha_e + \alpha_u + \alpha_s = 1$. We then vary the value of each tuning parameter with this constraint and test the algorithms performance on the 50 random radar configurations. The results of these tests are shown in Figures 4 and 5. The plots use a ternary plot to show the effects of the three parameters. Each side of the triangle represents the value of one parameter. The lines in the plot show the grid where that parameter is the same. Figure 4 shows the percentage of the random runs where the low-priority agents were able to find a path for the high-priority agent. As can be seen in the figure, the parameters need to have a balance. Each coroner of the triangle represents when all but one parameters is zero. None of these parameters perform well. From the figure, the distance from the target weight should be lower, with a value of $\alpha_s = 0.083$ showing the best performance. There also needs to be a balance between exploration and covariance reduction with the best performance occurring at $\alpha_u = 0.667$ and $\alpha_e = 0.25$. In this case, the agents were able to find the path for the high-priority agent in 94% of the random runs. Figure 5 shows the average time it took for low-priority agents to find a path for the high-priority agent with varying parameter values. This only includes the successful runs (because in the runs that were not successful, the agents never found a path for the high-priority agent). It can be seen that increasing the distance to the goal weight α_s, decreases the average time to find the path, however, in these cases a smaller percentage of runs were successful.

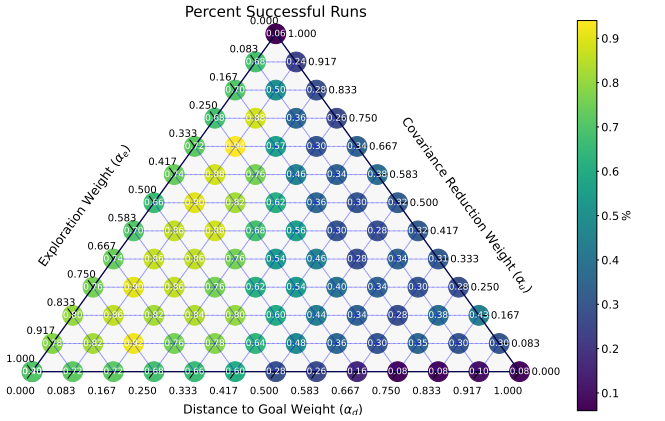

Fig. 4. This figure shows the percent of random runs where the low-priority agents were able to find the path for the high-priority agents with varying values of the exploration weight α_e, the covariance reduction weight α_u and the distance to goal weight α_s. The plot is a ternary plot, where each parameter is represented by one side of the triangle. The color of the points corresponds to the percent of runs that were successful.

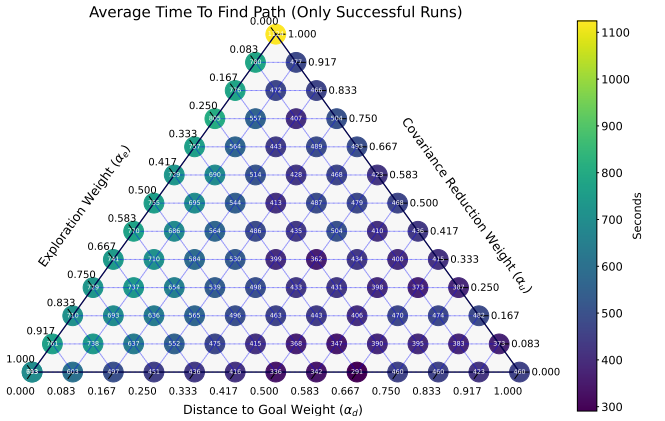

Fig. 5. This figure shows the average time it took for the low-priority agents to find a path for the high-priority agent with varying levels of parameters. The average time only includes runs that were successful (see Figure 4 for how many runs were successful. The color of the points shows the average time it took for the agents to find the path.

We next show a comparison of our low priority path planning algorithm with a baseline, "lawnmower" path planner. The "lawnmower" path planner divides the region equally between all agents and creates a sweeping pattern back and forth for each agent in their section. We choose the "rung" size of the pattern using the exploration objective function 38, by picking a threshold of how likely an undiscovered radar would lie in between rungs and solving for a distance. We also allowed the "lawnmower" path planner to travel back down through the region, splitting the original rungs in half. The agents were limited to running for 1200 seconds for both our algorithm and the "lawnmower" algorithm. Using our path planner, the low-priority agents were able to find the high-priority path in 94% of the random runs. The "lawnmower" low-priority agents were able to find the high priority path in 86% of the runs. The timing results are shown in Figure 6. As can be seen in the figure, the "lawnmower" path planner finds the path in roughly the same amount of time each run. This is

