

Enhancing Security in Multi-Robot Systems through Co-Observation Planning on Unsecured Trajectories

Ziqi Yang and Roberto Tron

Abstract—This paper addresses the critical issue of security in multi-robot systems (MRS) when facing threats from adversaries attempting to compromise the robots’ control which could lead to unauthorized access to forbidden areas, potentially causing harm or disrupting the mission. To tackle this problem, we propose a novel multi-robot planning algorithm that leverages mutual observations between robots, and introducing *reachability* constraints as additional security measures. By enforcing an empty intersection between the locations that the agents could potentially reach (using reachability constraint) and the forbidden regions, we provide guarantees that even in the face of adversarial movement by attackers, compromised robots cannot reach forbidden regions without missing scheduled co-observations. The reachability constraint is implemented with ellipsoidal over-approximation; this allows to quickly check intersection and compute gradients.

Furthermore, to further enhance system resilience and address the feasibility challenges encountered in the initial stage, we introduce the concept of *sub-teams*, where multiple robots form cohesive units along each route, replacing individual robot assignments. This allows redundant robots to deviate for co-observations across different sub-teams, ensuring the security of multiple trajectories without the need for trajectory modifications of the entire sub-team. To plan cross-trajectory co-observations that maintain security against plan-deviation attacks, we formulate the multi-flow problem on unsecured MRS trajectories,

We demonstrate the effectiveness and robustness of our proposed algorithm, which significantly strengthens the security of multi-robot systems in the face of adversarial threats.

I. INTRODUCTION

Multi-robot systems (MRS) have found wide applications in various fields, including warehouse organization, surveillance and precision agriculture, owing to their capacity for complex task execution through cooperation and coordination [1], [2]. While MRS offers numerous advantages, their distributed nature and dependence on network communication render them vulnerable to cyber threats, introducing new cybersecurity challenges such as unauthorized access, malicious attacks, and data manipulation [3]. This paper addresses the specific scenario in which attackers compromise robots and direct them to restricted regions, potentially leading to the exploitation of confidential information, physical property damages, or even human injuries [4]. We define these restricted regions as *forbidden regions*, which may contain security-sensitive equipment or human workers.

We focus on countering deliberate deviations, termed *plan-deviation attacks*, identified and addressed in previous studies [5]–[9]. As a security measure, robots perform inter-robot observations using their onboard sensing capabilities. Each robot’s self-report includes its own status and additional

observations of other robots’ status. This mutual observation and reporting establish an observation schedule for the path, ensuring that any attempts by a compromised robot to violate safety constraints (such as transgressing forbidden regions) would break the observation plan and be promptly detected.

In this paper, we transform the solution to continuous configuration spaces, allowing for more flexible objectives. We provide additional analysis on the reachability of robots to ensure the security of the co-observation schedule. Solving the path optimization problem with constraints based on the sets of locations that the agents could potentially reach, referred to as *reachability regions*, is a novel contribution to the literature.

A preliminary version of the paper was presented at conferences [8], [9]. Our proposed approach enforces an empty intersection between forbidden regions and an ellipsoidal reachability region during trajectory optimization, preventing undetected attacks if an adversary gains control of the robots. Inspired by the *heuristic sampling domain* introduced by [10] within the context of the RRT* path planning algorithm, we utilize an ellipsoidal boundary to constrain the search space and formulate the ellipsoid as the reachability region constraint. We present a mathematical formulation of reachability regions compatible with the solver as a spatio-temporal constraint.

This paper also extend the previous works by fixing two main challenges identified in [7]–[9]. Firstly, finding a co-observation and reachability secured plan may not always be feasible when robots are separated from others by obstacles or forbidden regions, or are located far away from each other. In such case, it is impossible for other robots to get close enough to establish co-observation schedules or find a reachability-secured path. Agent 3 in the Figure [zyang ref fig:ReachabilitySimulation](#) is a good example, as it required additional security checkpoint to create secured reachability areas. Secondly, security requirements may come at the cost of overall system performance, as illustrated by the comparison of Figure [zyang ref fig:trajectories-more-constraint](#) and [zyang ref fig:ReachabilitySimulation](#) where, after the introduction of security constraints, the top left corner is never explored by any robots. This is particularly concerning given that system performance is a key factor in the decision to use multi-agent systems. These challenges are addressed in Chapter III, in order to ensure the effectiveness of planning with reachability and co-observation in securing multi-agent systems.

We propose to form *sub-teams* on each route, and setup additional co-observations both within the sub-team and across different sub-teams. Notice that co-observations across sub-teams (named *cross-trajectory co-observation*) are always preferred when feasible, since they provide better security in

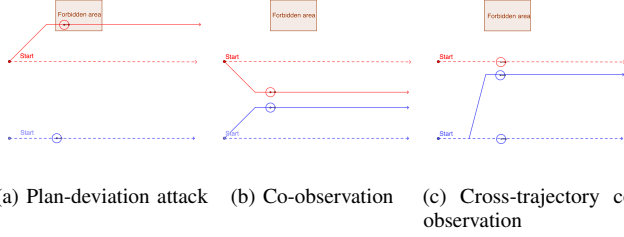


Fig. 1: 1b Limited by the co-observation requirement, both red and blue robot follow the co-observation secured routes (solid lines) and abandon the optimal ones (dashed line). 1c Through cross-trajectory co-observations, the blue team sends one robot to follow the red team (solid blue line) and performs co-observation while having the rest of the robots following the optimal trajectory.

potential situations where the entire sub-team is compromised. In this paper, we consider only the high-level path planning, while the assignment of the duties in the team will be performed dynamically as introduced later in later works, making it more difficult for attackers to successfully plan and execute attacks. Cross-trajectory co-observations allows sub-teams to preserve the optimal unsecured trajectories (as shown in Figure 1), as it does not require the entire *sub-team* to maintain a close distance with other teams when co-observation occurs.

a) Paper contributions: In this paper, we formulate a way to enforce an empty intersection between forbidden region and reachability regions, such that if an attacker takes control of the robots, they cannot perform an undetected attack by entering forbidden regions and meeting co-observation schedules at the same time. The constraints are formulated using an ellipsoidal bound of the reachability region.

We propose to incorporate redundancy in the form of *sub-teams* of multiple robots in place of individual robots along each route. This allows robots to deviate from their assigned sub-team, and join others to perform cross-trajectory co-observations, thereby securing multiple trajectories. We propose a formulation of the multi-flow problem on unsecured MRS trajectories to plan the cross-trajectory co-observations that can preserve the security against plan-deviation attacks.

II. MULTI-ROBOT TRAJECTORY PLANNING WITH CO-OBSERVATION

In this section, we aim to enhance the security of MRSs against malicious takeovers and deviations by attackers. Specifically, we design an inter-robot observation plan (co-observation schedule) to ensure that, during the task period, robots are constantly in proximity to one another according to a schedule, in order to observe and detect potential hazardous behaviors. This is achieved by keeping the potential region that the robots can possibly reach between each consecutive co-observations away from the forbidden regions.

Definition 1. A multi-robot trajectory plan is secure against plan-deviation attacks if it ensures that any potential deviations

to these forbidden regions will cause the corresponding robot to miss their next co-observation with other robots.

This problem is first solved by [5] in grid world for robots with fixed start and goal locations (Figure 2b) but does not offer the ability to optimize arbitrary smooth cost functions of the trajectories.

We formulate the planning problem as an optimal trajectory optimization problem to minimize arbitrary smooth objective functions. We denote as $q_{ij} \in \mathbb{R}^m$ the position of agent i at the discrete-time index j , with m representing the dimension of the state space. For a total of n_p robots, and a task time horizon of T , a total of n_p trajectories can be represented as an aggregated vector $\mathbf{q} \in \mathbb{R}^{nmT}$. The goal of our path planning problem is to minimize or maximize an objective function $\Phi(\mathbf{q})$ under a set of nonlinear constraints described by a set Ω . The set Ω is given by the intersection of spatio-temporal sets given by security constraints (co-observation schedule, reachability analysis) and traditional path planning constraints. Formally:

$$\begin{aligned} & \min / \max \quad \Phi(\mathbf{q}) \\ & \text{subject to} \quad \mathbf{q} \in \Omega. \end{aligned} \quad (1)$$

To give a concrete example of the cost Φ and the set Ω , we introduce a representative application that will be used for all the simulations throughout the paper.

Example 1. We consider the estimation of a slowly-varying scalar or vector field, denoted as \mathbf{x} , at discrete locations within an unknown environment. Autonomous agents navigate this environment, collecting sensory data to construct a corresponding map (see Figure 2a). Our map comprises points of interest arranged on a grid, each associated with a slowly-changing value. Our goal is to find paths that minimize uncertainty and effectively reconstruct the field. We associate a Kalman Filter (KF) (cf. [11]) to each point j , and use it to track the uncertainty through its estimated covariance P_j . The updates of the filters are based on measurements from the robot centered at q , and we use a Gaussian radial basis function modeling spatially-varying measurement quality.

rtron Cite conference article for details on the problem formulation. The optimization objective $\Phi(\mathbf{q})$ is the maximum uncertainty $\max_j P_j$. The set Ω includes traditional constraints like bounded velocity, convex obstacles, waypoints to reach with deadlines, in addition to security constraints including co-observation and reachability constraints.

rtron Make sure we refer back to Example 1 when makes sense

This section is organized as follows. We begin with foundational mathematical concepts, such as the basic ADMM optimal trajectory solver. Subsequently, our solver allows us to seamlessly integrate security constraints, encompassing co-observation schedule and reachability constraints. Concluding this section, we present the obtained results alongside limitations (due to infeasibility) that are then addressed in Section III.

A. Preliminaries

We review in this section the mathematical foundations for our ADMM solver and the constraints that we will use.

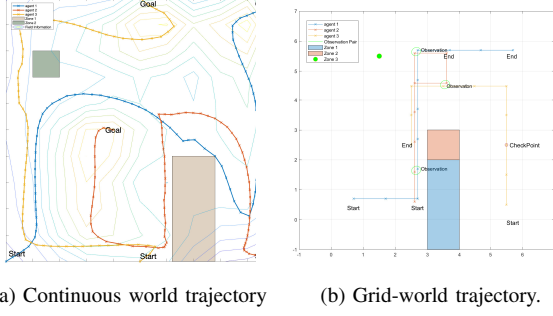


Fig. 2: Trajectory design of an example map exploration task for a three-robot system with start locations and destinations fixed. Task are planned in a 8×8 grid world. Zone 1 is obstacle, Zone 2 and Zone 3 are safe zones. Fig 2b shows the planned trajectory with co-observation schedule in grid-world. Fig 2a shows the trajectory design in continuous world optimized with respect to a map exploration task

1) *Differentials*: We define the differential of a map $f(x) : \mathbb{R}^m \rightarrow \mathbb{R}^n$ at a point x_0 as the unique matrix $\partial_x f \in \mathbb{R}^{n \times m}$ such that

$$\left. \frac{d}{dt} f(x(t)) \right|_{t=0} = \partial_x f(x(0)) \dot{x}(0) \quad (2)$$

where $t \mapsto x(t) \in \mathbb{R}^n$ is a smooth parametric curve such that $x(0) = x$ with any arbitrary tangent $\dot{x}(0)$. For brevity we will use \dot{f} for $\frac{d}{dt} f$ and $\partial_x f$ for $\frac{\partial f}{\partial x}$.

