Cooperative Multi-Quadrotor Pursuit of an Evader in an Environment with No-Fly Zones

Alyssa Pierson[1], Armin Ataei[1], Ioannis Ch. Paschalidis[2], and Mac Schwager[3]

***Abstract*— We investigate the cooperative pursuit of an evader by a group of quadrotors in an environment with no-fly zones. While the pursuers cannot enter into no-fly zones, the evader may freely move through zones to avoid capture. Once the evader enters a no-fly zone, the pursuers calculate a reachable set of evader positions. Using tools from Voronoi-based coverage control applied to the evader's reachable set, we provide an algorithm that distributes the pursuers around the zone's boundary and minimizes the capture time once the evader leaves the no-fly zone. Robust model predictive control (RMPC) tools are used to control the quadrotors and to ensure that they always remain in free space. We demonstrate the performance of our proposed algorithms through a series of experiments on KMEL Nano+ quadrotors.**

I. Introduction

In this paper, we propose an algorithm for a group of quadrotors to cooperatively pursue an evader. The quadrotors move in an environment that has a set of "no-fly zones," which are regions that the quadrotors cannot enter, such as buildings or forests. The evader can freely move through these zones, while the pursuing quadrotors must position themselves around the boundary of the no-fly zones, ready to pursue the evader when it eventually emerges. Furthermore, the quadrotors can sense the position of the evader when it is in free space, but may not have information about the evader when it is inside a no-fly zone. Our algorithm uses Robust Model Predictive Control (RMPC) tools to always stay a prescribed safe distance away from the no-fly zones while pursuing the evader, and it adapts methods from Voronoi-based coverage control to position the quadrotors when the evader is inside a no-fly zone. We demonstrate our algorithm in hardware experiments with three quadrotors pursuing a manually-controlled ground robot.

Our algorithm is useful in a number of applications of emerging importance, such as search and rescue, robotic aerial videography, and security and surveillance. One example is where the evader is a suspected criminal fleeing the scene of a crime, and the pursuers are police surveillance drones. The pursuers track the suspect as it flees, but also must avoid buildings, bridges, trees and other environmental obstacles. If the suspect enters a building where the drones cannot follow, our algorithm will position the drones around the building so that they can continue tracking the suspect once it re-emerges, illustrated by Figure 1. Other settings where the algorithm may be useful include tracking a lost person or endangered animal in a park, or the subject of a sports film such as a snowboarder or mountain biker.

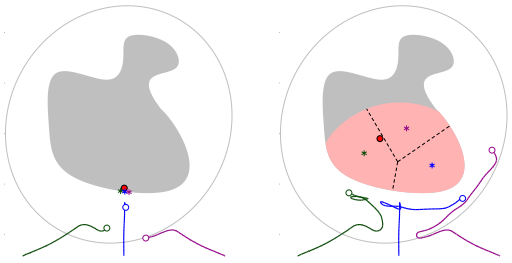


Fig. 1: Once an evader enters the no-fly region, the pursuers reposition themselves about the perimeter.

When the evader is in free space, the quadrotors pursue it directly using an RMPC controller to guarantee that they do not enter the no-fly zones. When an evader enters a no-fly zone, we adopt a three-part strategy for the quadrotors: (i) compute the reachable set of the evader, (ii) tessellate the reachable set with a centroidal Voronoi tessellation, then (iii) drive the quadrotors to points on the perimeter of the no-fly zone that are as close as possible to the centroids of the tessellation. The quadrotors may not have access to the evader's position inside the no-fly zone, so the evader's reachable set is a conservative estimate that grows in time, eventually filling up the no-fly zone. The RMPC control approach is again used to keep the quadrotors safely outside of the no-fly zones at all times as they navigate to their positions on the perimeter of the no-fly zone.

This work brings together elements of the authors' previous work in Robust Model Predictive Control for a single pursuer, and in Voronoi-based coverage control for sensor networks. In a previous work [1], we consider a single pursuer, and assume that once the evader enters a no-fly zone, the pursuer still has full information about its location. In contrast, here we consider multiple pursuers, and we assume that once the evader enters a no-fly zone, the pursuers may not have information about its position.

While there has been extensive research on path planning and obstacle avoidance for mobile robots [13], [10], most algorithms rely on simple approximations of the robot dynamics [20], [4], [17], [16]. If the underlying robot is highly nonlinear or is subject to uncertainties, these approximation may result in controller instability and collision with

This work was supported in part by ONR grants N00014-12-1-1000 and N00014-10-1-0952, by NSF grants CNS-1330036, IIS-1350904, CNS-1239021, and IIS-1237022, and by the ARO under grants W911NF-11-1-0227 and W911NF-12-1-0390. We are grateful for this financial support.

[1] Dept of Mechanical Engineering, Boston University, Boston, MA, {pierson,ataei}@bu.edu

[2] Dept of Electrical and Computer Engineering, Boston University, Boston, MA, yannisp@bu.edu

[3] Dept of Aeronautics and Astronautics, Stanford University, Stanford, CA, schwager@stanford.edu

obstacles. In [1], we proposed a robust MPC using linear matrix inequalities (LMIs) by extending the work of [12]. We showed robust performance could be achieved under modeling uncertainties and measurement noise. Furthermore, we proposed a path planning algorithm which combined with the RMPC technique could guarantee collision avoidance in presence of uncertainties.

