Optimal Sensing and Path Planning using Information Methods for a Multi-Agent Line of Sight Network

Collin Hudson
Aerospace Engineering Sciences
University of Colorado Boulder
collin.hudson@colorado.edu

***Abstract*—This project details a path planning system that seeks to optimally place a set of Uncrewed Aircraft Systems (UAS) to form a network that maintains line of sight of an uncertain target position with respect to information gathered about the target. In order to plan for the network, the target state is estimated using a particle filter, based on range-only measurements from each UAS. The estimate is then utilized within a mixed-integer nonlinear program to maximize the Fisher information of the target and satisfy problem constraints. Solutions to the problem were generated using the Bonmin optimizer and demonstrated via simulation using the Julia programming language. In addition, the effectiveness of multiple planning strategies for using Fisher information to better search the environment was evaluated.**

I. Introduction

This central problem at the heart of this project is the limitations faced by communication systems that rely on line of sight (LOS) when operating in obstructed environments. When necessary to increase the range of a radio signal, one approach is to chain the signal through radio repeaters to the receiver. However, manually placed repeaters can't easily adapt to changing environments or mission requirements and necessitate additional personnel to set them up. Instead, the repeaters could be fixed to UAS to form a network that enables communication. In addition, the problem of maintaining line of sight with some actor or agent in an environment has a variety of applications that the UAS network could help address. For this project, it is assumed that there is a fixed ground station that all UAS must have a path to connect to through the network, and another agent that the network wants to maintain line of sight of, referred to generally in this project as the target. For clarity, the ground station and target are collectively referred to as ground agents, and all other agents are referred to as UAS agents.

There are a variety of cases, such as in GPS-denied environments, where the actual position of the target has some uncertainty or is entirely unknown, and must be estimated. Since the underlying problem is that of a communication network, it is reasonable to assume that the signal propagation time between agents connected together could be recorded as a measure of the distance between them, similar to a pseudorange calculation. Assuming that these measurements could be communicated through the network to the planner, an estimator could be utilized to determine the position of the target based on the known position of all agents in the network. As a result, the estimator and motion planner work together to both infer the position of the ground agent based on the pseudorange measurements and plan for the position of the UAS network to take the aforementioned measurements.

The principal uncertainty to consider is that of the target position at the next time step so the motion planner can provide positions for each of the sensing UAS agents to take the next measurement. In order to maintain a reasonable scope for the project, the target is assumed to have a constant but unknown velocity, reducing the future state uncertainty to the position of the target at the current time step and its velocity. Since the UAS agents can only provide range measurements, probability distributions of the target position are highly non-Gaussian. For example, if one UAS agent receives a range measurement of the target, the resulting position probability distribution would be a noisy ring around the agent, which would be poorly represented by a single normal distribution.

While this complexity could possibly be addressed using a Gaussian mixture, the distributions are also subject to several point-based constraints such as line of sight with the sensing agent, collision avoidance with obstacles, and an assumed maximum communication radius. As a result, distributions over areas could overlap with obstacles or in areas outside the view of the UAS agent, leading to poor target position estimates. Thus, the estimation portion of the problem naturally lends itself to approximate inference techniques, and intuitively the particle filter, which can immediately handle the point-based constraints simply based on the choice of particle reweighting function.

In order to gain an estimate of the target, measurements must be taken by the UAS agents in the network. The quality of these measurements is dependent on the position of the agent relative to the target, as the measurements are assumed to be range-limited since they come from a signal being successfully received. Thus, the concept of information can be used to quantify the value of a network configuration for estimating the target position.

This concept has been explored using a gradient-based approach to find paths for a distributed team of robots in [1], where the trace of the inverse of the Fisher information matrix was minimized. This drives the minimum covariance of an unbiased estimator, called the Cramér-Rao Lower Bound,

towards zero. A similar problem was explored in [2] for a distributed hierarchical planner for multi-hop network communication that considered the probability of packet loss during communication multiplied by the Fisher information matrix as a cost function, with the determinant of the resulting cost matrix serving as the objective function. Particle filters have also been leveraged to estimate the predicted conditional entropy in a target distribution, which is related to the concept of information, and choose an optimal trajectory from a discrete set of trajectories for a single aircraft in [3]. In addition, they have been used in [4] for a distributed UAS network to estimate the mutual information when track a target using range-only measurements, which enabled an information-based control scheme. However, their solution did not take into account communication range or obstructed line of sight limitations.

II. Target Estimation

The bootstrap particle filter was implemented using the Julia package `ParticleFilters.jl` [5], which provides a simple framework to define the prediction and reweighting steps of a particle filter, while handling the resampling and normalization steps internally. The measurement model was defined with the package `Distributions.jl`, which supports sampling from truncated Gaussian distributions. Let i represent the particle index and k correspond to the time step under consideration.

A. Bootstrap particle filter

The bootstrap particle filter is a straightforward implementation of the particle filter which uses the transition probability $P(\mathbf{x}_{k+1}|\mathbf{x}_k)$ as the importance sampling function $q(\mathbf{x}^i_{k+1};\mathbf{x}^i_k,\mathbf{y}^i_{k+1})$. This results in a particle weight update prior to resampling in the following form.

$$\begin{aligned} w^i_{k+1} &\propto \frac{P(\mathbf{y}_{k+1}|\mathbf{x}^i_{k+1})P(\mathbf{x}^i_{k+1}|\mathbf{x}^i_k)}{q(\mathbf{x}^i_{k+1};\mathbf{x}^i_k,\mathbf{y}_{k+1})}w^i_k \\ &\propto P(\mathbf{y}_{k+1}|\mathbf{x}^i_{k+1})w^i_k \end{aligned}$$

In addition, the particle set is resampled after every measurement update, with weights normalized following the resampling step.

The implementation using ParticleFilters.jl is also fairly straightforward using the `BootstrapFilter()` function, which requires as input a particle prediction function and a particle reweighting function. The resampling and weight normalization steps are implemented within the filter object.