With a slight abuse of notation, we use the same notation $\partial_x f$ for the differential of a matrix-valued function with scalar arguments $f : \mathbb{R}^R \rightarrow \mathbb{R}^{m \times n}$. Note that in this case (2) is still formally correct, although the RHS needs to be interpreted as applying $\partial_x f$ as a linear operator to \dot{x} .

2) *Householder rotations*: In the derivation of the reachability constraint in Sections II-F and II-J, we need to define a differentiable transformation to a canonical ellipse. This transformation includes a rotation derived from a modified version of Householder transformations [12]. With respect to the standard definition, our modification ensures that the final operator is a proper rotation (i.e., not a reflection). We call our version of the operator a *Householder rotation*. In this section we derive Householder rotations and their differentials for the 3-D case; the 2-D case can be easily obtained by embedding it in the $z = 0$ plane.

Definition 2. Let $\nu_{\mathcal{F}}$ and $\nu_{\mathcal{E}}$ be two unitary vectors ($\|\nu_{\mathcal{F}}\| = \|\nu_{\mathcal{E}}\| = 1$). Define the normalized vector u as

$$u' = \nu_{\mathcal{F}} + \nu_{\mathcal{E}}, \quad u = \frac{u'}{\|u'\|}. \quad (3)$$

The Householder rotation $H(\nu_{\mathcal{F}}, \nu_{\mathcal{E}})$ is defined as

$$H(\nu_{\mathcal{F}}, \nu_{\mathcal{E}}) = 2uu^T - I. \quad (4)$$

The main property of interest for our application is the fact that H is a rotation mapping $\nu_{\mathcal{F}}$ to $\nu_{\mathcal{E}}$, as shown by the following.

Proposition 1. The matrix H has the following properties:

1) It is a rotation, i.e.

- a) $H^T H = I$;
 - b) $\det(H) = 1$.
- 2) $\nu_{\mathcal{E}} = H\nu_{\mathcal{F}}$.

Proof. See Appendix A. \square

We compute the differential of H implicitly using its definition (2). We use the notation $[v]_{\times} : \mathbb{R}^3 \rightarrow \mathbb{R}^{3 \times 3}$ to denote the matrix representation of the cross product with the vector v , i.e.,

$$[v]_{\times} = \begin{bmatrix} 0 & -v_3 & v_2 \\ v_3 & 0 & -v_1 \\ -v_2 & v_1 & 0 \end{bmatrix}, \quad (5)$$

such that $[v]_{\times} w = v \times w$ for any $w \in \mathbb{R}^3$.

Proposition 2. Let $\nu_{\mathcal{F}}(t)$ represent a parametric curve. Then

$$\dot{H} = H[-2M\dot{\nu}_{\mathcal{F}}]_{\times}, \quad (6)$$

where the matrix $M \in \mathbb{R}^{3 \times 3}$ is given by

$$M = [u]_{\times} \frac{(I - uu^T)(I - \nu_{\mathcal{F}}\nu_{\mathcal{F}}^T)}{\|u'\|\|\nu_{\mathcal{F}}\|}. \quad (7)$$

Proof. See Appendix B. \square

3) *Alternating Directions Method of Multipliers (ADMM)*: The basic idea behind our ADMM-based solver [9] is to split the constraints from the objective function using a different set of variables \mathbf{z} , and then solve an Augmented Lagrangian formulation of the optimization problem in (1). More specifically, we can rewrite the constraint $\mathbf{q} \in \Omega$ using an indicator function Θ , and include it in the objective function. In the traditional application of ADMM, the variables \mathbf{z} are simple duplicates of \mathbf{x} projected to the constraint set Ω . However, in path planning problems, some constraints are non-convex, rendering the projection step more difficult (due to the presence of multiple local minima). To allow for an easier projection step, we propose a minor generalization of the ADMM formulation where we allow \mathbf{z} to replicate an arbitrary function of the main variables \mathbf{q} (instead of being an exact copy), to transform constraint $\mathbf{q} \in \Omega$ to $D(\mathbf{q}) \in \mathcal{Z}$. In summary, we transform Equation (1) into

$$\begin{aligned} \max \quad & \Phi(\mathbf{q}) + \Theta(\mathbf{z}) \\ \text{s.t.} \quad & D(\mathbf{q}) - \mathbf{z} = 0 \end{aligned} \quad (8)$$

where $D(\mathbf{q}) = [D_1(\mathbf{q})^T, \dots, D_l(\mathbf{q})^T]^T$ is a vertical concatenation of different functions for different constraints.

The update steps of the ADMM algorithm are then derived as [13]:

$$\mathbf{q}^{k+1} := \underset{\mathbf{q}}{\operatorname{argmin}} (\Phi(\mathbf{q}^k) + \frac{\rho}{2} \|D(\mathbf{q}) - \mathbf{z}^k + \mathbf{u}^k\|_2^2), \quad (9a)$$

$$\mathbf{z}^{k+1} := \Pi_{\mathcal{Z}}(D(\mathbf{q}^{k+1}) + \mathbf{u}^k), \quad (9b)$$

$$\mathbf{u}^{k+1} := \mathbf{u}^k + D(\mathbf{q}^{k+1}) - \mathbf{z}^{k+1}, \quad (9c)$$

where $\Pi_{\mathcal{Z}}$ is the new projection to the modified constraint set \mathcal{Z} , \mathbf{u} represents a scaled dual variable that, intuitively, accumulates the sum of primal residuals

$$\mathbf{r}^k = D(\mathbf{q}^{k+1}) - \mathbf{z}^{k+1}. \quad (10)$$

Checking the primal residuals alongside with the dual residuals

$$\mathbf{s}^k = -\rho(\mathbf{z}^k - \mathbf{z}^{k-1}) \quad (11)$$

after each iteration, the steps are reiterated until convergence when the primal and dual residuals are small, or divergence when primal and dual residual remains large after a fixed number of iterations. Since $D(\mathbf{q})$ is the vertical concatenation of $D_i(\mathbf{q})$, the projection of each set \mathcal{Z}_i is independent for each constraint and can be computed separately. The advantage of this formulation is that we can choose $D(\mathbf{q})$ such that the new constraint set \mathcal{Z} becomes simple to compute, and the function $D(\mathbf{q})$ can be used to select only the subset of the variables on which a constraint depends, speeding up computations; however, the drawback is that we move the non-convexity to the primal cost function, i.e., in the update for \mathbf{q} in (9a).

We now provide the functions $D(\mathbf{q})$, the sets \mathcal{Z} , and the corresponding projection operators $\Pi_{\mathcal{Z}}$ for traditional path planning constraints (Section II-B-Section II-D), co-observation security constraints (Section II-E), and reachability constraints (Section II-G-Equation (56)). The latter are based on the definition of *ellipse-region-constraint* (Section II-F).

B. Velocity constraints

The movement of each agent is constrained by a maximum velocity defined as the distance in any direction over a single discrete time step. We define the function $D(\mathbf{q})$ to return the velocity vectors for the i -th agent at time step j , and constrain its result to a sphere.

ADMM constraint 1 (Velocity constraint).

$$D_{ij}(\mathbf{q}) = q_{ij} - q_{i(j-1)}, \quad j \in \{1, \dots, T\}, \quad (12)$$

$$\mathcal{Z}_{ij} = \{\mathbf{z} \mid \|\mathbf{z}\| \leq v_{max}\}, \quad (13)$$

$$\Pi_{\mathcal{Z}_{ij}}(\mathbf{z}) = \begin{cases} v_{max} \frac{\mathbf{z}}{\|\mathbf{z}\|} & \text{if } \|\mathbf{z}\| > v_{max}, \\ \mathbf{z} & \text{otherwise.} \end{cases} \quad (14)$$

C. Convex polygonal obstacles collision constraints

We use convex polygons to model regions defined by solid obstacles and forbidden regions that cannot be entered by agents. The convex region \mathcal{P} is defined using a collection of l hyperplanes $n_k q = m_k$, where n_k is the normal vector pointing toward the outside of the obstacle and m_k is the scalar offset defining the k -th hyperplane. The distance between a single waypoint q_{ij} and the k th hyperplane $d_k(q_{ij}, \mathcal{P})$ is defined as:

$$d_k(q_{ij}, \mathcal{P}) = p_{ij}^T n_k - m_k. \quad (15)$$

We define the function $D(\mathbf{q})$ to return the maximum distance from q_{ij} to all hyperplane boundaries of region \mathcal{P} , and constraint z to be a non-negative scalar. This constraint is applied to all $q_{ij} \in \mathbf{q}$.

ADMM constraint 2 (Obstacle constraint).

$$D(q_{ij}) = \max_{k=1, \dots, l} (d_k(q_{ij}, \mathcal{P})), \quad (16)$$

$$\mathcal{Z} = \{z \mid z \geq 0\}, \quad (17)$$

$$\Pi_{\mathcal{Z}}(z) = \begin{cases} 0 & \text{if } z < 0, \\ z & \text{otherwise.} \end{cases} \quad (18)$$

The motivating idea for (17) and (18) is to identify waypoints within a specific region and project them to the closest boundary. Non-convex obstacles can be handled by the union of (possibly overlapping) convex obstacles. By using outward-pointing normals, this method can also serve to constrain the robots within a designated workspace region.

D. Waypoints with flexible deadlines

Security-related constraints often necessitate that a robot reaches a specified location at a particular time. We assume that robot i is tasked with reaching a given point p within a radius of d_{max} at some time instant j within a predefined time window $[t_1, t_2]$, leading to the following:

ADMM constraint 3 (Waypoint constraint).

$$D(\mathbf{q}) = \min_{j \in \{t_1, \dots, t_2-1\}} (\text{dist}(p, \overrightarrow{q_{ij}q_{i(j+1)}})), \quad (19)$$

$$\mathcal{Z} = \{z \mid z < z_{max}\}, \quad (20)$$

$$\Pi_{\mathcal{Z}}(z) = \min(z, d_{max}), \quad (21)$$

where $\text{dist}(p, \overrightarrow{q_{ij}q_{i(j+1)}})$ returns the distance between the fixed point p and the segment $\overrightarrow{q_{ij}q_{i(j+1)}}$.

Note that this function returns the smallest distance between (p, q_{ij}) and $(p, q_{i(j+1)})$ if the projection of the point p does not lie on the line segment $\overrightarrow{q_{ij}q_{i(j+1)}}$; as a consequence, this constraint does not need to be satisfied exactly at one of the points on the discretized trajectory, but it can also be satisfied “en route” on the segment between them. [zyang plot new simpler example, add collision avoidance](#)

E. Co-observation schedule constraint

The co-observation constraint ensures that two robots come into close proximity at scheduled times to observe each other's behavior. This constraint is represented as a relative distance requirement between the two robots at a specific time instant, ensuring they are within a defined radius to inspect each other or exchange data.