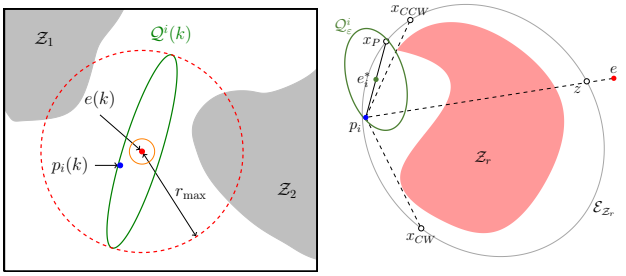
We also draw inspiration from Voronoi-based coverage control. A Voronoi-based coverage strategy first proposed by Cortés et al. [7], [6], often referred to as the move-to-centroid controller, drives all robots continuously towards the centroids of their Voronoi cells. This builds upon previous work in optimal location of retail facilities [8], and in data compression [9]. The Voronoi-based control strategy has also been used to track intruders by a team of robots within an environment [19], [14]. However, these works do not consider no-fly zones, and do not use the guaranteed safe RMPC tools that we use here. Other works [11], [18], [15] have used a Voronoi-based strategy for multi-agent pursuit-evasion, again without considering no-fly zones or RMPC control. Similarly to our approach for placing the pursuers around a no-fly zone, Susca et. al use a Voronoi-based strategy to control a group of agents to monitor an evolving boundary [21], and [3] uses a bug algorithm to explore an obstacle boundary for finding the best position for sensing inside the obstacle. These works do not consider the pursuit-evasion problem, and also do not use the RMPC tools that we use in this paper.

Here, we use a Voronoi tessellation to divide up the reachable set of possible evader locations, with the goal of minimizing the "cost of capture" once the evader emerges from the no-fly zone. Using a virtual pursuer position, we employ a move-to-centroid algorithm to optimally divide the reachable set among the pursuers. Since the pursuers cannot move inside the no-fly zone, we drive them to the point on the boundary of the no-fly zone that is closest to the virtual pursuer inside the no-fly zone. We show this minimizes a relevant "cost-to-capture" metric, and demonstrate our algorithm in hardware experiments using three KMEL Nano+ quadrotors pursuing an m3pi ground robot.

The remainder of this paper is organized as follows. In Section II we describe the algorithm for pursuing the evader in free space, and in Section III we consider the case when the evader is inside a no-fly zone. Section IV describes the RMPC control method used to robustly execute the algorithms in the previous sections using a model of the quadrotors' dynamics with modeling uncertainties. Section V describes our hardware experiments, and we offer our conclusions in Section VI.

II. Zone-Aware Pursuit in Free Space

We first consider the case where the evader is in free space and provide a path planning algorithm to steer the pursuers towards the evader while avoiding entering any no-fly zone. In Section III, we provide the tools necessary to track the evader when it is inside a no-fly zone. Throughout this paper, we assume that the pursuers can determine the position of the evader when it is in free space, e.g. from an on-board camera or other sensing system, or from a tracking beacon on the evader. We also assume that the pursuers know there own positions, e.g. from GPS.



(a) (b)

Fig. 2: (a) Construction of the maximum-volume ellipsoid. Shaded areas represent the no-fly zones and the blue and red dots represent the location of pursuer i and the evader, respectively. (b) Schematic of the Algorithm 1 (Path Planning).

Let n_p be the number of pursuers in the group. Define $p_i(k)$ and $e(k)$ to be the xy-coordinates of the pursuer i and the evader at time k, respectively. Similarly, let h_{p_i} $(i = 1, 2, \ldots, n_p)$ and h_e denote the altitude of pursuer i and the evader, respectively. To avoid collision between the pursuer quadrotors, we set $h_{p_i} \neq h_{p_j}$ for all $i \neq j$, however this may be achieved with existing collision avoidance algorithms, as well.

Consider the case where the evader is stationary and is close enough to the pursuer such that there exists an ellipsoid, completely outside any no-fly zone, that is centered at the position of the evader and includes pursuer i (see Fig. 2a). We refer to this ellipsoid as the "safety ellipsoid." Pursuer i can safely reach the evader if there exists a controller which guarantees both that it never leaves the safety ellipsoid and also that it asymptotically converges to the center of the ellipsoid. In Section IV-A, we review a robust MPC method which can be used to guarantee both conditions.

Let $r_{\min}$ and $r_{\max}$ be the lower and upper bounds on the semi-minor axis and the semi-major axis of the safety ellipsoid as shown in Fig. 2a. Let v_κ, $\kappa = 1, \ldots, n_v$ denote a sampled set of no-fly zone boundary points at the distance of less or equal to $r_{\max}$ from the evader. For pursuers $i = \{1, ..., n_p\}$, define the maximum-volume safety ellipsoid as

$$\mathcal{Q}^i(k) \triangleq \left\{ z \in \mathbb{R}^2 \,\middle|\, (z - e(k))^T Q^i_\varepsilon (z - e(k)) \leq 1 \right\}, \quad (1)$$

where Q^i_ε is the solution of the semidefinite programming problem

$$\begin{aligned} \min \ & \text{trace}(Q^i_\varepsilon) \\ \text{s.t. } & (v_\kappa - e(k))^T Q_\varepsilon (v_\kappa - e(k)) \geq 1 + \epsilon, \\ & (p_i(k) - e(k))^T Q^i_\varepsilon (p_i(k) - e(k)) \leq 1 - \epsilon, \\ & \kappa = 1, 2, ..., n_v, \quad \frac{1}{r^2_{\max}} I \preceq Q^i_\varepsilon \preceq \frac{1}{r^2_{\min}} I. \end{aligned}$$

In the above problem, $\epsilon > 0$ is a parameter that ensures neither the sampled boundary points nor $p_i(k)$ fall on the boundary of $\mathcal{Q}^i(k)$.

The optimization problem in (2) can occasionally become infeasible, e.g., when the distance between the evader and a

pursuer is larger than $r_{\max}$ or when the direct line between a pursuer and the evader crosses a no-fly zone. In such scenarios, we search for a "dummy" evader which can move the pursuer in a direction towards the evader. Let e_i^* be the dummy evader corresponding to pursuer i and define $\mathcal{Z}_i\,(i = 1, \ldots, n_z)$, a no-fly zone, where n_z denotes the number of no-fly zones in the environment. Since the no-fly zones can be non-convex, we calculate a minimum-volume ellipsoid around them, which we use to navigate around a no-fly zone. Let $\mathcal{E}_{\mathcal{Z}_r}$ be the minimum-volume ellipsoid around $\mathcal{Z}_r$ that is centered at the centroid of $\mathcal{Z}_r$ and is $r_{\min}$ distance from the closest boundary point of $\mathcal{Z}_r$.