B. Measurement and Dynamics Model

Consider UAS agent j at position $\mathbf{x}_{a_j,k}$ and the target at position $\mathbf{x}_{g_k}$. The measurement model is given below, with measurement noise $v_{j,k} \sim \mathcal{N}(0, R_j)$ and distance between agents $r_{j,k} = \left\|\mathbf{x}_{g_k} - \mathbf{x}_{a_j,k}\right\|_2$. The model is intended to represent an agent receiving a signal from the target and calculating the distance between them from the pseudorange, with the noise v_{j_k} encapsulating any errors between the pseudorange and true range.

$$y_{j,k} = \begin{cases} r_{j,k} + v_{j,k} & \text{if } \ r_{j,k} \le r_{\text{con}} \text{ and in LOS} \\ \text{inf} & \text{o.w.} \end{cases}$$

Because the measurement noise is assumed to be AWGN, the probability of receiving measurement $y_{i,k}$ given the state $\mathbf{x}_k$ could be modeled using a truncated Gaussian distribution with bounds at 0 and the maximum connection radius r_{con}. This will prove to be useful, as methods for sampling from truncated Gaussian distributions are well-known and readily available. Let Φ represent the CDF of a normal distribution. The measurement noise variance for each agent is assumed to be equal based on the assumption that all UAS agents are identical, and set to 0.01 based on simulation performance, $R_j = 0.01 \ \forall j \in \{1, ..., n_{uas}\}$.

The maximum connection radius r_{con} is also an arbitrary parameter that merely scales how much of the environment each agent can search for the target in a single time step. As a rule of thumb, the connection radius is limited to one quarter of the width of the environment. Let $\mathbf{x}_k = [\mathbf{x}^T_{g,k},\ \mathbf{x}^T_{a_1,k}, \ldots, \mathbf{x}^T_{a_{n_{uas}},k}]^T$, which is a vector of the target and all agent positions.

$$P(y_{j,k}|\mathbf{x}_k) \sim \frac{\mathcal{N}(r_{j,k},\ R_j)}{\Phi\left(\frac{r_{con}-r_{j,k}}{R_j}\right) - \Phi\left(\frac{-r_{j,k}}{R_j}\right)}\mathbb{I}(0 \le r_{j,k} \le r_{\max})$$

Because the measurements taken by each UAS agent at every time step are assumed to be independent, the joint probability of receiving the measurements from all UAS agents given the positions of all agents at time step k is equal to the product of the probability of receiving each agent's measurement.

$$P(\mathbf{y}_k|\mathbf{x}_k) = \prod_{i=1}^{n_{uas}} P(y_{j,k}|\mathbf{x}_k)$$

The state transitions are deterministic, with the target state following the discrete time-invariant linear system below.

$$\mathbf{x}_{g_{k+1}} = \begin{bmatrix} 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \mathbf{x}_{g_k}$$

The prediction function for the particle filter is simply the particle multiplied by the state transition matrix, plus an artificial process noise vector drawn from a zero mean multivariate Gaussian distribution with small variances. While the real transitions are deterministic, the particle filter would often collapse to a single particle far from the actual target state, so a small amount of process noise was added. This also helps add robustness to the estimation algorithm, as the true target may not have constant velocities.

Algorithm 1 Bootstrap particle filter reweighting function

if x_{k+1}^i is outside bounds or in obstacle **then**
  return 0
else
  $w_{k+1}^i \leftarrow \mathcal{N}(0, R_v)|_{\Delta v}$
  for j = 1:n_{uas} **do**
    $r_{j,k} \leftarrow \left|\left|\mathbf{x}_k^i - \mathbf{x}_{a_{j,k}}\right|\right|_2$
    if $y_{k+1}^j = -1$ (No measurement received) **then**
      $w_{k+1}^i \leftarrow w_{k+1}^i * (r_k^j > r_{con})$
    else if $r_k^j \leq r_{con}$ **then**
      $w_{k+1}^i \leftarrow w_{k+1}^i *$ truncated$(\mathcal{N}(r_{j,k}, R_j))|_{y_{k+1}^j}$
    else
      return 0
    end if
  end for
  return w_{k+1}^i
end if

The particle reweighting function returns a weight for particle i at time step $k+1$. The function takes as input the particle at time step k, and the predicted particle state at time step $k+1$, along with the measurements and UAS agent positions at step $k+1$. Within the particle weighting function, each particle is first weighted by the probability of drawing the difference in the change in position between time steps and the velocity of the particle at the last time step (Δv) from a zero mean Gaussian distribution. This is intended to weight particles according to how well the change in position matches the velocity, which should be constant.

III. Fisher Information Matrix

The measurement model developed in the last section can now be used to define the Fisher information matrix (FIM), which will be used to quantify the quality of a network configuration for an estimated target position. Because the measurements follow a multivariate Gaussian distribution and are independent, the FIM can be represented in the following form. Let $\mathbf{x}_k = [x_1, x_2]$ correspond to the x and y coordinates of the target at time step k, respectively.

$$I_k = \sum_{j=1}^{n_{uas}} H_{j,k}^T R_j^{-1} H_{j,k}$$

Here, $R_j^{-1} = R^{-1} = \frac{1}{0.01}$ according to the measurement noise defined in the previous section. $H_{j,k}$ is the Jacobian of the measurement function for agent j with respect to $\mathbf{x}_k$, as shown by the expression below.

$$\begin{aligned} H_j &= \frac{\partial h(\mathbf{x}, \mathbf{x}_{a_j,k})}{\partial \mathbf{x}_k} = \frac{\partial\ r_{j,k}}{\partial \mathbf{x}_k} \\ &= \left[\frac{\partial}{\partial x_1}\ r_{j,k} \quad \frac{\partial}{\partial x_2}\ r_{j,k}\right] \\ &= \left[\frac{\Delta x_{j,1}}{r_{j,k}} \quad \frac{\Delta x_{j,2}}{r_{j,k}}\right] \end{aligned}$$

Where $\Delta x_{j,1}$ is the difference between the the x coordinates of the target and agent j, and $\Delta x_{j,2}$ is the difference between the y coordinates of the target and agent j. Combining with the expression for the Fisher information matrix,

$$\begin{aligned} I_k &= \frac{1}{R}\sum_{j=1}^{n_{uas}} \begin{bmatrix} \frac{\Delta x_{j,1}}{r_{j,k}} \\ \frac{\Delta x_{j,2}}{r_{j,k}} \end{bmatrix} \begin{bmatrix} \frac{\Delta x_{j,1}}{r_{j,k}} & \frac{\Delta x_{j,2}}{r_{j,k}} \end{bmatrix} \\ &= \frac{1}{R}\sum_{j=1}^{n_{uas}} \begin{bmatrix} \left(\frac{\Delta x_{j,1}}{r_{j,k}}\right)^2 & \frac{\Delta x_{j,1}}{r_{j,k}}\frac{\Delta x_{j,2}}{r_{j,k}} \\ \frac{\Delta x_{j,2}}{r_{j,k}}\frac{\Delta x_{j,1}}{r_{j,k}} & \left(\frac{\Delta x_{j,2}}{r_{j,k}}\right)^2 \end{bmatrix} \end{aligned}$$

Examining the full Fisher information matrix, because the trace of a sum of matrices is equal to the sum of the traces of each matrix, the trace of the Fisher information matrix is as follows.

$$\begin{aligned} \text{tr}(I_k) &= \frac{1}{R}\sum_{j=1}^{n_{uas}} \left(\frac{\Delta x_{j,1}}{r_{j,k}}\right)^2 + \left(\frac{\Delta x_{j,2}}{r_{j,k}}\right)^2 \\ &= \frac{1}{R}\sum_{j=1}^{n_{uas}} \frac{(\Delta x_{j,1})^2 + (\Delta x_{j,2})^2}{(r_{j,k})^2} \\ &= \frac{1}{R}\sum_{j=1}^{n_{uas}} \frac{(r_{j,k})^2}{(r_{j,k})^2} \\ &= \frac{n_{uas}}{R}\ \forall\ k \in \{1, \ldots, k_{end}\} \end{aligned}$$

Regardless of agent or target position, the trace of the Fisher information matrix is a constant.