ADMM constraint 4 (Co-observation constraint).

$$D(\mathbf{q}) = \overrightarrow{q_{aj}q_{bj}}, \quad (22)$$

$$\mathcal{Z} = \{\mathbf{z} \mid \|\mathbf{z}\| \leq d_{max}\}, \quad (23)$$

$$\Pi_{\mathcal{Z}}(z) = \begin{cases} d_{max} \frac{\mathbf{z}}{\|\mathbf{z}\|} & \text{if } \|\mathbf{z}\| > d_{max}, \\ \mathbf{z} & \text{otherwise,} \end{cases} \quad (24)$$

where a, b are the indices of the pair of agents required for a mutual inspection.

The locations q_{aj} and q_{bj} where the co-observation is performed are computed as part of the optimization.

F. Definition of ellipsoidal reachability regions

In this section, we define *ellipsoidal reachability regions* based on pairs of locations on a trajectory, and a transformation of such region in axis-aligned form. These definitions will be used to define different types of reachability constraints (between an ellipsoid and different geometric entities such as

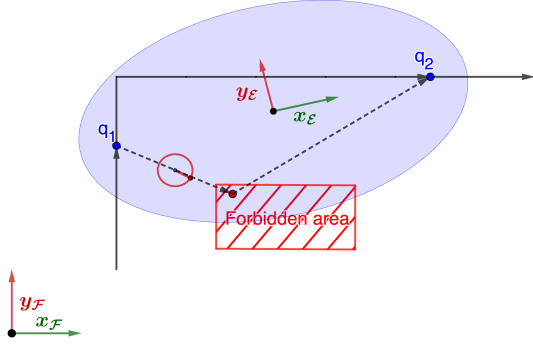


Fig. 3: The ellipse showcases the reachability region. The black line is the planned trajectory of a robot, q_1 and q_2 are two locations this robot are expected at given time t_1 and t_2 , dashed line is a possible trajectory if the robot is compromised after reaching q_1 . Axis of global frame \mathcal{F} and canonical frame \mathcal{F}_E are shown as $x_{\mathcal{F}}, y_{\mathcal{F}}$ and x_E, y_E respectively.

a point, line, line segment, and polygon), their projections, and differentials in subsequent sections. Together with ?? 4, these will be used in the ADMM formulation of section II-A3 to provide our security guarantees.

Definition 3. The reachability region for two waypoints $q(t_1) = q_1$, $q(t_2) = q_2$ is defined as the sets of points q' in the workspace such that there exist a trajectory $q(t)$ where $q(t') = q'$, $t_1 \leq t' \leq t_2$ and $q(t)$ satisfies the velocity constraint $d(q(t), q(t+1)) \leq v_{max}$.

This region can be analytically bounded via an ellipsoid:

Definition 4. The reachability ellipsoid \mathcal{E} is defined as the region $\mathcal{E}(q_1, q_2, t_1, t_2) = \{\tilde{q} \in \mathbb{R}^n : d(q_1, \tilde{q}) + d(\tilde{q}, q_2) < 2a\}$, where $a = \frac{v_{max}}{2}(t_2 - t_1)$.

The region $\mathcal{E}(q_1, q_2)$ is an ellipsoid with foci at q_1, q_2 , center $o_E = \frac{1}{2}(q_1 + q_2)$, and the major radius equal to a . We denote as $c_E = \frac{1}{2}\|q_1 - q_2\| = \|o_E - q_1\|$ the semi-axis distance from the center to a foci. The reachability ellipsoid is an over-approximation of the exact reachability region in Definition 3; the difference between the two is due to the discretization of the trajectory, and the fact that \mathcal{E} does not consider the presence of obstacles.

Expanding upon the concept of *co-observation* and *reachability*, Definition 1 can be written as,

Remark 1. A multi-robot trajectory is secured against plan-deviation attacks if there exist a co-observation plan such that the reachability region between each consecutive co-observation does not intersect with any forbidden regions.

1) *Transformation to canonical coordinates:* To simplify the problem, we employ a canonical rigid body transformation that repositions the ellipse \mathcal{E} from the global frame \mathcal{F} to a canonical frame \mathcal{F}_E .

The canonical frame \mathcal{F}_E is defined with the origin at the ellipsoid's center, and the first axis aligned with the foci Figure 3. We perform this transformation in a differentiable manner, with the two foci serving as waypoints for a robot. We parametrize the transformation from \mathcal{F} to \mathcal{F}_E using a rotation $R_E^{\mathcal{F}}$ and a translation $o_E^{\mathcal{F}}$, which, to simplify the notation, we

refer to as R and o , respectively. The transformation of a point from the frame \mathcal{F}_E to the frame \mathcal{F} and its inverse are given by the rigid body transformations:

$$q^{\mathcal{F}} = Rq^E + o, \quad q^E = R^T(q^{\mathcal{F}} - o). \quad (25)$$

We define $\nu_{\mathcal{F}}$ and ν_E to represent the x -axis unitary vector of \mathcal{F}_E in the frames \mathcal{F} and \mathcal{F}_E , respectively. Formally:

$$\nu_{\mathcal{F}}' = q_2 - q_1, \quad \nu_{\mathcal{F}} = \frac{\nu_{\mathcal{F}}'}{\|\nu_{\mathcal{F}}'\|}, \quad \nu_E = [1, 0, 0]^T; \quad (26)$$

see Fig.3 for an illustration. Note that ν_E is constant while, for the sake of clarity, $\nu_{\mathcal{F}}$ depends on q_1, q_2 . From the definition of \mathcal{F}_E , the rotation R can be found using a *Householder rotation*, while from (25) the translation o must be equal to the center of $\mathcal{E}(q_1, q_2)$ expressed in \mathcal{F} , i.e.:

$$R = H(\nu_{\mathcal{F}}(q_1, q_2), \nu_E(q_1, q_2)), \quad o = \frac{1}{2}(q_1 + q_2). \quad (27)$$

To simplify the notation, in the following, we will consider H to be a function of q_1, q_2 directly, i.e. $H(q_1, q_2)$. The ellipse \mathcal{E} expressed in \mathcal{F}_E is given by $\mathcal{E}^E = \{q^E \in \mathbb{R}^m : d(q_1^E, q^E) + d(q^E, q_2^E) < 2a\}$, with foci q_1^E, q_2^E in \mathcal{F}_E defined as

$$q_1^E = [c \ 0 \ 0]^T, \quad q_2^E = [-c \ 0 \ 0]^T, \quad (28)$$

and semi-axis distance $c = \frac{\|q_2 - q_1\|}{2}$.

Definition 5. The reachability ellipsoid \mathcal{E} in the canonical frame is defined as the zero level set of the quadratic function

$$E^E(q^E) = q^{E^T} Q q^E - 1 \quad (29)$$

where

$$Q = \text{diag}(a^{-2}, b^{-2}, b^{-2}), \quad (30)$$

and $b = \sqrt{a^2 - c^2}$. The ellipse parameters a, b represent the lengths of the major axes.

Lemma 1. The original ellipse \mathcal{E} in \mathcal{F} can be expressed as the zero level set of the quadratic function

$$E^{\mathcal{F}}(q^{\mathcal{F}}) = (q^{\mathcal{F}} - o_E)^T H^T Q H (q^{\mathcal{F}} - o_E) - 1 \quad (31)$$

Proof. The claim follows by substituting (25) into (29), and from the definition of R and o . \square

For all reachability ellipsoid constraints, we first solve the problem in the canonical frame \mathcal{F}_E , and transform the solutions to the global frame \mathcal{F} using the transformation (25).

We are now ready to formulate constraints based on reachability ellipsoids against different types of forbidden regions: a point, a plane, a segment, and a convex polygon. For each one, our goal is to define the function $D(q)$, its differential $\partial_q D(q)$, the set ζ and the projection Π_{ζ} that can be incorporated in the ADMM optimization (8).

G. Point-ellipsoid reachability constraint

zyang change this to a circle by adding radius to projection? As shown in Fig.4, we consider a forbidden region in the shape of a single point q_{avoid} . The goal is to design the trajectory $q(t)$ such that $q_{avoid} \notin \mathcal{E}(q_1, q_2)$. This constraint can be written as,

ADMM constraint 5 (Point-ellipsoid reachability constraint).

$$D(\mathbf{q}) = \begin{cases} \pi_p \mathcal{E}(\mathbf{q}) - q_{avoid} & q_{avoid} \in \mathcal{E}, \\ 0 & otherwise. \end{cases} \quad (32)$$

$$\mathcal{Z} = \{q \in \mathbb{R}^{nm} : \|D_p(\mathbf{q})\| = 0\}, \quad (33)$$

$$\Pi_{\mathcal{Z}}(\mathbf{z}) = 0, \quad (34)$$

where $\pi_{p\mathcal{E}}(q_{avoid}; q_1, q_2, a) = q_p$ is a projection function that returns a projected point q_p of q_{avoid} outside the ellipse, i.e., as the solution to

$$\pi_{p\mathcal{E}} = \underset{q_p \in \mathcal{E}^c}{\operatorname{argmin}} \quad \|q_{avoid} - q_p\|^2 \quad (35)$$

where \mathcal{E}^c is the set complement of region \mathcal{E} .

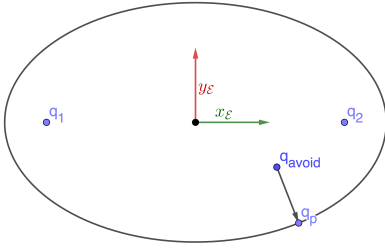


Fig. 4: Point-ellipsoid constraint, a point q_{avoid} inside ellipsoid is projected to the areas outside the ellipsoid f_p .

For cases where $q_{avoid} \notin \mathcal{E}(q(t_2), q(t_1), r)$, $\pi_{p\mathcal{E}}(q_{avoid}) = q_{avoid}$. And for cases where $q_{avoid} \in \mathcal{E}(q_1, q_2, r)$, $D(q)$ needs to be projected to the boundary of the ellipse. In the canonical frame $\mathcal{F}_{\mathcal{E}}$, the projected point $q_p^{\mathcal{E}}$ can be written as:

$$q_p^{\mathcal{E}} = (I + sQ)^{-1} q_{avoid}^{\mathcal{E}} = Sq_{avoid}^{\mathcal{E}} \quad (36)$$

where s can be solved as the root of the level set (29):

$$q_p^{\varepsilon^T} Q q_p^{\varepsilon} - 1 = q_{avoid}^{\varepsilon^T} Q'(s) q_{avoid}^{\varepsilon} - 1 = 0 \quad (37)$$

where

$$Q'(s) = S^T Q S = \text{diag} \left(\frac{a^2}{(s+a^2)^2}, \frac{b^2}{(s+b^2)^2}, \frac{b^2}{(s+b^2)^2} \right) \quad (38)$$

Detailed methods for computing s can be found in [8], [9], [14].