Let $\texttt{line}(p_i, e)$ to be the line segment from p_i to e. Define $\texttt{los}(p_i, e)$ to be the line-of-sight indicator function such that $\texttt{los}(p_i, e)$ is 0 if $\texttt{line}(p_i, e)$ crosses a no-fly zone and 1 otherwise. Consider the case where $\texttt{los}(p_i, e) = 1$. Note that the optimization problem (2) is infeasible if any point along $\texttt{line}(p_i, e)$ is not at least $r_{\min}$ distance away from the boundaries of all no-fly zones. Procedure 1 verifies if this problem exists and resolves it by assigning a new dummy evader (e_i^*) to pursuer i.

Procedure 1 Proximity Verification Procedure

Input: $p_i, e, \mathcal{Z}_j, \mathcal{E}_{\mathcal{Z}_j}\,(j = 1, \ldots, n_z)$
Output: e_i^* ▷ position of the dummy evader for pursuer i

procedure PROXIMITY(p_i, e)
  $(x_z, d_z, k) \leftarrow$ the closest point along the boundary of $\mathcal{Z}_j$ to $\texttt{line}(p_i, e)$, its distance, and the corresponding no-fly zone number, respectively
  if $d_z \geq r_{\min}$ **then**
    $e_i^* \leftarrow e$
  else
    $e_z \leftarrow$ the nearest point on $\mathcal{E}_{\mathcal{Z}_k}$ to x_z with $\texttt{los}(x_z, e_z) = 1$
    $e_i^* \leftarrow e_z$
  end if
end procedure

Even when $\texttt{los}(p_i, e) = 1$ and $\texttt{line}(p_i, e)$ is not in $r_{\min}$ proximity of any no-fly zone, the optimization problem in (2) can become infeasible due to the shape of no-fly zones and the distance between the evader and pursuer i. In such scenarios, we create and gradually move the dummy evader from the position of the evader towards pursuer i until (2) is feasible (see Step 18 of Algorithm 1 and its accompanying Fig. 2b). Finally, if $\texttt{los}(p_i, e) = 0$, we choose the dummy evader e_i^*, to be a point on the minimum-volume ellipsoid of the closest intersecting no-fly zone such that $\texttt{los}(p_i, e_i^*) = 1$ (see Step 11 of Algorithm 1).

Algorithm 1 provides a complete path planning algorithm for a given pursuer i. Each pursuer track its assigned target evader, e_i. When the evader is outside any no-fly zone, all target evaders are set to $e(k)$. However, when the evader is inside a no-fly zone, each pursuer is assigned a different target evader to improve coverage of the no-fly zone as described in Section III.

Algorithm 1 Path Planning Algorithm for Pursuer p_i

Input: $p_i(k), e_i(k), \mathcal{Z}_j, \mathcal{E}_{\mathcal{Z}_j}\,(j = 1, \ldots, n_z)$
Output: $e_i^*(k), Q_\varepsilon^i$ ▷ position of dummy evader for p_i

1: Pick $0 < \theta_p < 1$
2: $e_i^* \leftarrow e_i(k)$ ▷ $e_i(k) = e(k)$ if $e(k)$ is in free space
3: $\mathcal{L}_1 \leftarrow 1$, $\mathcal{L}_2 \leftarrow 1$ ▷ feasibility indicator for (2)
4: **while** $\mathcal{L}_1 = 1$ **do**
5: **if** e_i^* is inside a no-fly zone r **then**
6: $e_i^* \leftarrow$ nearest point to e_i^* on the boundary of $\mathcal{Z}_r$
7: **end if**
8: **if** $\texttt{los}(p_i, e_i^*) = 1$ **then**
9: $e_i^* \leftarrow$ PROXIMITY(p_i, e_i^*)
10: **end if**
11: **if** $\texttt{los}(p_i, e_i^*) = 0$ **then**
12: $r \leftarrow$ index of closest obstacle along $\texttt{line}(p_i, e_i^*)$
13: $z \leftarrow$ nearest point to e_i^* on $\mathcal{E}_{\mathcal{Z}_r}$
14: $x_{\{CW,CCW\}} \leftarrow$ nearest point to z on $\mathcal{E}_{\mathcal{Z}_r}$ in CW and CCW directions with $\texttt{los}(p_i, x_{\{CW,CCW\}}) = 1$
15: $e_i^* \leftarrow$ Pick x_{CW} or x_{CCW} that yields a shorter arc to z
16: $e_i^* \leftarrow$ PROXIMITY(p_i, e_i^*) ▷ x_P in Fig. 2b
17: **else**
18: **while** $\mathcal{L}_2 = 1$ **do**
19: $Q_\varepsilon^i \leftarrow$ Construct max-volume ellipsoid (2)
20: **if** $Q_\varepsilon^i = \emptyset$ **then**
21: $x_d \leftarrow \theta_p\, p_i + (1 - \theta_p) e_i^*$
22: **else**
23: $\mathcal{L}_1 \leftarrow 0$, $\mathcal{L}_2 \leftarrow 0$
24: **end if**
25: **end while**
26: **end if**
27: **end while**

III. Pursuing an Evader in a No-Fly Zone

The previous section described our multi-robot pursuit algorithm when the evader is outside a no-fly zone, and the pursuers know the evader's current position. In this section we address the case when the evader enters a no-fly zone. Once inside, the pursuers may not have reliable information about the evader's position, so the pursuers construct a reachable set of possible positions, based on the evader's maximum velocity and its entry point. The pursuers arrange themselves around the perimeter of the no-fly zone to be ready to capture the evader when it emerges. We adapt strategies from Voronoi-based coverage control to partition the reachable set of the evader, and position the pursuers as close as possible to a centroidal Voronoi configuration over the reachable set, without entering the no-fly zone. We note that the evader could choose the strategy to stay inside a no-fly zone so that it can never be captured. In this case, the pursuers distribute themselves around the perimeter, thereby effectively containing the evader within the no-fly zone.