IV. Optimization Problem Formulation

In the basic form of the planner, the optimization problem to be solved can be summarized as follows: **Given the known ground station and estimated target positions, find valid positions and connections for n_{uas} available UAS agents that define a line of sight network which connects to all ground agents.** The agents must not collide with any obstacle in the environment, which are modeled as convex polygons. The problem is presented as a Mixed-Integer Nonlinear Programming problem at every time step for a finite number of time steps. The state is represented as follows:

TABLE I: Optimization state variables

Variable	Definition
x_i	x coordinate of agent i
y_i	y coordinate of agent i
$c_{i,j}$	Binary variable for connection between agents i and j
$o_{h,i}$	Binary variable for obstacle half-space constraint h for agent i
$l_{h,i,j}$	Binary variable for obstacle half-space constraint h at midpoint between agents i and j
a_i	Binary variable for flow reception
$f_{i,j}$	Flow from agent i to agent j

Let the position of ground agent i at time step k be specified by $g(i,k)$. Let the index sets for ground agents, UAS agents,

obstacles, and obstacle edges be represented by $\mathcal{G}$, $\mathcal{U}$, $\mathcal{O}$, and $\mathcal{O}_e$, respectively. In addition, let the total number of obstacles in the environment equal n_{obs}, and the sum of the number of edges of all obstacles in the environment equal n_e.

A. Connection Constraints

The connection of two agents is limited by some communication radius r_{con} as an analog to real-world limits on signal strength. This constraint is implemented using the Big M method such that the constraint is completely relaxed if two agents are not connected.
Let $M_c = 1.1((x_{max} - x_{min})^2 + (y_{max} - y_{min})^2)$, where the minimum and maximum values are the bounds of the environment. The constraint is defined as follows:

$$||\mathbf{p}_i - \mathbf{p}_j||^2 \leq r_{con}^2 + M_c(1 - c_{i,j}) \quad \forall i, j \in \mathcal{G} \cup \mathcal{U} \tag{1}$$

Thus, if agent i and j are not connected, the inequality becomes trivial. Since x and y are constrained to be within the bounds of the environment, their difference can never exceed M_c. As a result of this constraint, the Mixed-Integer Program is quadratic.

B. Obstacle Avoidance Constraints

The MIP formulation for obstacle avoidance mentioned in [6] was integrated into the problem formulation, which also utilizes Big M relaxation methods. The following constraints are added to the formulation. Let $\{E_j\}$ equal the set of indices of half-spaces corresponding to obstacle j.

$$-\mathbf{h_j}^T \mathbf{p}_i \leq -k_j + M_j(o_{j,i}) \quad \forall i \in \mathcal{U}, j \in \mathcal{O}_e \tag{2}$$

$$\sum_{k \in E_j} o_{k,i} \leq |E_j| - 1 \quad \forall i \in \mathcal{U}, j \in \mathcal{O} \tag{3}$$

Constraint 3 ensures that at least one constraint 2 for obstacle j is enforced, such that agent i is outside of at least one edge of obstacle j.

C. Network Constraints

The following constraints are derived from the formulation introduced in [7], with some modifications to fit the application. The goal of the constraints is to solve for connections between nodes and nonnegative units of units of artificial "flow" in and out of each node in a network, where each node must consume one unit of flow unless it is the designated source node. Since each node needs to consume a unit of flow, each node must be connected to a node with a nonnegative out flow, and therefore must have a path to the source node.

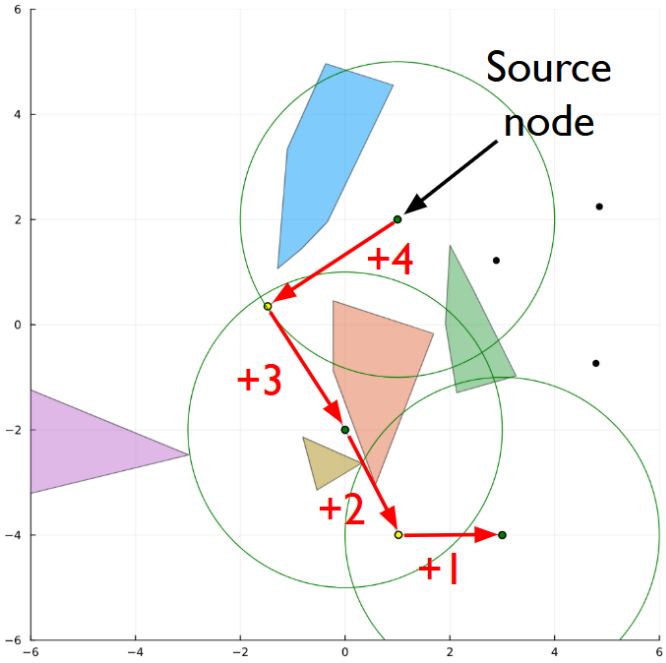

Fig. 1: Network flow example

In the formulation for the Minimum Connectivity Inference (MCI) problem discussed in [7], the subset of nodes, called a cluster, to be connected is known. For the LOS network application, the number of agents in the network is part of the state to be optimized, and is therefore unknown with respect to the MCI problem, requiring modifications to the MCI constraints. Summing over column i of the state variable f will give the total flow into agent i. Similarly, summing over row i gives the total flow out of agent i. For simplicity, the ground station is defined as the source node. The following constraints are added.