The point-to-ellipse projection function in \mathcal{F} is then:

$$\begin{aligned}\pi_{p\mathcal{E}}(q) &= R^{-1}(q(t_1), q(t_2))q_p^{\mathcal{E}} + o \\ &= R^{-1}(q(t_1), q(t_2))Sq_{avoid}^{\mathcal{E}} + o \\ &= R^{-1}SR(q_{avoid} - o) + o \quad (39)\end{aligned}$$

In our derivations, we consider only the 3-D case ($m = 3$); for the 2-D case, let $P = \begin{bmatrix} I & 0 \end{bmatrix} \in \mathbb{R}^{2 \times 3}$: then $\pi_{p\mathcal{E}}^{2D} = P\pi_{p\mathcal{E}}^{3D}(P^\top q_{avoid}; P^\top q_1, P^\top q_2, a)$.

Proposition 3. *The differential of the projection operator $\pi_{\mathcal{P}}(q_{\text{avoid}}; q_1, q_2, a)$ with respect to the foci q_1, q_2 is given by the following (using q as a shorthand notation for $q_{\text{avoid}}^{\mathcal{E}}$)*

$$\begin{aligned} \partial_{[q_1 \atop q_2]} \pi_{p\mathcal{E}} = & -2H[SH(q-o)]_{\times} U \\ & + ((q^T \partial_s Q' q)^{-1} H^{-1} Q' q q^T (4Q' H[q-o]_{\times} U \\ & + 2Q' H \partial_q o - \partial_b Q' q q \partial_q b) - s H^{-1} S^2 \partial_b Q q \partial_q b) \\ & - 2H^{-1} SH[q-o]_{\times} U + (H^{-1} SH - I) \partial_q o \quad (40) \end{aligned}$$

Proof. See Appendix C.

The differential of D_p is the same as the one for $\pi_p\mathcal{E}$.

H. Plane-ellipsoid reachability constraint

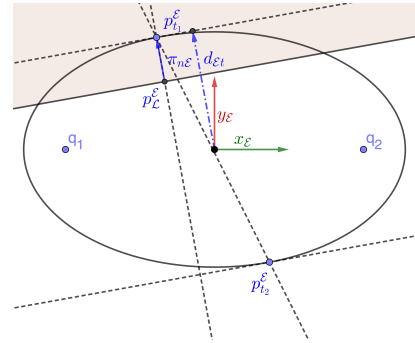


Fig. 5: Plane-ellipse constraint. Projected a ellipsoid to one side of a plane. The projection is simplified to the point-ellipse constraint as projecting a point inside ellipse p_L to p_t on outside the ellipse.

For forbidden region in the shape of a hyperplane $\mathcal{L}(q^F) = \{q^F \in \mathbb{R}^n : \mathbf{n}^T q^F = d\}$ (as shown in Figure 5), the reachability constraint can be defined as $\mathcal{L} \cap \mathcal{E}(q_1, q_2, a) = \emptyset$. When transformed into the canonical frame, the hyperplane can be written as $\mathcal{L}^\mathcal{E}(q^\mathcal{E}) = \{q^\mathcal{E} \in \mathbb{R}^m : \mathbf{n}_\mathcal{E}^T q^\mathcal{E} = d_\mathcal{E}\}$, where,

$$\mathbf{n}_{\mathcal{E}} = H(q_1, q_2)\mathbf{n}, \quad d_{\mathcal{E}} = -\mathbf{n}^T \mathbf{o} + d. \quad (41)$$

Intuitively, $d_{\mathcal{E}}$ can be thought as a distance between the the plane \mathcal{L} and the origin (i.e., the center of \mathcal{E}).

For every $\mathcal{L}^\varepsilon(q^\varepsilon)$, there always exist two planes that are both parallel to \mathcal{L} and tangential to the ellipse (i.e. resulting in a unique intersection point), $\mathcal{L}_1^\varepsilon = \{q^\varepsilon \in \mathbb{R}^m : \mathbf{n}_\varepsilon^\top q^\varepsilon = d_{\varepsilon t}\}$ and $\mathcal{L}_2^\varepsilon = \{q^\varepsilon \in \mathbb{R}^m : \mathbf{n}_\varepsilon^\top q^\varepsilon = -d_{\varepsilon t}\}$ (Figure 5). And the intersection point can be written as:

$$p_{t_1}^\varepsilon = \frac{d_{\varepsilon t} Q^{-1} n_\varepsilon}{n_\varepsilon^\text{T} Q^{-1} n_\varepsilon} = \frac{Q^{-1} n_\varepsilon}{d_{\varepsilon t}}, \quad p_{t_2}^\varepsilon = -p_{t_1}^\varepsilon, \quad (42)$$

where $d_{\mathcal{E}t} = \sqrt{n_{\mathcal{E}}^T Q^{-1} n_{\mathcal{E}}}$; intuitively, $d_{\mathcal{E}t}$ can be thought as a distance between the tangent plane $\mathcal{L}_1^{\mathcal{E}}$ (or $\mathcal{L}_2^{\mathcal{E}}$) and the origin (i.e., the center of the ellipse \mathcal{E}).

We introduce the concept of a tangent interpolation point on the hyperplane to characterize the relationship between the plane and the ellipse.

Definition 6. The tangent interpolation point $p_{\mathcal{L}}^{\mathcal{E}} \in \mathcal{L}$ is defined on the plane by

$$p_{\mathcal{L}}^{\mathcal{E}} = \frac{d_{\mathcal{E}} Q^{-1} n_{\mathcal{E}}}{n_{\mathcal{E}}^T Q^{-1} n_{\mathcal{E}}}. \quad (43)$$

Intuitively, the point $p_{\mathcal{L}}^{\mathcal{E}}$ is the point on \mathcal{L} which is closest to either $p_1^{\mathcal{E}}$ or $p_2^{\mathcal{E}}$. Note that when $d_{\mathcal{E}} = d_{\mathcal{E}t}$ or $d_{\mathcal{E}} = -d_{\mathcal{E}t}$, $p_{\mathcal{L}}^{\mathcal{E}} = p_{t_1}^{\mathcal{E}}$ or $p_{\mathcal{L}}^{\mathcal{E}} = p_{t_2}^{\mathcal{E}}$, respectively. When $d_{\mathcal{E}} \in [-d_{\mathcal{E}t}, d_{\mathcal{E}t}]$, the plane \mathcal{L} and the ellipsoid \mathcal{E} have at least one intersection, thus violating our desired reachability constraint.

With these definitions, the constraint can be written as:

ADMM constraint 6 (Plane-ellipsoid reachability constraint).

$$D(\mathbf{q}) = H^{-1}(q_1, q_2) \pi_{\mathbf{n}_{\mathcal{E}}}^{\mathcal{E}}(\mathbf{q}) + o, \quad (44)$$

$$\mathcal{Z} = \{\mathbf{q} \in \mathbb{R}^{nm} : \|D_{\mathbf{n}}(\mathbf{q})\| = 0\}, \quad (45)$$

$$\Pi_{\mathcal{Z}}(\mathbf{z}) = \vec{0}, \quad (46)$$

where $\pi_{\mathbf{n}_{\mathcal{E}}}^{\mathcal{E}}(q)$ is the projection operator defined as:

$$\pi_{\mathbf{n}_{\mathcal{E}}}^{\mathcal{E}}(\mathbf{q}) = \begin{cases} p_{t_1}^{\mathcal{E}} - p_{\mathcal{L}}^{\mathcal{E}} & \text{if } d_{\mathcal{E}} \in [0, d_{\mathcal{E}t}], \\ p_{t_2}^{\mathcal{E}} - p_{\mathcal{L}}^{\mathcal{E}} & \text{if } d_{\mathcal{E}} \in [-d_{\mathcal{E}t}, 0), \\ 0 & \text{otherwise.} \end{cases} \quad (47)$$

Proposition 4. The differential of the projection function $\pi_{\mathbf{n}_{\mathcal{E}}}^{\mathcal{E}}(\mathbf{q})$ with respect to the foci q_1 and q_2 is given by:

$$\partial_q \pi_{\mathbf{n}_{\mathcal{E}}}^{\mathcal{E}}(q) = \begin{cases} \partial_q p_{t_1}^{\mathcal{E}} - \partial_q p_{\mathcal{L}}^{\mathcal{E}} & d_{\mathcal{E}} \in [0, d_{\mathcal{E}t}], \\ \partial_q p_{t_2}^{\mathcal{E}} - \partial_q p_{\mathcal{L}}^{\mathcal{E}} & d_{\mathcal{E}} \in [-d_{\mathcal{E}t}, 0), \\ 0 & \text{otherwise.} \end{cases} \quad (48)$$

where

$$\begin{aligned} \partial_q p_{\mathcal{L}} = & \left(-\frac{d_{\mathcal{E}t} n^T \partial_q o - 2d_{\mathcal{E}} \partial_q d_{\mathcal{E}t}}{d_{\mathcal{E}t}^3} \right) Q^{-1} n_{\mathcal{E}} \\ & + \frac{d_{\mathcal{E}} \partial_b Q^{-1} n_{\mathcal{E}} \partial_q b - 2d_{\mathcal{E}} Q^{-1} H[n]_{\times} U}{d_{\mathcal{E}t}^2}, \end{aligned} \quad (49)$$

$$\partial_q p_1 = -\frac{Q^{-1} n_{\mathcal{E}} \partial_q d_{\mathcal{E}t}}{d_{\mathcal{E}t}^2} + \frac{\partial_b Q^{-1} n_{\mathcal{E}} \partial_q b - 2Q^{-1} H[n]_{\times} U}{d_{\mathcal{E}t}}. \quad (50)$$

Proof. See Appendix D. \square

Based on the Proposition 4, the differential of (44) can be written as:

$$\partial_q D_{\mathbf{n}_{\mathcal{E}}} = -2H[\Pi_{\mathbf{n}_{\mathcal{E}}}^{\mathcal{E}}]_{\times} M + H^{-1} \partial_q \Pi_{\mathbf{n}_{\mathcal{E}}}^{\mathcal{E}} \quad (51)$$

I. Line-segment-ellipse reachability constraint

For an intermediate step to get the relationship between a polygon shaped forbidden region, the relative position between the ellipse and segment of the hyperplane that defines the region is studied. Assume the segment on hyperplane $\mathcal{L}^{\mathcal{E}}(q^{\mathcal{E}}) = \{q^{\mathcal{E}} \in \mathbb{R}^m : \mathbf{n}_{\mathcal{E}}^T q^{\mathcal{E}} = d_{\mathcal{E}}\}$ has endpoints $p_1^{\mathcal{E}}$ and $p_2^{\mathcal{E}}$; then the segment can be defined as:

$$\begin{bmatrix} (p_1^{\mathcal{E}} - p_2^{\mathcal{E}})^T \\ (p_2^{\mathcal{E}} - p_1^{\mathcal{E}})^T \end{bmatrix} p^{\mathcal{E}} \leq \begin{bmatrix} p_2^{\mathcal{E}T} \\ -p_1^{\mathcal{E}T} \end{bmatrix} (p_1^{\mathcal{E}} - p_2^{\mathcal{E}}), \quad \mathbf{n}_{\mathcal{E}}^T q^{\mathcal{E}} = d_{\mathcal{E}}. \quad (52)$$

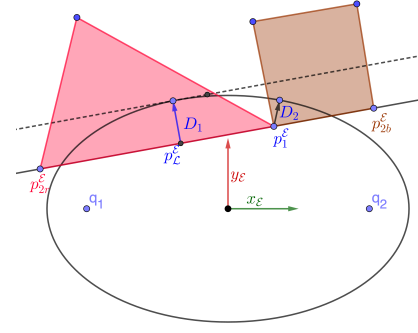


Fig. 6: Polygon-ellipse constraint. This constraint is simplified to either a plane-ellipse constraint D_1 or a point-ellipse constraint D_2 .