Consider the no-fly zone $\mathcal{Z}_j$, where $q \in \mathcal{Z}_j$ is an arbitrary point in the zone. The set $\mathcal{Z}_j$ need not be convex. Let $v_{\max}$ be the maximum speed of the evader. Suppose an evader enters the no-fly zone at some time $t = \tau$. The entry point is denoted as $e(\tau)$. For some time $t \geq \tau$, we can find the

reachable set of the evader locations as a ball of radius $(v_{\max}(t-\tau))$ centered at $e(\tau)$, written $\mathcal{B}(e(\tau), v_{\max}(t-\tau))$. This ball may include points outside the no-fly zone, so we define $\mathcal{R}_j(t, \tau, v_{\max})$ to be the part of the reachable set that is inside the no-fly zone, written

$$\mathcal{R}_j(t, \tau, v_{\max}) = \mathcal{B}(e(\tau), v_{\max}(t-\tau)) \cap \mathcal{Z}_j.$$

We then define the indicator function $\rho(q,t)$ such that $\rho(q,t) = 1$ if $q \in \mathcal{R}_j(t, \tau, v_{\max})$ at time t, and 0 otherwise. We can write $\rho(q,t)$ as

$$\rho(q,t) = \begin{cases} 1, & \text{if } q \in \mathcal{R}_j(t, \tau, v_{\max}), \\ 0, & \text{otherwise.} \end{cases}$$

If the pursuers were free to move inside the no-fly zone, we would want to distribute the pursuers to minimize the average distance to any possible location of the evader. Let $\bar{p}_i$ be the unrestricted desired position of a pursuer. We can also define the Voronoi tessellation of the reachable set $\mathcal{R}_j$ based on these desired positions as

$$V_i = \left\{ q \in \mathcal{R}_j \;\middle|\; \|q - \bar{p}_i\|^2 \leq \|q - \bar{p}_k\|^2 \;,\; k = 1, 2, ..., n_p \right\}.$$

This allows us to formulate a "cost of capture," modeled after previous Voronoi-based coverage cost functions introduced in [7], [6] and written

$$\mathcal{V}(\bar{p}_1, ..., \bar{p}_n) = \sum_{i=1}^{n_p} \int_{V_i} \|q - \bar{p}_i\|^2 \, \rho(q,t) \, \mathrm{d}q. \tag{2}$$

Intuitively, we see that a low value of $\mathcal{V}$ indicates a good configuration of the pursuers. It is also useful to define a "mass" and "centroid" of each Voronoi cell V_i, analogous to physical masses and centroids, written

$$M_{V_i} = \int_{V_i} \rho(q,t) \, \mathrm{d}q, \quad C_{V_i} = \frac{1}{M_{V_i}} \int_{V_i} q \rho(q,t) \, \mathrm{d}q. \tag{3}$$

Although there is a complex dependency between the desired positions $\bar{p}_i$ and the geometry of the Voronoi cells, it is known from locational optimization [8] that minima of $\mathcal{V}$ correspond to configurations where all robots are at the centroids of their Voronoi cells, or $\bar{p}_i = C_{V_i}$ for all $i = 1, \ldots, n_p$. In [7] Cortés et al. introduced a move-to-centroid controller that guarantees that the robots are driven to the centroids of their Voronoi cells, and hence they reach a local minimum of (2). In our scenario, if the pursuers could enter the no-fly zone, a move-to-centroid controller would position them to locally minimize the cost of capture of the evader. However, given that our pursuers must remain outside the no-fly zone, we must adapt the move-to-centroid algorithm to this constrained case.

Remark 1. *Note that the above approach can easily be extended to more complex scenarios where predictions about the position of the evader in a no-fly zone is available. Such scenarios can be captured with either a complex reachable set $\mathcal{R}_j$ or by using indicator function to represent the probability of the position estimate at a given point $q \in \mathcal{R}_j$.*

Let $d_i = p_i - \bar{p}_i$ be the vector from the ideal pursuer position (if the pursuer could enter the no-fly zone) to the actual pursuer position (constrained to lie outside the no-fly zone). Consider minimizing the cost of capture in this case, $\mathcal{V}(p_1, ..., p_n)$, with $p_i = \bar{p}_i + d_i$, and with p_i constrained to lie outside $\mathcal{Z}_j$. For $i = \{1, ..., n_p\}$, $j = \{1, ..., n_z\}$, we can re-write the cost as a function of the vectors d_i as

$$\mathcal{H}(d_1, \ldots, d_n) = \sum_{i=1}^{n_p} \int_{V_i} \|q - \bar{p}_i - d_i\|^2 \rho(q,t) \, \mathrm{d}q \tag{4}$$

where $\bar{p}_i = C_{V_i}$, are the unconstrained locally optimal positions. Then our problem is to solve the following optimization

$$\min_{(d_1, \ldots, d_n)} \mathcal{H}(d_1, \ldots, d_n)$$
$$\text{s.t.} \quad d_i + \bar{p}_i \notin \mathcal{Z}_j.$$

We find constrained positions to optimize the cost of capture function (4) with the following proposition.

Proposition 1 (Projected Centroids)**.** *Given $\bar{p}_i = C_{V_i}$ for all $i = 1, \ldots, n_p$, the locations p_i that minimize the cost to capture (4) are given by $p_i = \bar{p}_i + d_i^*$, where*

$$d_i^* = \arg \min_{d_i + \bar{p}_i \in \partial \mathcal{Z}_j} \|d_i\|, \tag{5}$$

and $\partial \mathcal{Z}_j$ denotes the boundary of the no-fly zone $\mathcal{Z}_j$.