$$\sum_j \sum_i c_{i,j} \geq \sum_i (a_i) - 1, \tag{4}$$

$$\sum_j c_{i,j} \leq (n_{ga} + n_{uas}) a_i \quad \forall i \in \mathcal{G} \cup \mathcal{U}, \tag{5}$$

$$\sum_i c_{i,j} \leq (n_{ga} + n_{uas}) a_j \quad \forall j \in \mathcal{G} \cup \mathcal{U}, \tag{6}$$

$$\sum_j f_{i,j} \leq (n_{ga} + n_{uas}) a_i \quad \forall i \in \mathcal{G} \cup \mathcal{U}, \tag{7}$$

$$\sum_j f_{i,j} - \sum_k f_{k,i} = -a_i \quad \forall i \neq 1 \in \mathcal{U}, \tag{8}$$

$$f_{i,j} - f_{j,i} \leq (\sum_i (a_i) - 1) c_{i,j} \quad \forall i < j \in \mathcal{G} \cup \mathcal{U}, \tag{9}$$

$$f_{i,j} \geq 0 \quad \forall i = j \in \mathcal{G} \cup \mathcal{U} \tag{10}$$

Constraint 4 ensures that there are enough connections to connect all agents in the in the network. Constraints 5 and 6 ensure that agents must be in the network to have any connections. Constraint 7 ensures that agents must be in the network to have any flow out of them. Constraint 8 specifies that one unit of flow must be consumed when passing through an agent in the network, unless it is the source node (here the ground station is used with $i = 1$). Constraint 9 requires no net

flow between unconnected agents, and that connected agents have a net flow less than the number of agents in the network. Constraint 10 defines flow as between different agents.

Because the network must act as a sensing network to estimate the target, and it is assumed that the UAS agent measurements are only received by the ground station if the agent is connected to the network, only the target is allowed to be outside of the network. This is crucial due to the fact that the estimated target position may change drastically between time steps when no connections have been made with the target yet. Thus, the following constraint is added.

$$a_i = 1 \quad \forall i \in \{(\mathcal{G} \cup \mathcal{U}) \setminus i_{target}\} \tag{11}$$

Finally, the flow out of the node representing the estimated target position ($i = 2$) should be set to zero, as the target may not actually be there to complete the network.

$$\sum_{k \in \mathcal{U}} f_{k,2} = 0 \tag{12}$$

This constraint ensures that a UAS agent must be connected to another UAS agent or the ground station in order to have a valid network, rather than allowing agents connect to the network through the estimated target position. An alternative approach could be to utilize properties of the Laplacian matrix for a given network graph as demonstrated in [8] to determine network connectivity, but constraint construction with that method seemed too complex for this formulation.

D. LOS Constraints

Constraints from the formulation introduced in [9] were included to ensure line of sight in the network. The nodes are considered in LOS if the midpoint $\mathbf{m_{i,p}}$ of the line between agents i and k belongs to both of the external half-spaces that contain each node. The following constraints are added:

$$-\mathbf{h_j}^T \mathbf{m_{i,p}} \leq -k_j + M_j(o_{j,i}) + M_j(l_{j,i,p,1}) \quad \forall i, p \in \mathcal{G} \cup \mathcal{U}, j \in \mathcal{O}_e \tag{13}$$

$$-\mathbf{h_j}^T \mathbf{m_{i,p}} \leq -k_j + M_j(o_{j,p}) + M_j(l_{j,i,p,1}) \quad \forall i, p \in \mathcal{G} \cup \mathcal{U}, j \in \mathcal{O}_e \tag{14}$$

$$c_{i,p} \leq 1 - l_{j,i,p,1} \quad \forall i, p \in \mathcal{G} \cup \mathcal{U}, j \in \mathcal{O} \tag{15}$$

Constraint 15 ensures that if agents i and p are connected, the LOS constraint must not be relaxed.

E. Time Step Constraints

In order for the solution paths to be viable, UAS agents must be able to reach their specified position at the next time step. A radius constraint is added for each UAS agent, with the radius corresponding to an assumed maximum UAS velocity.

$$||\mathbf{p}_{i,k} - \mathbf{p}_{i,(k-1)}||^2 \leq r_{uas}^2 \quad \forall i \in \mathcal{U}, \forall k > 1 \tag{16}$$

F. Objective Function

The Fisher information matrix quantifies the amount of information about the target at a presumed position that can be gathered by the network configuration denoted as $\mathbf{x}$. Taking the estimated target position as the true target position, an optimal network configuration is calculated such that the Fisher information matrix is maximized. As a result of the measurement model and problem formulation, the trace of the Fisher information matrix is always equal to the number of agents in the environment, and would not be an effective objective function. Instead, the determinant of the Fisher information matrix is used, so the generated network configuration can be described as d-optimal with respect to the Fisher information of the system about the estimated target position.

$$\begin{aligned} &\underset{\mathbf{x}}{\text{Maximize}} \ |I_k| \\ &\text{Subject to:} \\ &\text{Constraints 1-16} \end{aligned}$$

In addition, due to the model assumption that the velocity of the target is constant, the estimate can be propagated forward in time very easily, enabling the solver to optimize the Fisher information matrix for future time steps as well.

$$\begin{aligned} &\underset{\mathbf{x}}{\text{Maximize}} \ |I_k| + |I_{k+1}| + \cdots + |I_{k_{horizon}}| \\ &\text{Subject to:} \\ &\text{Constraints 1-16} \end{aligned}$$

By altering the objective function to sum the determinants of the Fisher information matrices over a time horizon, the generated agent positions should be more optimal in gathering information about the target state. In addition, because the target estimator is a particle filter, the estimate tends to jump around the environment far from its prior, leading to difficulties in planning network movement. By planning forward in time, the objective function could still have some value for the solver to find during optimization when far from the estimate by reasoning about the possible information gained at a future target state.

G. Solver

To solve the optimization problem at each time step, the problem was formulated using the `JuMP.jl` package for the Julia programming language using the open-source mixed-integer nonlinear program solver `Juniper.jl` [10]. The Juniper solver was chosen from a variety of MINLP solvers based on its ability to handle the objective function, and its execution time. Other solvers were either unable to support the objective function, were unable to produce useful or valid solutions, or simply took much longer to solve the same problem.

H. Motion planning and simulation

In order to evaluate both the estimator and optimizer, the following simulation loop was implemented in the Julia programming language.