For cases, where the *tangent interpolation point* $p_{\mathcal{L}}^{\mathcal{E}}$ stays within the segment (i.e. red region in Figure 6, where $p_{\mathcal{L}}^{\mathcal{E}}$ lies between $p_1^{\mathcal{E}}$ and $p_2^{\mathcal{E}}$), the constraint can be seen as a plane-ellipse constraint in Section II-H. Otherwise (i.e. brown region in Figure 6, where $p_{\mathcal{L}}^{\mathcal{E}}$ lies outside $p_1^{\mathcal{E}}$ and $p_2^{\mathcal{E}}$), the constraint can be seen as a point-ellipse constraint in Section II-G.

ADMM constraint 7 (Line-segment-ellipsoid reachability constraint).

$$D(\mathbf{q}) = \begin{cases} D_{p_1}(\mathbf{q}) & (p_1^{\mathcal{E}} - p_2^{\mathcal{E}})^T (p_{\mathcal{L}}^{\mathcal{E}} - p_2^{\mathcal{E}}) < 0 \\ D_{p_2}(\mathbf{q}) & (p_2^{\mathcal{E}} - p_1^{\mathcal{E}})^T (p_{\mathcal{L}}^{\mathcal{E}} - p_1^{\mathcal{E}}) < 0 \\ D_{p_{\mathcal{L}}^{\mathcal{E}}}(\mathbf{q}) & \text{otherwise} \end{cases} \quad (53)$$

$$\mathcal{Z} = \{\mathbf{q} \in \mathbb{R}^{nm} : \|D(\mathbf{q})\| = 0\}, \quad (54)$$

$$\Pi_{\mathcal{Z}}(\mathbf{z}) = 0, \quad (55)$$

where D_{p_1} and D_{p_2} are the point-ellipsoid constraint projection function with respect to $p_1^{\mathcal{E}}$ and $p_2^{\mathcal{E}}$ in (32), and $D_{p_{\mathcal{L}}^{\mathcal{E}}}$ is the plane-ellipsoid constraint with respect to frame $\mathcal{L}^{\mathcal{E}}$ in (44).

J. Convex-polygon-ellipse reachability constraint

To keep an ellipse away from a convex polygon, first, we need to keep all segments of the hyperplanes outside the ellipse. Similar to (43), we define

ADMM constraint 8 (Convex-polygon-ellipsoid reachability constraint).

$$D(\mathbf{q}) = \begin{bmatrix} D_{seg1}(\mathbf{q}) \\ D_{seg2}(\mathbf{q}) \\ \vdots \end{bmatrix} \quad (56)$$

$$\mathcal{Z} = \{\mathbf{q} \in \mathbb{R}^{nm} : \|D(\mathbf{q})\| = 0\}, \quad (57)$$

$$\Pi_{\mathcal{Z}}(\mathbf{z}) = 0, \quad (58)$$

where D_{seg} are the constraint functions for each line segment used to define convex polygon region. ADMM constraint 8 needs to be supplemented with a convex obstacle constraint for the polygon (as introduced in section II-C) to prevent cases where the ellipse is a subset of the region.

K. Secured planning results and limitation

In this section, we apply our ADMM path planning algorithm to an instance of a map exploration problem (Figure 2a), and test the algorithm in both simulations and an experimental testbed. The environment to be explored, with three agents, is a $10m \times 10m$ region and two forbidden regions *Zone 2*, *Zone 3* which is shown in Fig.2b. We assume that the robots can meaningfully collect information on the underlying vector field for data locations no further than $3m$. The maximum velocity constraint v_{max} is set to $0.5m/dt$ a time limit of 20.

We utilize the APMAPF solver introduced in [5] to generate a MAPF plan with co-observation schedule on a grid-world application (Fig.2b) and transform the result to a continuous configuration space, under the map exploration tasks. All robots have the task of collecting sensor information on the underlying vector field, with agent 3 having an additional task to visit the *checkpoint* at time 8. Assuming that the sensors on the robots return data with higher accuracy for locations closer to the agent, the robots should ideally perform a boustrophedon pattern (like the unconstrained result in Figure 2a). We first set up an co-observation schedules considering two forbidden regions using the APMAPF algorithm, and add reachability constraints ensure a empty intersection between all robots' reachability regions between co-observations and forbidden regions.

As shown in Fig.2b, no task has been assigned to the agent besides the start and goal location. Agent 1 and 2 need to meet at time 8 and 14, whereas agent 2 and 3 need to meet at time 18. This result is also used as the initial trajectory input for the ADMM solver.

Based on the schedule, reachability for agent 1 in-between the scheduled co-observations are constrained. Notice that additional checkpoint (i.e. stationary security camera) need to be incorporated in order to generated an attack-proof solution. Additional solutions to avoid these checkpoints will be discussed in Section III.

The result of the simulation is shown in Fig. 7. The reachability regions are shown as black ellipsoids, and we can show that all of them have empty intersections with Zone 2 and 3. Notice that no explicit constraints have been enabled between reachability regions and obstacles, because we assume that robots have the basic obstacle avoidance capability and can not go through any hard obstacles. Therefore, the intersections between obstacles and ellipsoids, i.e. the case here between agent 3 and zone 1, are tolerable. All constraints have been satisfied and, to the best of their capability, agents have spread out across the map to perform the best possible exploration task.

1) *Limitations:* Our solution shows that planning with reachability and co-observation is a promising approach to enhance the security of multi-agent systems. However, two main challenges have been identified in [7]. Firstly, finding a co-observation and reachability secured plan may not always be feasible when robots are separated from others by obstacles or forbidden regions, or are located far away from each other. In such case, it is impossible for other robots to get close enough to establish co-observation schedules or find a reachability-secured path. Agent 3 in the Figure 7 is a good example, as

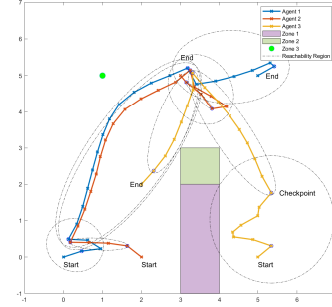


Fig. 7: Simulation result for map exploration task. Reachability regions are shown as black ellipses in the result. Zone 1 is obstacle, Zone 2 and Zone 3 are safe zones

it required additional security checkpoint to create secured reachability areas. Secondly, security requirements may come at the cost of overall system performance, as illustrated by the comparison of Figure 2a and 7 where, after the introduction of security constraints, the top left corner is never explored by any robots. This is particularly concerning given that system performance is a key factor in the decision to use multi-agent systems. These challenges are addressed in Chapter III, in order to ensure the effectiveness of planning with reachability and co-observation in securing multi-agent systems.

III. CROSS-TRAJECTORY CO-OBSERVATION PLANNING

To address the feasibility and performance trade-offs of the ADMM problem discussed in Section II-K1, we propose to form *sub-teams* on each route, and setup additional co-observations both within the sub-team and across different sub-teams (as shown in Figure 1). These *cross-trajectory co-observations* allow sub-teams to obtain trajectories that are closer to the optimal (as shown in Figure 1), because they do not require the entire *sub-team* to meet with other teams for co-observations, thus providing more flexibility.

Technically, we can always create a trivial schedule that involves only co-observations between members of the same team; this, however, would make the solution more vulnerable in the case where multiple agents are compromised in the same team. While in this paper we explicitly consider only the single attacker scenario, multi-attacks can be potentially handled by taking advantage of the *decentralized blocklist protocol* introduced in [6]. For this reason, we setup the methods presented below to always prefer *cross-trajectory co-observation* when feasible.

A. Problem overview

To solve the infeasibility problem like the cases shown in Figure 7 where a secured co-observation schedule cannot be found without external observation approach (Checkpoint at the bottom right of the map), we first use planner (in this paper, an ADMM-based planner introduced in Section II is used, but other planners are applicable) to generate n_p routes $\{q_p\}_{p=1}^{N_p}$ without reachability and co-observation constraints. Here, the

state $q_p(t), p \in \{\mathcal{I}_0, \dots, \mathcal{I}_p\}$ represents the reference position of the i -th sub-team at time $t \in \{0, \dots, T\}$. Instead of having the number of robot equal to the number of routes n_p , in this section, we assume a total of $n > n_p$ robots are available and organized into *sub-teams* through a time-varying partition $\mathcal{I}(t) = \cup_p \mathcal{I}_p(t)$ where robots in each *sub-team* \mathcal{I}_p share the same nominal trajectory. We then capitalize on the presence of additional $n_p - n$ robots to enhance security.

In scenarios where at least one robot is mandated to adhere to the reference trajectory to fulfill essential tasks, these additional robots' initial focus are on performing co-observations within their respective sub-teams. Whenever necessary, they are designed to seamlessly transition between sub-teams. By deviating from their original sub-team's route and joining teams requiring heightened attention, they become instrumental in providing co-observations at critical junctures.

We refer to the new type of co-observations as *cross-trajectory co-observations*. Our objective is to formulate a cross-trajectory co-observation strategy that fortifies an unsecured Multi-Robot System (MRS) trajectory, while minimizing the number of additional robots required.

To solve this problem, we model the planned trajectory $\{q_p\}_{p=1}^{N_p}$ as a directed *checkpoint graph* $G_q = (V_q, E_q)$. Vertices V_q are a key locations in $\{q_p\}$ that requires attention. The set $E_q = E_t \cup E_c$ consists of directed edges connecting V_i to V_j if there exist possible paths between the corresponding locations. Inspired by [15], we show that additional robots in the sub-team can be formulated as flows in the checkpoint graph, transforming the co-observation planning problem into a network multi-flow problem that can be solved using general linear programming techniques as specialized solver.

This section presents the two components of our approach: constructing the checkpoint graph based on unsecured multi-robot trajectories, and the formulation and solution of the network multi-flow problem.

B. Rapidly-exploring Random Trees

In this section, we use the RRT* [16] algorithm to find paths from a single starting location to multiple destination points (i.e., the reference trajectories of other robots, in our method).

As a optimal path planning algorithm, RRT* returns the shortest paths between an initial location and points in the free configuration space, organized as a tree. We assume that the generated paths can be travelled in both directions (this is used later in our analysis). Key functions from RRT* that are also used during constructions of the checkpoint graph are

Cost(v) This function assigns a non-negative cost (total travel distance in our application) to the unique path from the initial position to v .

Parent(v) This is a function that maps the vertex v to $v' \in V$ such that $(v', v) \in E$.