Proof. To see that (5) minimizes (4) given our constraints, we first expand the cost function as

$$\mathcal{H} = \sum_{i=1}^{n_p} \int_{V_i} \left(\|q + \bar{p}_i\|^2 - 2 d_i^T (q - \bar{p}_i) + \|d_i\|^2 \right) \rho(q,t) \, \mathrm{d}q.$$

Breaking the above equation into three parts, we find $\mathcal{H} = \mathcal{H}_1 + \mathcal{H}_2 + \mathcal{H}_3$, where

$$\mathcal{H}_1 = \sum_{i=1}^{n_p} \int_{V_i} \left(\|q - \bar{p}_i\|^2 \right) \rho(q,t) \, \mathrm{d}q,$$

$$\mathcal{H}_2 = -2 \sum_{i=1}^{n_p} d_i^T \int_{V_i} (q - \bar{p}_i) \rho(q,t) \, \mathrm{d}q,$$

$$\mathcal{H}_3 = \sum_{i=1}^{n_p} \|d_i\|^2 \int_{V_i} \rho(q,t) \, \mathrm{d}q = \sum_{i=1}^{n_p} \|d_i\|^2 M_{V_i}.$$

We see that $\mathcal{H}_1$ is the same as the unconstrained cost function (2), and is independent of d_i, so letting $\bar{p}_i = C_{V_i}$ yields a local minimum of $\mathcal{H}_1$ [8]. When $\bar{p}_i = C_{V_i}$, it follows from (3) that $\mathcal{H}_2 = 0$. It remains then to minimize $\mathcal{H}_3$, which is accomplished by minimizing $\|d_i\|$, since M_{V_i} does not depend on d_i. Given that the pursuer cannot enter the no-fly zone, $\|d_i\|$ is minimized when p_i is the closest point to the centroid C_{V_i} on the boundary of the no-fly zone $\mathcal{Z}_j$. □

Note that Step 5 in Algorithm 1 in effect ensures that $e_i^* = d_i$, where e_i^* denotes the position of the dummy evader for pursuer i. Therefore, each centroid C_{V_i} can be treated as an evader target position that can be assigned to a pursuer.

Figure 3 illustrates the evolution of our no-fly zone planning algorithm. Once the evader enters a no-fly zone, the pursuers generate an estimate of the reachable set $\mathcal{R}_j$, updated at each time step. Between time step, the pursuers

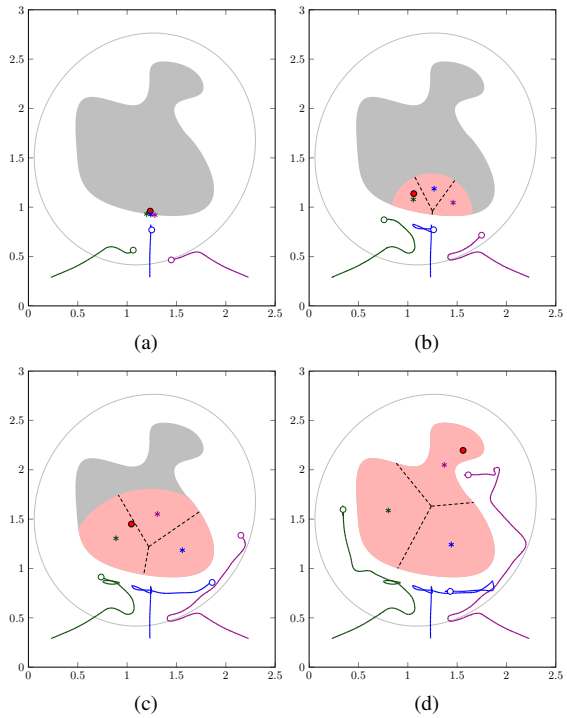


Fig. 3: Simulation of three pursuers executing the no-fly-zone planning algorithm. At each step, the pursuers move to the nearest feasible point to the centroids of the Voronoi cells.

use a move-to-centroid algorithm that projects their desired position to the center of their Voronoi cell (see Algorithm 2). A simple target assignment, using the well-known Hungarian algorithm, matches each centroid point $\bar{p}_i = C_{V_i}$ with a pursuer. Algorithm 1 is then applied to each pursuer and its associated dummy evader.

Algorithm 2 No-Fly Zone Planning Algorithm

Input: $e(\tau)$, t, $\mathcal{Z}_j$, v_{max}
Output: $\bar{p}_i$

```
Calculate B(e(τ), v_max(t−τ)) and R_j(e(τ), v_max(t−τ))
Compute Voronoi tessellation about p̄_i
while p̄_i ≠ C_{V_i} ∀ i do
    Assign p̄_i = C_{V_i}
    Recompute Voronoi tessellation
end while
Assign p̄_i as target points for p_i
```

IV. Quadrotor Modeling and Control

Up to this point, we have assumed that we can control the pursuer positions to a desired waypoint, while staying inside an elipse. Here we describe the low-level controller used to accomplish this. We first control the rotational dynamics of the quadrotor to a desired attitude, which then allows us to control the translational dynamics to a desired location [2]. The translational sub-system controls the position and velocity of the quadrotor by generating the total rotor output and the desired roll and pitch angles. The rotational sub-system regulates the attitude of the vehicle to the desired values computed by the translational sub-system.

In [1], we designed a robust MPC controller for the translational sub-system and an LQR controller for the rotational sub-system. Here, we assume the rotational sub-system is regulated through an on-board controller which rotates the frame to the desired angles and consider the control design only for the translational dynamics of the aircraft. We borrow elements from the translational controller in [1] and expand it to a group of quadrotors. While the relevant theorems and propositions are included for completeness, we refer the reader to [1] for their proof.