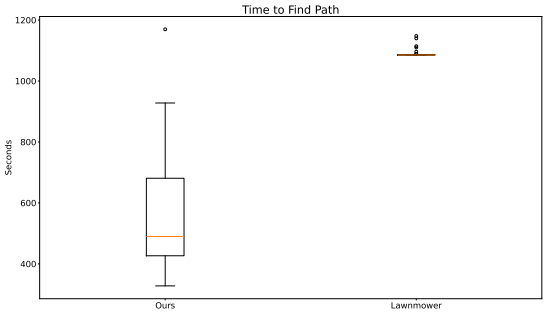

Fig. 6. This figure shows a box plot distribution of the time it took the low-proirty agents to find the high priority path between the baseline "lawnmower" path planner and our algorithm using the best parameters: $\alpha_s = 0.083$, $\alpha_u = 0.667$, $\alpha_e = 0.25$..

determined by how long it takes the agents to complete their coverage paths. Our algorithm's times are more distributed because how its performance depends on the radar layout. Our algorithm is able to find the high-priority path faster in all cases but one than the "lawnmower" baseline.

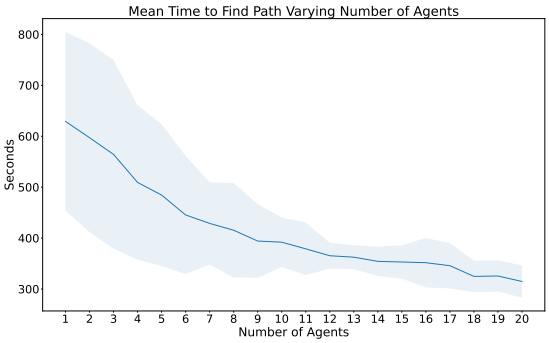

Fig. 7. This figures shows the average time it took for the low-proirty agents to find a safe path for the high-priority agent using the best parameters from the previous test: $\alpha_s = 0.083$, $\alpha_u = 0.667$, $\alpha_e = 0.25$. The time decreases with increasing number of agents because there are more agents to rapidly explore the environment. One standard deviation above and below the mean time are shaded in the figure.

Figure 7 shows how our algorithm performs with varying numbers of agents. The plot shows the mean time it took the low-priority agents to find the high-priority path with varying numbers of agents. The same 50 random scenario were used in this test. The best parameters from the above tests were used: $\alpha_s = 0.083$, $\alpha_u = 0.667$, $\alpha_e = 0.25$. As expected, with increasing numbers of low-priority agents, the time it takes them to find the high-priority path decreases because there are more agents to explore the environment.

To show how well our high-priority path planning algorithm performs we provide statistics from the run with 20 low-priority agents and parameters $\alpha_s = 0.083$, $\alpha_u = 0.667$, $\alpha_e = 0.25$ where the agents were able to find a safe path in all 50 cases. To show how the path planner performs we first report the average maximum ground truth PD accounted along the trajectories. We evaluate the ground truth PD along each of the 50 planned paths and find the maximum, then compute the average, which is 11.59%. This is below the 15% threshold used in the algorithm. The maximum ground truth PD was above the 15% threshold in 3 cases, meaning the success rate of the probabilistic threshold was 94%. We used a 90% threshold so we would expect the probabilistic threshold to work in about 90% of the cases if the linearization of the PD was accurate.

I am considering adding a couple figures here where I vary the PD threshold and the confidence threshold.

IX. Conclusion

In this paper, we presented three path planning algorithms for operating in contested enemy environments using the paradigm of low-cost scouting agents and a high-priority agent with a specific mission. The enemy environment was characterized by enemy radar, and we modeled the probability of detection from those radar. We assumed low-priority agents could intercept enemy radar signals and estimate parameters of enemy radar from the intercepted signals. The low-priority agents were tasked with exploring the environment to find a safe path for the high-priority agent. We ed a decentralized path planning algorithm for the low-priority agents that planned paths to discover undiscovered enemy radar stations and to measurement locations to reduce the uncertainty in the estimate of the discovered radar. We provided simulation results to compare our method with a "lawnmower" baseline and show improved performance. We also presented two algorithms for the high-priority agent, one for the deterministic case, where all radar parameters are known, and the other for when the radar parameters are uncertain. Both rely on using a Voronoi diagram and the A-star algorithm to find an initial feasible trajectory through enemy radar, and then refine that trajectory using an interior point optimization algorithm, accounting for kinematic feasibility and PD constraints. Future work includes accounting for agent RCS that varies with the view angle of the vehicle (in this work we assume RCS was the same from all angles). This could be incorporated into our algorithms by using the maximum RCS of the vehicle when selecting the initial trajectory, then allowing the optimization algorithm to vary the view angle of the vehicle using the true RCS to improve performance.