Notice that our objective is not to optimize specific constraints but to ascertain the existence of feasible paths. While the ADMM based solver Section II presented earlier offers a broader range of constraint handling, the RRT* algorithm is better suited to the problem in this section. RRT*'s efficient exploration of the solution space, coupled with its ability to

incorporate obstacle constraints, makes it a fitting choice for building the checkpoint graph.

C. Checkpoint graph construction

In this section, we describe in detail how we define and search for security checkpoints and how to use RRT* to construct the checkpoint graph G_q .

1) *Checkpoints*: Let $q_p \in \mathbb{R}^{nT}$ be the MRS trajectory for sub-team p with time horizon T , with $q_p(t) \in \mathbb{R}^n$ be the reference waypoint at time t .

Definition 7. Checkpoint $v_{pi} = (q_{pi}, t_i)$ is defined as a pair composed of a location $q_{pi} = q_p(t_i)$ and a time t_i representing a co-observation required for sub-team p .

For convenience, let \mathcal{I}_{v_i} represent the sub-team that v_i belongs to, **zyang** and let $q_{v_{pi}} = q_{pi}, t_{v_{pi}}$ be the corresponding waypoint and time for checkpoint v_{pi} **Is this necessary?**

Using the idea of co-observation and reachability constraints introduced in Section II, we define a set of checkpoints for the trajectory of sub-team p to provide security. **zyang** definition or remark?

Definition 8. The set of $V_p = \{v_{p0}, \dots, v_{pT}\}$ (arranged in ascending order of $t_{v_{pi}}$) that can be used to secure the reference trajectory for sub-team p , if the reachability region between consecutive checkpoints v_{pi} and $v_{p(i+1)}$ does not intersect with any of the forbidden areas. The requirement can be formally stated as $\mathcal{E}(q_{v_{pi}}, q_{v_{p(i+1)}}; t_{v_{pi}}, t_{v_{p(i+1)}}) \cap F = \emptyset$ for every i , where F is the union of all forbidden areas as F .

We provide a heuristic approach to locate the checkpoints on given trajectories (an optimal solution would likely be NP-hard, while the approach below works well enough for our purpose). The approach is shown in Algorithm 1, and a toy example is shown in Figure 8.

Algorithm 1 Secure Checkpoint Generate for sub-team p

```

 $v_{p0} \leftarrow (q_p(0), 0); v_{pT} \leftarrow (q_p(T), T)$ 
 $V_p = \{v_{p0}, v_{pT}\}$ 
 $t_0 \leftarrow 0; t_1 \leftarrow T$ 
while  $\mathcal{E}(q_p(t_0), q_p(t_1), t_0, t_1) \cap F \neq \emptyset$  or  $t_1 - t_0 > 1$  do
   $i \leftarrow 1; j \leftarrow 1$ 
  while  $\mathcal{E}(q_p(t_0), q_p(t_0 + i), t_0, t_0 + i) \cap F = \emptyset$  and  $t_0 + i > t_1$  do
     $v_{pf} \leftarrow (q_p(t_0 + i), t_0 + i)$ 
     $i \leftarrow i + 1$ 
  end while
   $V_p \leftarrow V_p \cup v_{pf}$ 
  while  $\mathcal{E}(q_p(t_1 - j), q_p(t_1), t_1 - j, t_1) \cap F = \emptyset$  and  $t_0 + i \geq t_1 - j$  do
     $v_{pb} \leftarrow (q_p(t_1 - j), t_1 - j)$ 
     $j \leftarrow j + 1$ 
  end while
  if  $t_0 + i \neq t_1 - j$  then
     $V_p \leftarrow V_p \cup v_{pb}$ 
  end if
end while

```

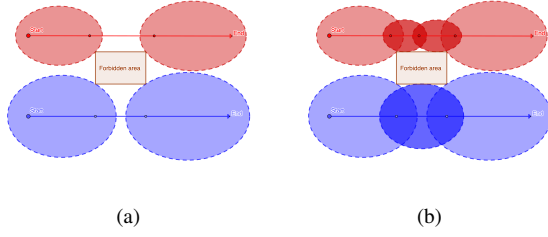


Fig. 8: Checkpoint generation example

2) *Cross-trajectory edges*: After identifying checkpoints $\cup_p V_p$ for all sub-teams, the next step is to search for cross-trajectory edges connecting all checkpoints to the respective trajectories. **rtron** Add definition here for edges to be the counterpart of Definition 8 These cross-trajectory edges define viable paths between two reference trajectories, allowing robots to closely approach and collaborate with robots on a different reference trajectory for co-observations with the relevant sub-teams. Consequently, at least one endpoint of each edge must correspond to a security checkpoint $v_p \in \cup_p V_p$. Assuming a single robot is transferred at a time between trajectories, the cross-trajectory edges must adhere to the reachability constraints detailed in Section II-F to ensure that no deviations into forbidden areas can occur during trajectory switches.

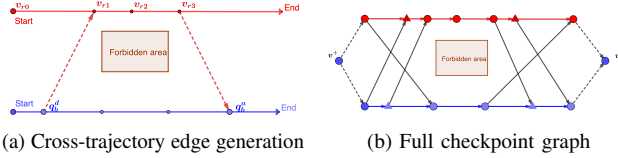


Fig. 9: Figure 9a, Latest departure node q_b^d found for v_{r1} and earliest arrival node q_b^a found for v_{r3} . Figure 9b, example of a full checkpoint graph where round vertices are checkpoint generated through the heuristic search and triangle vertices are added with the cross-trajectory edges. Virtual source v^+ and sink v^- are added to be used later in planning problem.

To find cross-trajectory edges, we first find trees of feasible paths using RRT* and position information alone; we then prune these trees by considering time constraints (that consider the time needed to physically travel from one trajectory to the other while meeting other robots at the two endpoints) and ellipsoidal reachability constraints.

We search for feasible paths between each security checkpoint and all other trajectories (ignoring, for the moment, any timing constraint), as shown in Figure 10 More precisely, after a fixed number of iterations, RRT* finds all feasible, quasi-optimal paths between one security checkpoint location $q_{v_p}(t_{v_p})$ where $v_p \in V_p$ for sub-team \mathcal{I}_p , and all the waypoints $\{q_r(t_{r_i})\}$ on any reference trajectory for a different sub-team \mathcal{I}_r . We then calculate the minimal travel time $t_{\text{path}} = \text{Cost}(q)/v_{\text{max}}$ for a robot to traverse each path. In order to see if the deviating robot is able to catch up and merge with the corresponding sub-team, we define two types of nodes.

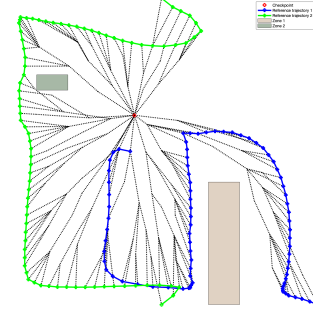


Fig. 10: All viable path found from a waypoint to all reference trajectories of other sub-teams.

Arrival nodes are waypoints on $\{q_r\}$ where robots from sub-team \mathcal{I}_p , deviating at v_p , can meet with sub-team \mathcal{I}_r at $(q_r(t_{r_i}), t_{r_i})$. For these, RRT* must have found a path from v_p to $q_r(t_{r_i}^a)$ with $t_p + t_{\text{path}} < t_{r_i}$, and $\mathcal{E}(q_r(t_{r_i}), q_p, t_{r_i}, t_p) \cap F = \emptyset$.

For each trajectory $r \neq p$, we define the *earliest arrival node* from v_p to $(q_r(t_{r_i}^a), t_{r_i}^a)$ as the arrival node characterized by the minimum t_r^a discovered.

Departure nodes are the waypoints on $\{q_r\}$ such that a robot from sub-team \mathcal{I}_r , deviating from $(q_r(t_{r_i}), t_{r_i})$, can meet with robots in sub-team \mathcal{I}_p at v_p . For these nodes, RRT* must find a path from v_p to $q_r(t_{r_i})$ for v_p if $t_p > t_{r_i} + t_{\text{path}}$ and $\mathcal{E}(q_p, q_r(t_{r_i}), t_p, t_{r_i}) \cap F = \emptyset$.

For each trajectory $r \neq p$, we define the *latest departure node* $(q_r(t_{r_i}^d), t_{r_i}^d)$ as the departure node characterized by the maximum t_r^d discovered.

For each $v_p \in V_p$ found through Algorithm 1, we add all the latest departure and earliest arrival nodes to V_q , and their corresponding paths as cross-trajectory edges E_q . Examples are shown in Figure 9a.

3) *In-trajectory edges*: We arrange all $V_p = \cup_p \{v_p^0, \dots, v_p^T\}$ found in Sections III-C1 and III-C2 in ascending order of $t_{v_p^i}$. We then add to E_q the *in-trajectory edges* $\{(v_p^i \rightarrow v_p^{i+1})\}$ obtained by connecting all consecutive checkpoints $v_p^i \in V_q$ with the checkpoints v_p^{i+1} that follow them in their original trajectories. Examples are shown in Figure 9b.

D. Co-observation planning problem

In this section, we describe in detail how to formulate the cross-trajectory planning problem as a network multi-flow problem, and solve it using mixed-integer linear programs (MILP). We assume that n_p reference robots are dedicated (one in each sub-team) to follow the reference trajectory. The goal of this section is to minimize the number and plan the routes of the $n - n_p$ additional cross-trajectory robots dedicated to cross-trajectory co-observations.

Remark 2. Note that we assign fixed roles to reference robots for convenience in explaining the multi-flow formulation. In practice, after a cross-trajectory robot joins a team, it is considered interchangeable, and could switch roles with the

reference robot of that trajectory (i.e., the original reference robot might serve as a cross-trajectory robot in a later cross-trajectory co-observation).

To formulate the planning problem as a network multi-flow problem, we augment the checkpoint graph G_q to a flow graph $G = (V, E)$. The vertices of the new graph G are defined as $V = V_q \cup \{v^+, v^-\}$, where v^+ is a *virtual source* node, and v^- is a *virtual sink* node. The edges of the new graph are defined as $E = E_q \cup \{(v^+, v_p^0)\}_p \cup \{(v_p^T, v^-)\}_p$, i.e., we add directed edges from v^+ to all the start vertices, and from all end vertices to v^- , with v_p^0 and v_p^T representing the start and end vertices of sub-team \mathcal{I}_p .