Let U_i be the total output of the rotors for pursuer i and define U^* to be the total output required for hovering. We denote the control inputs to the translational sub-system corresponding to the desired roll and pitch angles of the pursuer i by ϕ_{d_i}, θ_{d_i}, respectively. We treat the desired yaw angle as a parameter set by the user, and without loss of generality we set it to zero. Let $\xi_i = (p_i, h_{p_i})$, where p_i denotes the xy-coordinates of pursuer i and h_{p_i} denotes its altitude. Define $\hat{x}_i = (\xi_i, \dot{\xi}_i)$ and $\hat{u}_i = (U_i, \phi_{d_i}, \theta_{d_i})$ to be the translational state space and control input vectors for pursuer i, respectively.

Suppose e_i^* is the dummy evader for pursuer i as defined in Sections II and set $\xi_i^* = (e_i^*, h_e)$. Let $x_i^* = (\xi_i^*, 0_3)$ be a hovering position with $u^* = (U^*, 0, 0)$. A discrete-time linear model of the translational sub-system can be derived by linearizing the nonlinear system dynamics around x^* and u^*, which yields [1],

$$x_i(k+1) = A_i x_i(k) + B_i u_i(k), \tag{6}$$

where $x_i = \hat{x}_i - x_i^*$ and $u_i = \hat{u}_i - u^*$. The state and input matrices in (6) are given by,

$$A_i = \begin{bmatrix} I_3 & T_s I_3 \\ 0_3 & I_3 \end{bmatrix}, \quad B_i = T_s \begin{bmatrix} 0_3 \\ \hline 0 & 0 & -g \\ 0 & g & 0 \\ \frac{-g}{U^*} & 0 & 0 \end{bmatrix},$$

where T_s denotes the sampling time and g is the gravitational acceleration.

The linearized system in (6) is subject to modeling errors induced by system linearization. These effects become specially dominant when the quadrotor performs more aggressive maneuver with large roll and pitch angles. Furthermore, the uncertainty caused by disturbance and measurement errors in p_i and e can also be captured as uncertainties in the system. We can therefore treat the system as a linear time-variant system with uncertain A_i and B_i matrices. Since the algorithm requires each pursuer to remain inside its safety ellipsoid, we wish to construct a controller which not only achieves stability but also guarantees that a pursuer trajectory never leaves its corresponding safety ellipsoid even in the presence of uncertainties. To this end, we use the following robust MPC technique proposed in our earlier work in [1] and included here for completeness.

A. Robust Model Predictive Control

To improve the readability, throughout this section, we use superscript κ to denote a variable corresponding to pursuer κ. Consider the discrete-time linear time-variant system in

(6) for pursuer κ with the output vector y^κ defines as,

$$\begin{aligned} x^\kappa(k+1) &= A^\kappa(k)x^\kappa(k) + B^\kappa(k)u^\kappa(k), \\ y^\kappa(k) &= C^\kappa x^\kappa(k), \end{aligned} \tag{7}$$

where $x(k) \in \mathbb{R}^{n_x}$, $u \in \mathbb{R}^{n_u}$ and $y \in \mathbb{R}^{n_y}$. Let $D^\kappa \triangleq [A^\kappa \,|\, B^\kappa]$ and define the uncertainty set as,

$$\mathcal{U} \triangleq \{D \in \mathbb{R}^{n_x \times (n_x+n_u)} \big|\ |D^\kappa_{ij} - \bar{D}_{ij}| \le \Delta^\kappa_{ij},\ \forall i,j\}, \tag{8}$$

where $\bar{D} = [\bar{A}|\bar{B}]$ denotes the matrix corresponding to the nominal values of A^κ and B^κ, and $\Delta^\kappa = [\Delta^\kappa_{ij}]$ is a matrix with non-negative elements.

Let δ be the vector containing the non-zero elements of Δ^κ. For the l-th element of δ, let i_{δ_l} and j_{δ_l} denote the row and column index of δ_l in Δ, respectively (i.e. $\Delta_{i_{\delta_l} j_{\delta_l}} = \delta_l$). Define $U^\kappa(i,j,\Delta_{ij})$ to be a matrix whose $ij-$th element is set to Δ^κ_{ij} and the rest are zero. We can now write D as,

$$D^\kappa = \bar{D} + \sum_{l=1}^{n_\delta} \zeta_l U^\kappa(i_{\delta_l}, j_{\delta_l}, \Delta^\kappa_{i_{\delta_l} j_{\delta_l}}), \tag{9}$$

where $|\zeta_l| \le 1$. Note that the above definition effectively covers the set of uncertain A^κ and B^κ matrices with polytopic uncertainties.

Let $x^\kappa(k+i|k)$ be an estimate of x^κ at sampling time $k+i$ based on the measurements obtained at time k. For brevity, we denote $x^\kappa(k+i|k)$ for all $i \ge 0$ by $x^\kappa(k+i)$ and extend the same notation for other variables. Consider an MPC problem with an infinite prediction horizon, where at each sampling time k, a control law $u^\kappa(k+i)\,(i \ge 0)$ is designed to solve the following min-max optimization problem

$$\min_{u^\kappa(k+i),\, i\ge 0} \max_{(A^\kappa, B^\kappa)\in\mathcal{U}} J^\kappa_\infty(k), \quad \text{where} \tag{10}$$

$$\begin{aligned} J^\kappa_\infty(k) \triangleq \sum_{i=0}^{\infty} & x^\kappa(k+i)^T Q_\infty x^\kappa(k+i) \\ & + u^\kappa(k+i)^T R_\infty u^\kappa(k+i), \end{aligned} \tag{11}$$

$Q_\infty \succ 0$, and $R_\infty \succeq 0$.

Suppose there exists a feedback control law $u^\kappa(k+i) = F^\kappa x^\kappa(k+i)$ and a quadratic function $V^\kappa(x) = (x^\kappa)^T P^\kappa x^\kappa$ with $P \succ 0$ such that for all $i \ge 0$,

$$\begin{aligned} & V^\kappa(x^\kappa(k+i)) - V^\kappa(x^\kappa(k+i+1)) \ge \\ & x^\kappa(k+i)^T Q_\infty x^\kappa(k+i) + u^\kappa(k+i)^T R_\infty u^\kappa(k+i). \end{aligned} \tag{12}$$

Then, $V^\kappa(x^\kappa(k))$ is a Lyapunov function for the optimization problem (11) [5]. Note that the control law should satisfy (12) for all $(A^\kappa, B^\kappa) \in \mathcal{U}^\kappa$. This, in general, will require solving the optimization problem over all vertices of $\mathcal{U}^\kappa$ which grow exponentially with the dimension of $\mathcal{U}^\kappa$. In [1], we show an alternative approach where the size of the problem grows linearly with the dimension of $\mathcal{U}^\kappa$.