Algorithm 2 Simulation loop

$b \leftarrow$ uniform belief $\mathcal{U}$
$\mathbf{x}_{est,0} \leftarrow \text{rand}(b)$
for $k = 1 : k_{end}$ **do**
  $\mathbf{x}_{g,k} \leftarrow A\mathbf{x}_{g,k-1}$
  Get agent states from optimizer with target = $A\mathbf{x}_{est,k-1}$
  for $i = 1 : n_{uas}$ **do**
    $r_k^i \leftarrow$ distance between $\mathbf{x}_{g,k}$ and agent i
    $y_k^i \leftarrow \text{rand}\left(\mathcal{N}(r_k^i, R_i)\right)$
    if $y_k^i > r_{con}$ OR $\mathbf{x}_{g,k}$ out of LOS of agent i **then**
      $y_k^i \leftarrow -1$
    end if
  end for
  $b \leftarrow$ filter estimate given prior b, $\mathbf{y}_k$, and agent states $\mathbf{x}_{est,k}$
  $\mathbf{x}_{est,k} \leftarrow \text{mode}(b)$
end for

V. Results

In order to verify that the solutions generated with the Fisher information objective function lead to connected networks, a simple obstacle-free environment was used with four UAS agents over 20 time steps. The communication range of the ground station and estimated target position are shown in green, limited to 2.5 units. The UAS agents are also constrained to move within a radius of 2 units between time steps. Solutions for the empty environment were generated within two minutes, though the time to plot all time steps is also included in that count. A few time steps are shown from the solution using the one-step horizon objective function as a demonstration and to describe the emergent behavior from the agents. For comparison, the same process was run using the same rng seed with an objective function that only seeks to maximize the number of agent connections to the estimated target position.

A. One-step horizon objective function

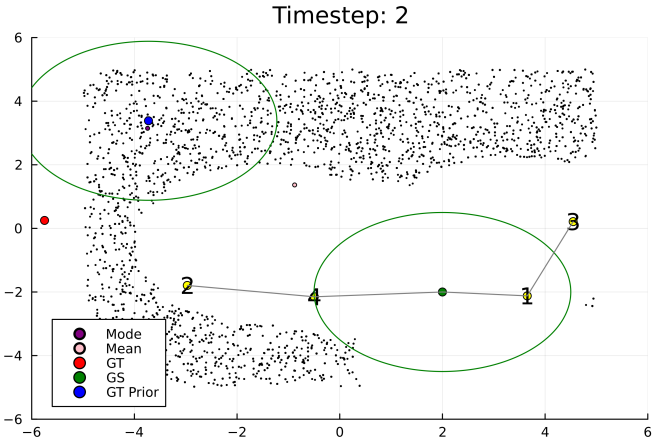

Fig. 2: FIM: Agents spread out to search large area

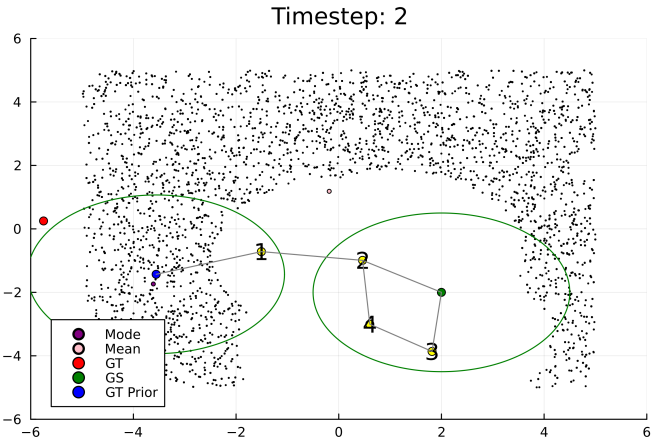

Fig. 3: Connection maximization objective function

After several time steps, the network has swept through the environment leading to a large particle mass in the unsearched area.

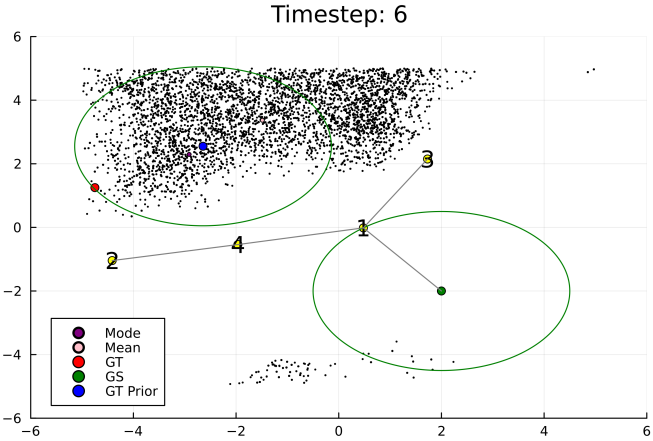

Fig. 4: FIM: Network extends to measure estimated target position.

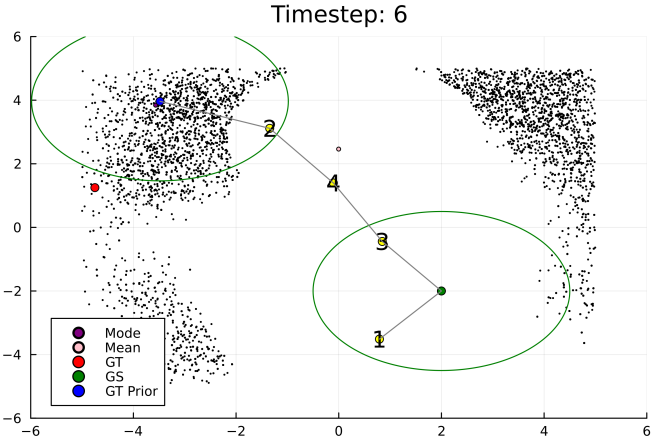

Fig. 5: Connection maximization objective function

At time step 6, UAS agent 2 receives a signal from the actual target, leading to the particles at the next step congregating around its position at step 6.