Our goal is to find a path in G for all the cross-trajectory robots that starts from the virtual source v^+ and ends in the virtual sink v^- . The path of a robot k can be represented as a flow vector $\mathbf{f}^k = \{f_{ij}^k\}$, where $f_{ij}^k \in \{1, 0\}$ is an indicator variable representing whether robot k 's path contains the edge $v_i \rightarrow v_j$. In order to perform co-observations as required by the security introduced in Remark 1, reference robots must be seen by at least one cross-trajectory robot at every checkpoint. Thus, the planning problem can be formulated as a path cover problem on G_q , i.e., as finding a set of paths $F = [\mathbf{f}_1, \dots, \mathbf{f}_K]$ for cross-trajectory robots such that every checkpoint in V_p is included in at least one path in F . Additionally, to encourage the robots' exchange between different sub-teams, cross-trajectory co-observations should be preferred compared to co-observation within the same team.

rtrn Move remark about the distributed blocklist here

Finally, edges from the virtual source and to the virtual sink should have zero cost, to allow robots to automatically get assigned to the starting point that is most convenient for the overall solution. These requirements are achieved with the weights for edges $(v_i, v_j) \in E$ defined as:

$$w_{i,j} = \begin{cases} -w_t & \mathcal{I}_{v_i} = \mathcal{I}_{v_j}, (v_i, v_j) \in E_q \\ w_c & \mathcal{I}_{v_i} \neq \mathcal{I}_{v_j}, (v_i, v_j) \in E_q \\ 0 & (v_i, v_j) \in E/E_q \end{cases} \quad (59)$$

where $w_c > w_t$.

With the formulation above, the planning problem can be written as an optimization problem, where the optimization cost balances between the co-observation performance and total number of flows (cross-trajectory robots) needed:

$$\min_F \sum_k \sum_{(+i) \in E} f_{+i}^k - \rho \sum_k \sum_{(ij) \in E} w_{ij} f_{ij}^k \quad (60a)$$

$$\text{s.t.} \quad \sum_{\{h:(hi) \in E\}} f_{hi}^k = \sum_{\{j:(ij) \in E\}} f_{ij}^k, \forall k, \forall v \in V_q / \{v^+, v^-\} \quad (60b)$$

$$\sum_k \sum_{\{i:(ij) \in E\}} f_{ij}^k \geq 1, \forall v_j \in \{V_p^s\} \quad (60c)$$

$$f_{ij}^k \in \{0, 1\} \forall (ij) \in E \quad (60d)$$

where, for convenience, we used (ij) to represent the edge (v_i, v_j) , and $(+i)$ to represent the edge (v^+, v_i) .

The first term in the cost 60a is total outgoing flow from the source v^+ while the second term is the total cost of all flows

representing the overall co-observation performance (defined as the total number of cross-trajectory edges taken by all flows beyond the regular trajectory edges); the constant ρ is a manually selected penalty parameter to balance between the two terms. The constraint (60b) is the flow conservation constraint, which ensures that the amount of flow entering and leaving a given node v is equal (except for v^+ and v^-). The constraint (60c) is the flow coverage constraints, which ensures that all security checkpoints $\{V_p^s\}$ have been visited by at least one flow (robot). Since the security graph G_q is acyclic, this problem is in complexity class P , and can be solved in polynomial time [17].

E. Co-observation performance

Notice that problem (60) is guaranteed to have a solution for $K = N_p$ where all K robots follows the reference trajectory ($f_{ij}^k = 1, \forall \mathcal{I}_{v_i} = \mathcal{I}_{v_j} = k$). Additionally, for a fixed number of robots $K \geq N_p$, it is possible to have a subset of flows $F_e \in F$ that is empty, i.e. $f_{ij} = 0, \forall (v_i, v_j) \in E, f \in F_e$; these flow will not increase the cost and can be discarded from the solution.

The constant ρ selects the trade-off between the number of surveillance robots and security performance. An increase in robots generally enhancing security via cross-trajectory co-observations; this also increases the complexity of the coordination across robots.

Remark 3. The value of row ρ is upper bounded such that the second term for each single flow in (60a) is always smaller than one, i.e., $\rho \sum_{(ij) \in E} w_{ij} f_{ij}^k \leq 1$. This is because, otherwise

rtrn ... explain why the optimal cost would be unbounded.

In order to identify the minimum number of robots necessary, we propose an iterative approach where we start with $K = 1$ and gradually increases it F contains an empty flow, indicating the point where further robot additions do not improve performance.

F. Result and simulation

In this section, we test the proposed method starting from the results in Figure ??, where three trajectories are provided for the map exploration task with no security related constraints (co-observation schedule and reachability). We use the same environment and dynamic setup, where we have a $10m \times 10m$ task space, two forbidden regions (two rectangles in ??) and robots with a max velocity of $0.5m/dt$.

We start by adding a total of $K = 3$ surveillance robots into the workspace using the parameters $w_c = 10$, $w_t = 1$ and $\rho = 0.01$. The three trajectories are transformed into the graph G shown in Figure 11, where the number on each vertex v_i represents the corresponding time t_i . Vertices in each column belong to the same trajectory, and edges across different columns are cross-trajectory edges.

The flows derived from the solution of the optimization problem (60) are highlighted in the graph. The planning result in the workspace is shown in Figure 12 as dash-dotted arrows with the same color used for each flow in Figure 11. This

rtron Collect these four figures into a single figure. Maybe plot the graphs rotated 90 degrees so that "time" goes from left to right, and to use the space more efficiently.

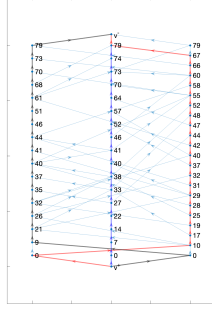


Fig. 11: Security Graph and result for 3 flows

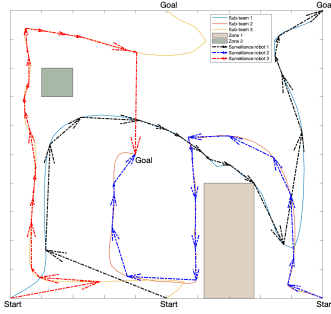


Fig. 12: Result of 3 surveillance agents' plan in workspace

plan is not ideal in terms of our security criteria since co-observations happens between the same pair of surveillance robot and sub-team for the majority of the time with only three cross-trajectory edges that are covered.

For the case where $\mathcal{K} = 4$, the results are shown in Figure 13-14. With one additional robot added, there is a significant increase in the total cross-trajectory edges covered, and sub-teams performs co-observation with different robots during the task period. If we further increase $\mathcal{K} > 4$, we do not get a better result; instead, the planner will generate four flows with the rest $\mathcal{K} - 4$ flows empty. **rtron** Reference Fig. 15

rtron Run the ADMM planner again on Fig. 14, and show that you get a $\Phi(q)$ (i.e., better mapping).

IV. SUMMARY

In this paper, we provide a method to enhance the security of a multi-robot system without sacrificing the performance. This is done by introducing additional robots to perform cross-trajectory co-observations by traveling between different trajectories. For cases where the secured planning problem is infeasible (or where the user desires to improve the optimality of the trajectories), we model the unsecured multi-robot trajectories as a checkpoint graph by identifying checkpoints that requires observation and cross-trajectory paths that can

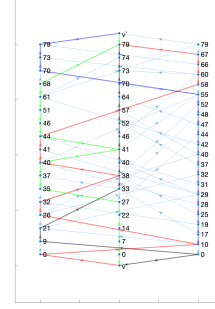


Fig. 13: Security Graph and result for 4 flows.

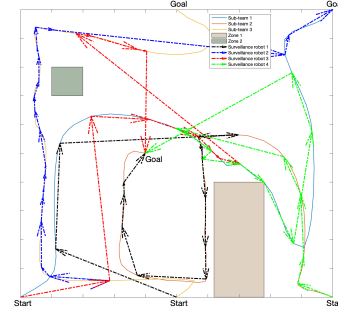


Fig. 14: Result of 3 surveillance agents' plan in work space.

safely access the checkpoints from different trajectories. We have shown that the co-observation planning problem across different trajectories can be transformed into a multi-flow problem on the graph, and that we can find the minimal number of robot needed to finish the co-observation task.

APPENDIX

A. Proof of proposition 1

For subclaim 1)a:

$$H^T H = H^2 = 4uu^T uu^T - 4uu^T + I^2 = I, \quad (61)$$

since $u^T u = 1$.

For subclaim 1)b, let $U = [u \ u_1^\perp \ u_2^\perp]$, where u_1^\perp, u_2^\perp are two orthonormal vectors such that $I = UU^T = uu^T + u_1^\perp (u_1^\perp)^T + u_2^\perp (u_2^\perp)^T$; then, substituting I in (4), we have that the eigenvalue decomposition of H is given by

$$H = U \text{diag}(1, -1, -1)U^T. \quad (62)$$

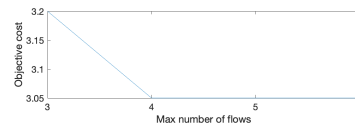


Fig. 15: Plan reach optimality when $\mathcal{K} = 4$, further increase of \mathcal{K} does not increase the cost of objectives.

Since the determinant of a matrix is equal to the product of the eigenvalues, $\det(H) = 1$.

For subclaim 2), first note that $Hu = 2uu^T u - u = u$. It follows that the sum of $\nu_{\mathcal{F}}$ and $\nu_{\mathcal{E}}$ is invariant under H :

$$H(\nu_{\mathcal{F}} + \nu_{\mathcal{E}}) = Hu \| \nu_{\mathcal{F}} + \nu_{\mathcal{E}} \| = u \| \nu_{\mathcal{F}} + \nu_{\mathcal{E}} \| = \nu_{\mathcal{F}} + \nu_{\mathcal{E}}, \quad (63)$$

and that their difference is flipped under H :

$$H(\nu_{\mathcal{F}} - \nu_{\mathcal{E}}) = 2uu^T(\nu_{\mathcal{F}} - \nu_{\mathcal{E}}) - (\nu_{\mathcal{F}} - \nu_{\mathcal{E}})^2 = -(\nu_{\mathcal{F}} - \nu_{\mathcal{E}}). \quad (64)$$

Combining (63) and (64) we obtain

$$H\nu_{\mathcal{F}} = \frac{1}{2}(H(\nu_{\mathcal{F}} + \nu_{\mathcal{E}}) + H(\nu_{\mathcal{F}} - \nu_{\mathcal{E}})) = \nu_{\mathcal{E}} \quad (65)$$

B. Proof of proposition 2

From the definition of H in (4), we have

$$\dot{H} = 2(\dot{u}u^T + u\dot{u}^T) \quad (66)$$

Recall that $\dot{u} = \frac{1}{\|u'\|}(I - uu^T)\dot{u}'$ (see, for instance, [18]), which implies $(I - uu^T)\dot{u}' = \dot{u}'$. It follows that \dot{u} flips sign under the action of H^T :

$$\begin{aligned} H^T \dot{u} &= (2uu^T - I) \frac{(I - uu^T)}{\|u'\|} \dot{u}' \\ &= \frac{1}{\|u'\|} (2uu^T - I - 2uu^T uu^T + uu^T) \dot{u}' \\ &= -\frac{1}{\|u'\|} (I - uu^T) \dot{u}' = -\dot{u} \end{aligned} \quad (67)$$

Inserting $HH^T = I$ in (66), and using ??, we finally have

$$\begin{aligned} \dot{H} &= 2HH^T(\dot{u}u^T + u\dot{u}^T) = 2H(-\dot{u}u^T + u\dot{u}^T) \\ &= -2H[[u]_{\times} \dot{u}]_{\times} \\ &= -2H \left[[u]_{\times} \frac{(I - uu^T)(I - \nu_{\mathcal{F}}\nu_{\mathcal{F}}^T)}{\|u'\| \|\nu_{\mathcal{F}}\|} \dot{\nu}_{\mathcal{F}} \right]_{\times} \\ &= -2H[M\dot{\nu}_{\mathcal{F}}]_{\times}, \end{aligned} \quad (68)$$

which is equivalent to the claim.