Theorem 1 ([1])**.** *Consider the uncertain discrete system in* (7)*, where uncertainties are defined by* (8)*. Let $u^\kappa(k+i) = F^\kappa x^\kappa(k+i)$ be the control action for time $k+i$ for all $i \ge 0$. Consider the following LMI problem:*

$$\begin{aligned} \min_{\gamma^\kappa, Q^\kappa, G^\kappa, Z^\kappa_{\delta_l \,|\, l=1,\ldots,n_\delta}} \quad & \gamma^\kappa \\ \text{s.t.} \quad & \begin{bmatrix} 1 & * \\ x^\kappa(k) & Q^\kappa \end{bmatrix} \succeq 0,\ L^\kappa_0 \succeq 0,\ \sum_{l=1}^{n_\delta} Z^\kappa_{\delta_l} \preceq L^\kappa_0, \\ & Z^\kappa_{\delta_l} \succeq L^\kappa_{\delta_l},\ Z^\kappa_{\delta_l} \succeq -L^\kappa_{\delta_l}, \quad \forall l = 1,2,\ldots,n_\delta \end{aligned} \tag{13}$$

where $Z^\kappa_{\delta_l} = (Z^\kappa_{\delta_l})^T$,

$$L^\kappa_0 = \begin{bmatrix} Q & * & * & * \\ \bar{A}Q^\kappa + \bar{B}G^\kappa & Q^\kappa & * & * \\ Q^{\frac{1}{2}}_\infty Q^\kappa & 0 & \gamma^\kappa I & * \\ R^{\frac{1}{2}}_\infty G^\kappa & 0 & 0 & \gamma^\kappa I \end{bmatrix},$$

$$L^\kappa_{\delta_l} = \begin{bmatrix} 0 & * & * & * \\ A^\kappa_{\delta_l} Q^\kappa + B^\kappa_{\delta_l} G^\kappa & 0 & * & * \\ 0 & 0 & 0 & * \\ 0 & 0 & 0 & 0 \end{bmatrix},$$

$A^\kappa_{\delta_l}$ and $B^\kappa_{\delta_l}$ are matrices with the same dimensions as A^κ and B^κ, respectively, whose values are extracted from $U^\kappa(i_{\delta_l}, j_{\delta_l}, \Delta^\kappa_{i_{\delta_l} j_{\delta_l}}) = [A^\kappa_{\delta_l} \,|\, B^\kappa_{\delta_l}]$ in (9)*. If the above optimization problem has a solution, then J^κ_∞ in the worst-case scenario is bounded from above by γ^κ and the minimizing feedback control gain is given by,*

$$F^\kappa = G^\kappa (Q^\kappa)^{-1}. \tag{14}$$

While the above theorem guarantees stability of the quadrotor system for the given uncertainty set, it does not ensure that the trajectories will always remain inside the safety ellipsoid. Let Q^κ_ε be the safety ellipsoid for pursuer κ calculated from (2). The following proposition guarantees that the pursuer trajectory does not leave the safety ellipsoid.

Proposition 2 (Output Constraints [1])**.** *Consider the uncertain discrete system in* (7)*, the feedback control law $u^\kappa(k+i) = F^\kappa x^\kappa(k+i)$, where F^κ is given by* (14)*, and the safety ellipsoid $\mathcal{Q}^\kappa$ in* (1)*. Let $C = [I_2 \,|\, 0_{2\times 4}]$. The pursuer κ is guaranteed to remain in the safety ellipsoid $\mathcal{Q}^\kappa$ if the following LMI feasibility problem has a solution,*

$$\begin{aligned} \text{find} \quad & \tilde{Z}^\kappa_{\delta_l \,|\, l=1,\ldots,n_\delta} \\ \text{s.t.} \quad & M^\kappa_0 \succeq 0,\ \sum_{l=1}^{n_\delta} \tilde{Z}^\kappa_{\delta_l} \preceq M^\kappa_0, \tilde{Z}^\kappa_{\delta_l} \succeq M^\kappa_{\delta_l},\ \tilde{Z}^\kappa_{\delta_l} \succeq -M^\kappa_{\delta_l}, \end{aligned}$$

where $\tilde{Z}^\kappa_{\delta_l} = (\tilde{Z}^\kappa_{\delta_l})^T$, $l = 1,\ldots,n_\delta$, and

$$M^\kappa_0 = \begin{bmatrix} Q^\kappa & * \\ C(\bar{A}Q^\kappa + \bar{B}G^\kappa) & (Q^\kappa_\varepsilon)^{-1} \end{bmatrix},$$

$$M^\kappa_{\delta_l} = \begin{bmatrix} 0 & * \\ C(A^\kappa_{\delta_l} Q^\kappa + B^\kappa_{\delta_l} G^\kappa) & 0 \end{bmatrix}.$$

To ensure stability and safety conditions are satisfied simultaneously, we add the constraints in Proposition 2 to Theorem 1 and solve the resulting optimization problem.

V. Experiments

In this section, we present experimental results demonstrating our proposed tracking algorithm. For the pursuers,

we used three KMEL Nano+ quadrotors equipped with Kbee radios for communication. Localization was performed with NaturalPoint's OptiTrack system. MATLAB was used to perform all calculations, and the updated waypoints were transmitted to the quadrotors. Two scenarios are presented. The first experiment uses a simulated evader following a pre-planned trajectory. The pursuers do not know the pre-planned trajectory, but react to the evader in real time. The second experiment uses a Pololu m3pi robot manually driven with a joystick, and pursuers react in real time. A video with both experimental runs can be found in the video attachment, and can be viewed on our website[1]. In both experiments, we see the pursuers are able to successfully track the evader as it moves in and out of the no-fly zones.