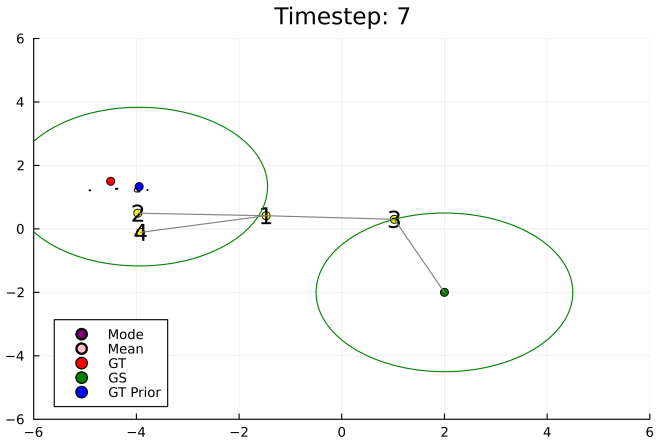

Fig. 6: FIM: Network shifts toward grouped particles

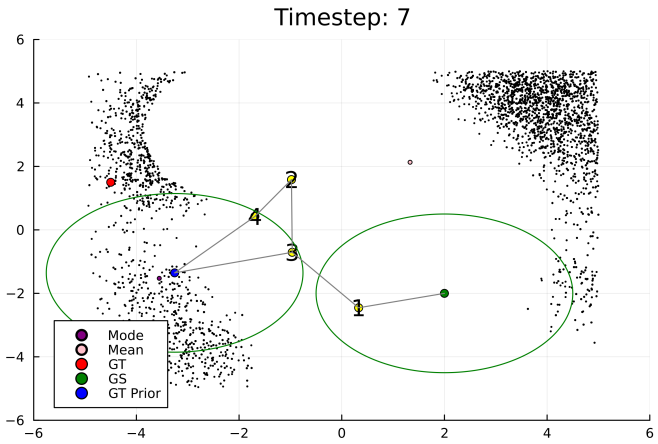

Fig. 7: Connection maximization objective function

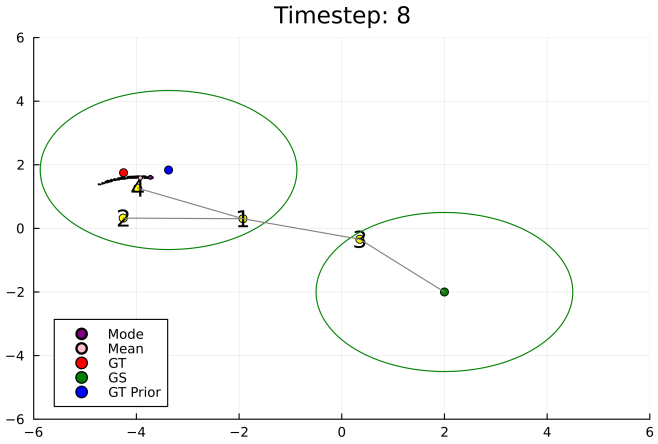

Fig. 8: FIM: Network moving to gather multiple measurements

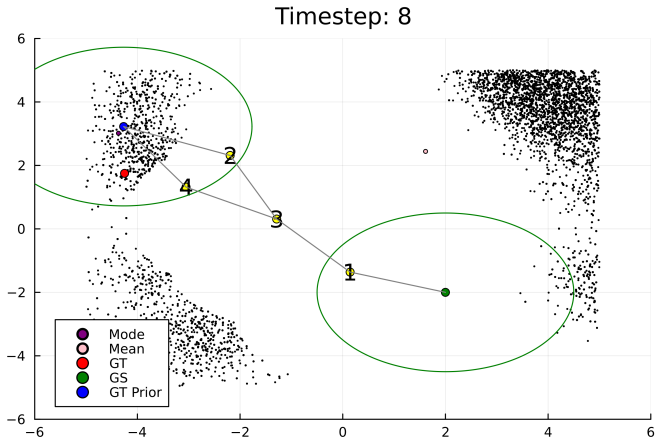

Fig. 9: Connection maximization objective function

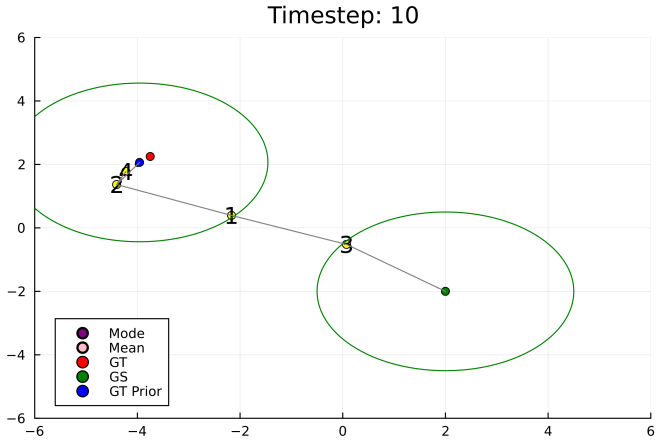

Fig. 10: FIM: Network successfully tracks target

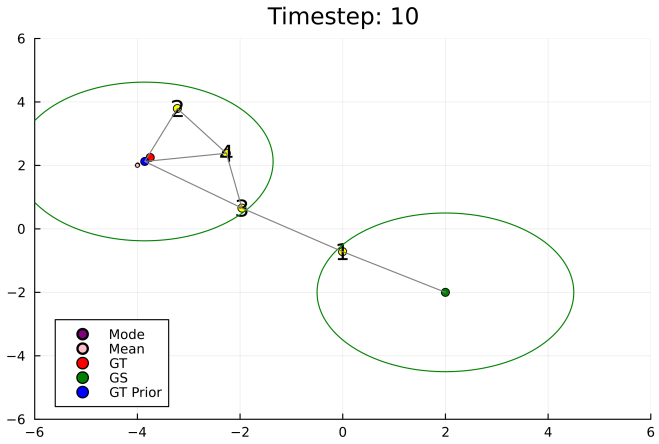

Fig. 11: Connection maximization objective function

B. Three-step horizon

Using the same seeded empty environment as before, the solver was then configured to utilize the multiple horizon objective function to maximize the sum of the determinants of the Fisher information matrix of the estimated target position over the next three steps.

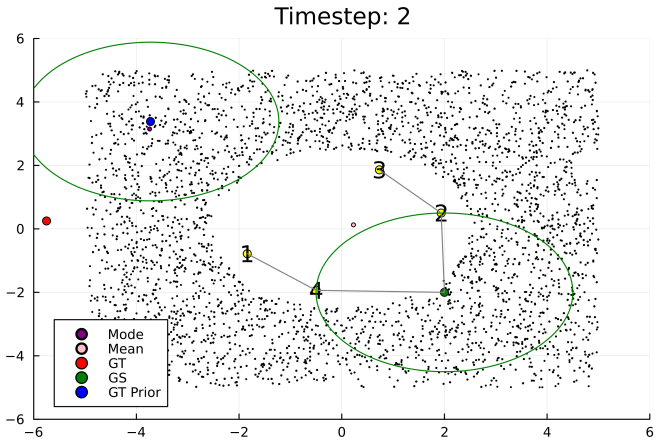

Fig. 12: FIM Horizon: Agents spread out to search large area

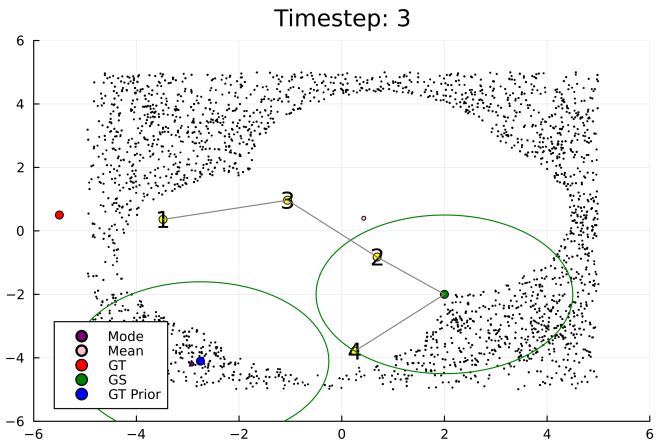

Fig. 13: FIM Horizon: Network aligned to catch estimate predicted to move towards origin