C. Proof of proposition 3

To make the notation more compact, we will use $\partial_q f$ instead of $\partial_{[q_1]} f$ for the remainder of the proof. The differential of (39) can be represented as:

$$\begin{aligned} \dot{\pi}_{p\mathcal{E}} &= \dot{H}^{-1}SH(q_{\text{avoid}} - o) + H^{-1}\dot{S}H(q_{\text{avoid}} - o) \\ &\quad + H^{-1}S\dot{H}(q_{\text{avoid}} - o) + (H^{-1}SH - I)\dot{o} \end{aligned} \quad (69)$$

where

$$\begin{aligned} \dot{S} &= -S^2(Q\dot{s} + s\dot{Q}) \\ &= -S^2(Q\partial_q \dot{q} - \partial_b Q \partial_q b \dot{q}) \end{aligned} \quad (70)$$

where

$$\partial_b Q = 2\frac{s}{b^3} \text{diag}\{0, 1, 1\} \quad (71)$$

To compute the derivative $\partial_q \pi$, we need the expression of $\partial_q b$, $\partial_q o$ and $\partial_q s$; the first two can be easily derived using the equations above:

$$\partial_q b = \frac{1}{4b} [q_1 - q_2, q_2 - q_1]^T \quad (72)$$

$$\partial_q o = [I/2, I/2]^T \quad (73)$$

In order to get $\partial_q s$, we use the fact that $F(s(q)) = 0$ for all q ; hence $F(\tilde{q}(t)) \equiv 0$, and $\partial_q F = 0$. We then have:

$$0 = \dot{F} = 2q^T Q' \dot{q} + q^T \partial_s Q' q \dot{s} + q^T \partial_b Q' q \dot{b} \quad (74)$$

where

$$\partial_s Q' = -\text{diag} \left(\frac{2a^2}{(s+a^2)^3}, \frac{2b^2}{(s+b^2)^3}, \frac{2b^2}{(s+b^2)^3} \right). \quad (75)$$

By moving term \dot{s} to the left-hand side we can obtain:

$$\begin{aligned} \dot{s} &= (q^T \partial_s Q' q)^{-1} (2q^T Q' \dot{q} + q^T \partial_b Q' q \dot{b}) \\ &= (q^T \partial_s Q' q)^{-1} (-4q^T Q' H[U\dot{q}]_{\times} (q_{\text{avoid}} - o) \\ &\quad - 2q^T Q' H\dot{o} + q^T \partial_b Q' q \dot{b}) \\ &= (q^T \partial_s Q' q)^{-1} (-4q^T Q' H[q_{\text{avoid}} - o]_{\times} U\dot{q} \\ &\quad - 2q^T Q' H\dot{o} + q^T \partial_b Q' q \dot{b}) \end{aligned} \quad (76)$$

The second term of equation (69) turns into:

$$\begin{aligned} H^{-1}\dot{S}H(q_{\text{avoid}} - o) &= -H^{-1}Q' q \dot{s} - sH^{-1}S^2 \partial_b Q q \dot{b} \\ &= ((q^T \partial_s Q' q)^{-1} H^{-1}Q' q q^T (4Q' H[q_{\text{avoid}} - o]_{\times} U \\ &\quad + 2Q' H\partial_q o - \partial_b Q' q q \partial_q b) - sH^{-1}S^2 \partial_b Q q \partial_q b) \dot{q} \end{aligned} \quad (77)$$

Thus equation (69) could be written as:

$$\begin{aligned} \dot{\pi}_{p\mathcal{E}} &= (-2H[SH(q_{\text{avoid}} - o)]_{\times} U \\ &\quad + ((q^T \partial_s Q' q)^{-1} H^{-1}Q' q q^T (4Q' H[q_{\text{avoid}} - o]_{\times} U \\ &\quad + 2Q' H\partial_q o - \partial_b Q' q q \partial_q b) - sH^{-1}S^2 \partial_b Q q \partial_q b) \\ &\quad - 2H^{-1}SH[q_{\text{avoid}} - o]_{\times} U \\ &\quad + (H^{-1}SH - I)\partial_q o) \dot{q}, \end{aligned} \quad (78)$$

from which the claim follows.

D. Proof of proposition 4

We first need to derive $\dot{d}_{\mathcal{E}}$ and $\dot{d}_{\mathcal{E}t}$

$$\dot{d}_{\mathcal{E}} = -n^T \partial_q o \dot{q} \quad (79)$$

$$\begin{aligned} \dot{d}_{\mathcal{E}t} &= (\dot{n}_{\mathcal{E}}^T Q^{-1} n_{\mathcal{E}} + n_{\mathcal{E}}^T \dot{Q}^{-1} n_{\mathcal{E}} + n_{\mathcal{E}}^T Q^{-1} \dot{n}_{\mathcal{E}}) / \sqrt{n_{\mathcal{E}}^T Q^{-1} n_{\mathcal{E}}} \\ &= (\sqrt{n_{\mathcal{E}}^T Q^{-1} n_{\mathcal{E}}})^{-1} (-2n_{\mathcal{E}}^T H[Q^{-1} n_{\mathcal{E}}]_{\times} U \\ &\quad + n_{\mathcal{E}}^T \partial_b Q^{-1} n_{\mathcal{E}} \partial_q b - 2n_{\mathcal{E}} Q^{-1} H[n]_{\times} U) \dot{q} \end{aligned} \quad (80)$$

Next, we need to derive \dot{p}_{t1} , \dot{p}_{t2} and $\dot{p}_{\mathcal{L}}$. Since $p_{\mathcal{L}}$ could be written as

$$p_{\mathcal{L}} = \frac{d_{\mathcal{E}} Q^{-1} n_{\mathcal{E}}}{d_{\mathcal{E}t}^2}, \quad (81)$$

we have

$$\dot{p}_{\mathcal{L}} = \left(\begin{aligned} & \left(-\frac{d_{\mathcal{E}t} n^T \partial_q o - 2d_{\mathcal{E}} \partial_q d_{\mathcal{E}t}}{d_{\mathcal{E}t}^3} \right) Q^{-1} n_{\mathcal{E}} \\ & + \frac{d_{\mathcal{E}} \partial_b Q^{-1} n_{\mathcal{E}} \partial_q b - 2d_{\mathcal{E}} Q^{-1} H[n]_{\times} U}{d_{\mathcal{E}t}^2} \end{aligned} \right) \dot{q} \quad (82)$$

$$\dot{p}_1 = \left(\begin{aligned} & -\frac{Q^{-1} n_{\mathcal{E}} \partial_q d_{\mathcal{E}t}}{d_{\mathcal{E}t}^2} \\ & + \frac{\partial_b Q^{-1} n_{\mathcal{E}} \partial_q b - 2Q^{-1} H[n]_{\times} U}{d_{\mathcal{E}t}} \end{aligned} \right) \dot{q} \quad (83)$$

subtracting \dot{q} from (82) and (83), we can derive the result shown in (43)

REFERENCES

- [1] Gonzalo Pajares. Overview and current status of remote sensing applications based on unmanned aerial vehicles (uavs). *Photogrammetric Engineering & Remote Sensing*, 81(4):281–330, 2015.
- [2] Brian J Julian, Michael Angermann, Mac Schwager, and Daniela Rus. Distributed robotic sensor networks: An information-theoretic approach. *The International Journal of Robotics Research*, 31(10):1134–1154, 2012.
- [3] Martin Brunner, Hans Hofinger, Christoph Krauß, Christopher Roblee, P Schoo, and S Todt. Infiltrating critical infrastructures with next-generation attacks. *Fraunhofer Institute for Secure Information Technology (SIT), Munich*, 2010.
- [4] Conner Forrest. Robot kills worker on assembly line, raising concerns about human-robot collaboration. <https://tinyurl.com/y2tzg4te>, March 2017.
- [5] Kacper Wardega, Roberto Tron, and Wenchao Li. Resilience of multi-robot systems to physical masquerade attacks. In *2019 IEEE Security and Privacy Workshops (SPW)*, pages 120–125. IEEE, 2019.
- [6] Kacper Wardega, Max von Hippel, Roberto Tron, Cristina Nita-Rotaru, and Wenchao Li. Byzantine resilience at swarm scale: A decentralized blocklist protocol from inter-robot accusations. In *Proceedings of the 2023 International Conference on Autonomous Agents and Multiagent Systems*, pages 1430–1438, 2023.
- [7] Kacper Wardega, Max von Hippel, Roberto Tron, Cristina Nita-Rotaru, and Wenchao Li. Hola robots: Mitigating plan-deviation attacks in multi-robot systems with co-observations and horizon-limiting announcements. *arXiv preprint arXiv:2301.10704*, 2023.
- [8] Ziqi Yang and Roberto Tron. Multi-agent trajectory optimization against plan-deviation attacks using co-observations and reachability constraints. In *2021 60th IEEE Conference on Decision and Control (CDC)*, pages 241–247. IEEE, 2021.
- [9] Ziqi Yang and Roberto Tron. Multi-agent path planning under observation schedule constraints. In *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 6990–6997. IEEE, 2020.
- [10] Jonathan D Gammell, Siddhartha S Srinivasa, and Timothy D Barfoot. Informed rrt*: Optimal sampling-based path planning focused via direct sampling of an admissible ellipsoidal heuristic. In *2014 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 2997–3004. IEEE, 2014.
- [11] Brian DO Anderson and John B Moore. *Optimal filtering*. Courier Corporation, 2012.
- [12] Alston S Householder. Unitary triangularization of a nonsymmetric matrix. *Journal of the ACM (JACM)*, 5(4):339–342, 1958.
- [13] Stephen Boyd, Neal Parikh, Eric Chu, Borja Peleato, Jonathan Eckstein, et al. Distributed optimization and statistical learning via the alternating direction method of multipliers. *Foundations and Trends® in Machine learning*, 3(1):1–122, 2011.
- [14] David Eberly. Distance from a point to an ellipse, an ellipsoid, or a hyperellipsoid, 2013.
- [15] Jingjin Yu and Steven M LaValle. Multi-agent path planning and network flow. In *Algorithmic Foundations of Robotics X: Proceedings of the Tenth Workshop on the Algorithmic Foundations of Robotics*, pages 157–173. Springer, 2013.
- [16] Sertac Karaman and Emilio Frazzoli. Incremental sampling-based algorithms for optimal motion planning. *Robotics Science and Systems VI*, 104(2):267–274, 2010.
- [17] S.C. Ntafos and S.L. Hakimi. On path cover problems in digraphs and applications to program testing. *IEEE Transactions on Software Engineering*, SE-5(5):520–529, 1979.
- [18] Roberto Tron and K. Daniilidis. Technical report on optimization-based bearing-only visual homing with applications to a 2-d unicycle model. 2014.