References

[1] S. Ponda, R. Kolacinski, and E. Frazzoli, "Trajectory optimization for target localization using small unmanned aerial vehicles," in *AIAA guidance, navigation, and control conference*, p. 6015, 2009.

[2] S. Xu, B. Zhu, X. Li, X. Wu, and T. Xu, "Systematical sensor path optimization solutions for aoa target localization accuracy improvement with theoretical analysis," *IEEE Transactions on Vehicular Technology*, 2024.

[3] S. Xu, "Optimal sensor placement for target localization using hybrid rss, aoa and toa measurements," *IEEE Communications Letters*, vol. 24, no. 9, pp. 1966–1970, 2020.

[4] N. Sahu, L. Wu, P. Babu, B. S. MR, and B. Ottersten, "Optimal sensor placement for source localization: A unified admm approach," *IEEE Transactions on Vehicular Technology*, vol. 71, no. 4, pp. 4359–4372, 2022.

[5] S. A. A. Shahidian and H. Soltanizadeh, "Single-and multi-uav trajectory control in rf source localization," *Arabian Journal for Science and Engineering*, vol. 42, no. 2, pp. 459–466, 2017.

[6] S. A. A. Shahidian and H. Soltanizadeh, "Autonomous trajectory control for limited number of aerial platforms in rf source localization," in *2015 3rd RSI International Conference on Robotics and Mechatronics (ICROM)*, pp. 755–760, IEEE, 2015.

[7] K. Dogancay, "Uav path planning for passive emitter localization," *IEEE Transactions on Aerospace and Electronic systems*, vol. 48, no. 2, pp. 1150–1166, 2012.

[8] R. Hameed, A. Maqsood, A. Hashmi, M. Saeed, and R. Riaz, "Reinforcement learning-based radar-evasive path planning: A comparative analysis," *The Aeronautical Journal*, vol. 126, no. 1297, pp. 547–564, 2022.

[9] W. Li, L. Cheng, and J. Hu, "Research on stealthy uav path planning based on improved genetic algorithm," in *2022 International Conference on Artificial Intelligence and Computer Information Technology (AICIT)*, pp. 1–6, IEEE, 2022.

[10] Z. Zhang, J. Wu, J. Dai, and C. He, "Rapid penetration path planning method for stealth uav in complex environment with bb threats," *International Journal of Aerospace Engineering*, vol. 2020, no. 1, p. 8896357, 2020.

[11] B. Basbous, "2d uav path planning with radar threatening areas using simulated annealing algorithm for event detection," in *2018 international conference on artificial intelligence and data processing (IDAP)*, pp. 1–7, IEEE, 2018.

[12] Z. Zhao, Y. Niu, Z. Ma, and X. Ji, "A fast stealth trajectory planning algorithm for stealth uav to fly in multi-radar network," in *2016 IEEE International Conference on Real-time Computing and Robotics (RCAR)*, pp. 549–554, IEEE, 2016.

[13] C. Gao, H. Gong, Z. Zhen, Q. Zhao, and Y. Sun, "Three dimensions formation flight path planning under radar threatening environment," in *Proceedings of the 33rd Chinese Control Conference*, pp. 1121–1125, IEEE, 2014.

[14] P. T. Kabamba, S. M. Meerkov, and F. H. Zeitz III, "Optimal path planning for unmanned combat aerial vehicles to defeat radar tracking," *Journal of Guidance, Control, and Dynamics*, vol. 29, no. 2, pp. 279–288, 2006.

[15] M. Pelosi, C. Kopp, and M. Brown, "Range-limited uav trajectory using terrain masking under radar detection risk," *Applied Artificial Intelligence*, vol. 26, no. 8, pp. 743–759, 2012.

[16] M. Zabarankin, S. Uryasev, and R. Murphey, "Aircraft routing under the risk of detection," *Naval Research Logistics (NRL)*, vol. 53, no. 8, pp. 728–747, 2006.

[17] A. Costley, G. Droge, R. Christensen, R. C. Leishman, and J. Swedeen, "Path planning with uncertainty for aircraft under threat of detection from ground-based radar," *arXiv preprint arXiv:2207.03716*, 2022.