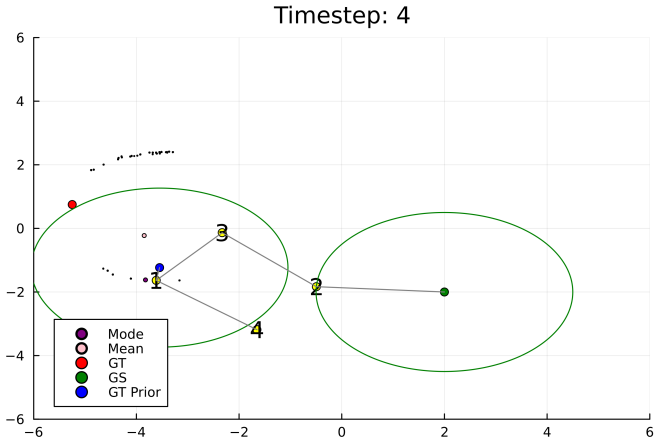

Fig. 14: FIM Horizon: Moving toward received measurement

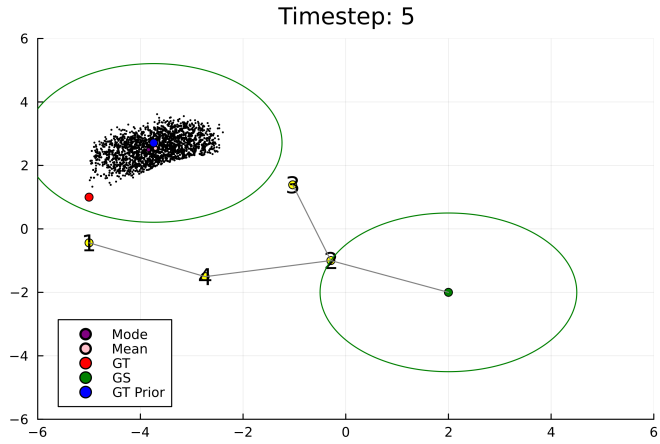

Fig. 15: FIM Horizon: Aligned network to catch estimate

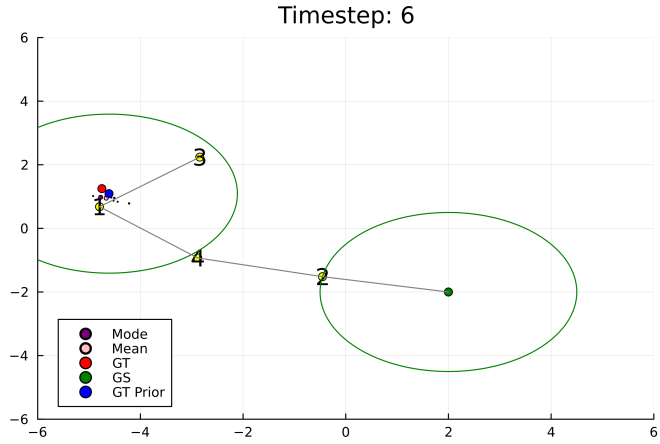

Fig. 16: FIM Horizon: Target localized

The three-step horizon planner not only was able to receive its first measurement of the target three time steps faster than the one-step planner, but was also able to reorganize itself effectively to rapidly changing estimates from the particle filter. This is particularly striking when comparing Figure 4 with Figure 13, as the one-step horizon is largely in the same configuration as it was at the second time step, and covering much of the same area. The three-step horizon planner is able to reason over the positions the estimator has predicted the target will be a short time in the future, and move with anticipation even when it does not expect to receive a measurement at the current time step.

C. One-step horizon with obstacles

The planner for an environment of 3 triangular obstacles with a network of 3 agents also successfully returned positions for each time step, and was able to locate and track the target. However, the algorithm took nearly an hour to return the plan, making larger problems essentially infeasible to solve in this way.

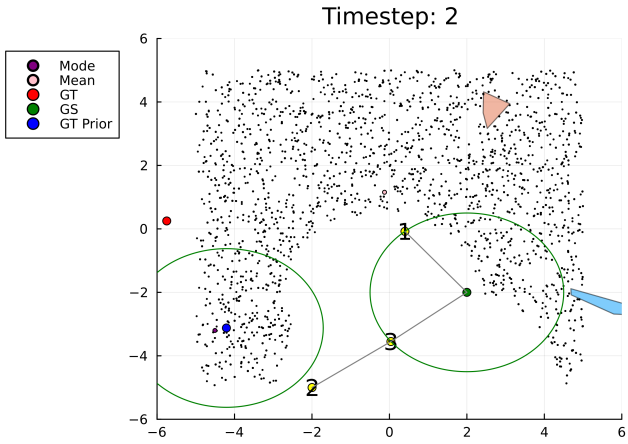

Fig. 17: Agents spread out to search large area

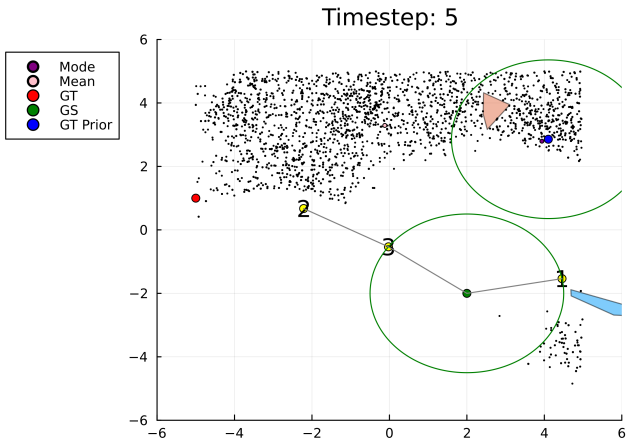

Fig. 18: Network extends to measure estimated target position.

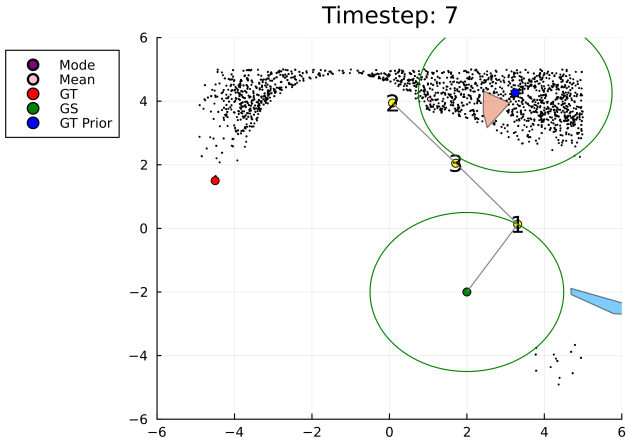

Fig. 19: Network shifts toward grouped particles

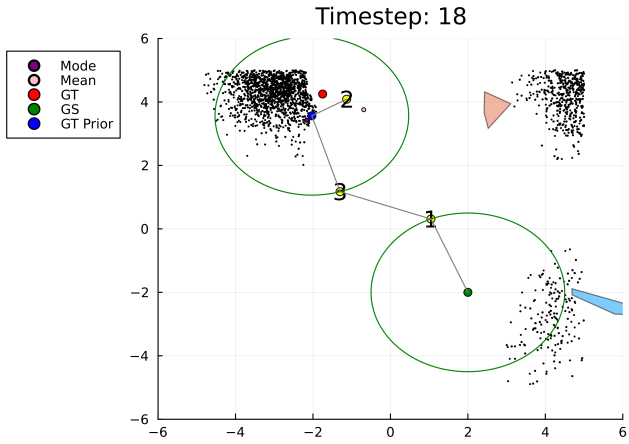

Fig. 20: Network moving to gather multiple measurements

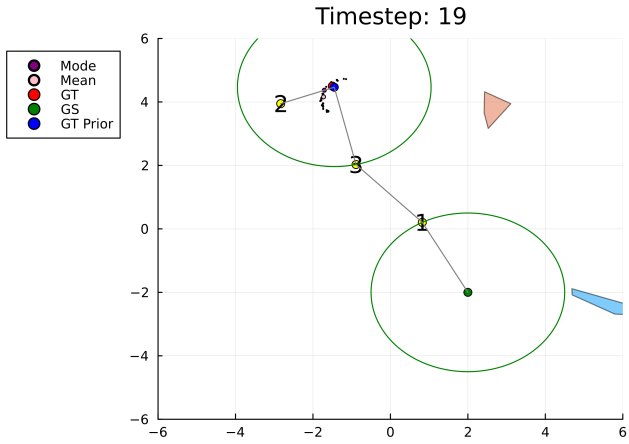

Fig. 21: Network successfully tracks target

VI. Conclusion

The multi-agent LOS network problem for ensuring connectivity between a set of agents in a cluttered environment was successfully solved using a Mixed-Integer Program approach and the Juniper solver for constraint integer programming. An information-based objective function was developed that improved the performance of the planner by more effectively searching the environment for the target. In addition, the multiple-horizon information-based objective function was able to improve the plans significantly, by reasoning about information gains over future predicted target states.

While a valid solution to the network planning problem in a cluttered environment was generated, the computation time required to solve the problem for a small network in an environment with a few obstacles required makes this planning algorithm infeasible in an actual mission environment. Further work could investigate the actual optimality of the generated paths with respect to the reduction in uncertainty of the target state, and less computationally intensive obstacle avoidance and line of sight constraint formulations to increase the feasibility of generating information sensitive plans in cluttered environments.

REFERENCES

[1] T. Zhang, V. Qin, Y. Tang, and N. Li, "Distributed information-based source seeking," *IEEE Transactions on Robotics*, vol. 39, no. 6, pp. 4749–4767, 2023. DOI: 10.1109/TRO.2023.3309099.

[2] M. Stachura and E. Frew, "Cooperative target localization with a communication-aware unmanned aircraft system," *Journal of Guidance, Control, and Dynamics*, vol. 34, Sep. 2011. DOI: 10.2514/1.51591.

[3] A. Ryan and J. K. Hedrick, "Particle filter based information-theoretic active sensing," *Robotics and Autonomous Systems*, vol. 58, no. 5, pp. 574–584, 2010, ISSN: 0921-8890. DOI: https://doi.org/10.1016/j.robot.2010.01.001. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0921889010000023.

[4] G. M. Hoffmann and C. J. Tomlin, "Mobile sensor network control using mutual information methods and particle filters," *IEEE Transactions on Automatic Control*, vol. 55, no. 1, pp. 32–47, 2010. DOI: 10.1109/TAC.2009.2034206.

[5] Z. Sunberg, *Particlefilters.jl*, version v0.5.3, 2021. [Online]. Available: https://github.com/JuliaPOMDP/ParticleFilters.jl/tree/master.

[6] D. Ioan, I. Prodan, S. Olaru, F. Stoican, and S.-I. Niculescu, "Mixed-integer programming in motion planning," *Annual Reviews in Control*, Nov. 2020. DOI: 10.1016/j.arcontrol.2020.10.008. [Online]. Available: https://centralesupelec.hal.science/hal-03108529.

[7] M. A. Dar, A. Fischer, J. Martinovic, and G. S. and, "An improved flow-based formulation and reduction principles for the minimum connectivity inference problem," *Optimization*, vol. 68, no. 10, pp. 1963–1983, 2019. DOI: 10.1080/02331934.2018.1465944.

[8] R. Zhou, Y. Feng, B. Di, J. Zhao, and Y. Hu, "Multi-uav cooperative target tracking with bounded noise for connectivity preservation," *Frontiers of Information Technology & Electronic Engineering*, vol. 21, no. 10, pp. 1494–1503, Oct. 2020, ISSN: 2095-9230. DOI: 10.1631/FITEE.1900617. [Online]. Available: https://doi.org/10.1631/FITEE.1900617.

[9] A. Caregnato-Neto, M. R. O. A. Maximo, and R. J. M. Afonso, "A novel line of sight constraint for mixed-integer programming models with applications to multi-agent motion planning," in *2023 European Control Conference (ECC)*, 2023, pp. 1–6. DOI: 10.23919/ECC57647.2023.10178275.

[10] O. Kröger, C. Coffrin, H. Hijazi, and H. Nagarajan, "Juniper: An open-source nonlinear branch-and-bound solver in julia," in *Integration of Constraint Programming, Artificial Intelligence, and Operations Research*, Springer International Publishing, 2018, pp. 377–386, ISBN: 978-3-319-93031-2.