[18] M. R. H. Al-Dahhan and K. W. Schmidt, "Voronoi boundary visibility for efficient path planning," *IEEE Access*, vol. 8, pp. 134764–134781, 2020.

[19] K. Judd and T. McLain, "Spline based path planning for unmanned air vehicles," in *AIAA Guidance, Navigation, and Control Conference and Exhibit*, p. 4238, 2001.

[20] Z. Liu, L. Gao, F. Liu, D. Liu, and W. Han, "Fusion of weighted voronoi diagram and A* algorithm for mobile robot path planning," in *2022 2nd International Conference on Electrical Engineering and Mechatronics Technology (ICEEMT)*, pp. 403–406, 2022.

[21] J.-w. Choi, R. Curry, and G. Elkaim, "Real-time obstacle-avoiding path planning for mobile robots," in *AIAA Guidance, Navigation, and Control Conference*, p. 8411, 2010.

[22] C. Zhang, H. Liu, and Y. Tang, "Quantitative evaluation of voronoi graph search algorithm in uav path planning," in *2018 IEEE 9th International Conference on Software Engineering and Service Science (ICSESS)*, pp. 563–567, IEEE, 2018.

[23] X. Chen, X. Chen, and G. Xu, "The path planning algorithm studying about uav attacks multiple moving targets based on voronoi diagram," *International Journal of Control and Automation*, vol. 9, no. 1, pp. 281–292, 2016.

[24] P. Chen, L. Xiaoqing, D. Jiyang, and Y. Linfei, "Research of path planning method based on the improved voronoi diagram," in *2013 25th Chinese Control and Decision Conference (CCDC)*, pp. 2940–2944, IEEE, 2013.

[25] E. Tzoreff and A. J. Weiss, "Path design for best emitter location using two mobile sensors," *IEEE Transactions on Signal Processing*, vol. 65, no. 19, pp. 5249–5261, 2017.

[26] A. Costley, R. Christensen, R. C. Leishman, and G. N. Droge, "Sensitivity of single-pulse radar detection to aircraft pose uncertainties," *IEEE Transactions on Aerospace and Electronic Systems*, vol. 59, no. 3, pp. 2286–2295, 2022.

[27] A. Costley, R. Christensen, R. C. Leishman, and G. Droge, "Sensitivity of the probability of radar detection to radar state uncertainty," *IEEE Transactions on Aerospace and Electronic Systems*, 2023.

[28] H. Li, W. Jing, and Y. Bai, "Radar emitter recognition based on deep learning architecture," in *2016 CIE International Conference on Radar (RADAR)*, pp. 1–5, 2016.

[29] B. Boots, A. Okabe, and K. Sugihara, "Spatial tessellations," *Geographical information systems*, vol. 1, pp. 503–526, 1999.

[30] M. Held and S. de Lorenzo, "An efficient, practical algorithm and implementation for computing multiplicatively weighted voronoi diagrams," *arXiv preprint arXiv:2006.14298*, 2020.

[31] M. G. Cox, "The numerical evaluation of b-splines," *IMA Journal of Applied mathematics*, vol. 10, no. 2, pp. 134–149, 1972.

[32] D. Buccieri, D. Perritaz, P. Mullhaupt, Z.-P. Jiang, and D. Bonvin, "Velocity-scheduling control for a unicycle mobile robot: Theory and experiments," *IEEE Transactions on Robotics*, vol. 25, no. 2, pp. 451–458, 2009.

[33] J. Bradbury, R. Frostig, P. Hawkins, M. J. Johnson, C. Leary, D. Maclaurin, G. Necula, A. Paszke, J. VanderPlas, S. Wanderman-Milne, and Q. Zhang, "JAX: composable transformations of Python+NumPy programs," 2018.

[34] A. Wächter and L. T. Biegler, "On the implementation of an interior-point filter line-search algorithm for large-scale nonlinear programming," *Mathematical programming*, vol. 106, no. 1, pp. 25–57, 2006.

[35] T. Norton, G. Stagg, D. Ward, and C. K. Peterson, "Decentralized sparse gaussian process regression with event-triggered adaptive inducing points," *Journal of Intelligent & Robotic Systems*, vol. 108, no. 4, p. 72, 2023.

[36] P. E. Hart, N. J. Nilsson, and B. Raphael, "A formal basis for the heuristic determination of minimum cost paths," *IEEE transactions on Systems Science and Cybernetics*, vol. 4, no. 2, pp. 100–107, 1968.