In the first experiment, three quadrotors tracked a simulated evader as it moved throughout the environment. The trajectories can be seen in Figure 4(a), with the evader shown in red and final positions denoted by the circles. Over time, we see that the trajectories of the pursuers remain outside the no-fly zones, demonstrating a successful implementation of the path-planning algorithm. Figure 4(b) shows the minimum distance from any pursuer to the evader over time. The shaded areas indicate when the evader was within a no-fly zone, and the red dashed line corresponds to the maximum distance the evader could achieve given its entry point and maximum velocity. By employing the centroidal Voronoi algorithm, we find that the minimum distance remains relatively small, despite the pursuers not knowing the true evader position. By the end of the experiment, the pursuers are within 10 cm of the evader.

Figure 5 shows stills from the experimental video demonstrating the Voronoi-based coverage strategy while the evader is in the no-fly zone. Over time, we see the pursuers spread out as $\mathcal{R}_j$ grows. In the final frame, the evader emerges from the top of the no-fly zone, with a pursuer nearby.

For our second experiment, we controlled an m3pi robot with a joystick to create a "live" evader for our three quadrotors to track. The trajectories over time are shown in Figure 6(a). Again, we see the pursuers never enter the no-fly zone. Figure 6(b) plots the minimum distance from any pursuer to the evader, as well as the maximum possible distance in red. Our Voronoi-based control strategy keeps the distance to the evader relatively small, even though its position is unknown. Stills from the experimental video are shown in Figure 7. As with the simulated evader, we see the pursuers distributing themselves around the boundary of $\mathcal{Z}_j$ while the evader remains inside the no-fly zone.

VI. Conclusions and Future Work

This paper presents a series of algorithms to coordinate a group of pursuers tracking an evader while avoiding no-fly zones. While the evader remains outside the no-fly zone, we assume the pursuers know the evaders position. Once the evader enters a no-fly zone, an estimate of the reachable set of all evader positions is generated based on the entry point and the maximum velocity of the evader. Each pursuer is then assigned to the centroid of a Voronoi cell, which is used to distribute the pursuers about the zone's boundary. Through experiments, we show that as a result of the coordinated pursuit, the quadrotors remain in close proximity of the evader even when it enters a no-fly zone.

[1]http://sites.bu.edu/msl/research/cooperative-multiquad-pursuit

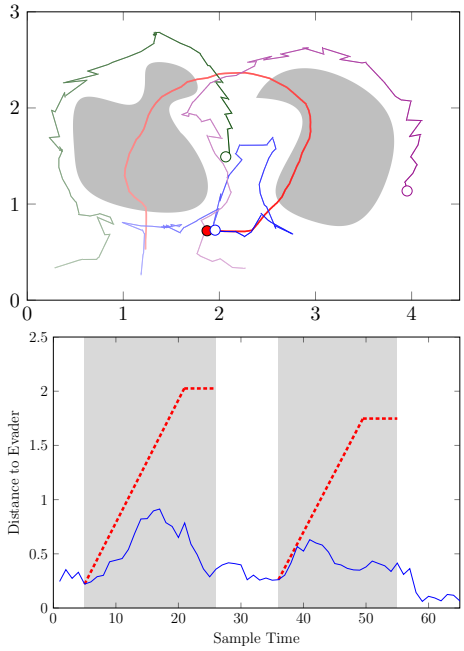


Fig. 4: (Top) Trajectories of the pursuers (green, blue, purple) and evader (red) over time. (Bottom) Minimum distance from any pursuer to the evader over time. The red dashed line shows the maximum distance to the evader inside the no-fly zone. Our strategy keeps the distance well below the max.

Here, when the evader is in free space, all pursuers track to the same target. Future work may consider a better way to track the target using the group for more efficient capture. Another direction may consider the case where pursuers only know the evader's position if there exists a direct line of sight from at least one pursuer.

References

[1] Armin Ataei and Ioannis Ch. Paschalidis. Robust model predictive control of quadrotors for zone-aware tracking. Technical Report 2015-IR-0002, Center for Information and Systems Engineering (CISE), Boston University, 2015.

[2] Samir Bouabdallah and Roland Siegwart. Full control of a quadrotor. In *Intelligent robots and systems, 2007. IROS 2007. IEEE/RSJ international conference on*, pages 153–158, 2007.

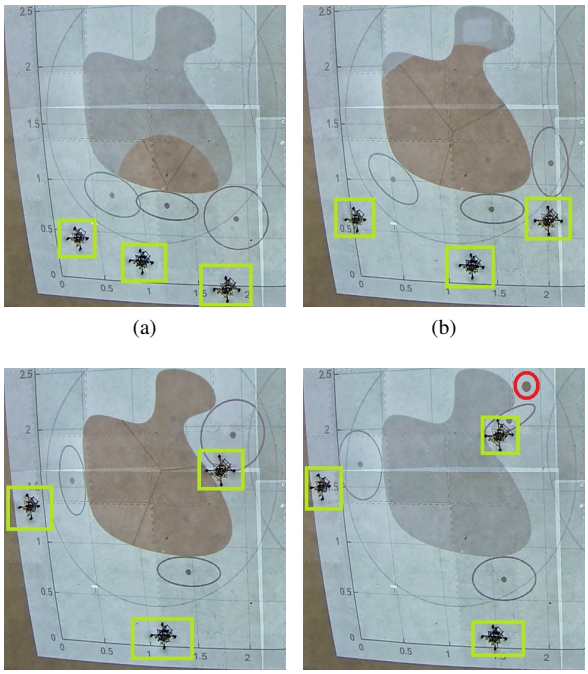
[3] A. Breitenmoser, M. Schwager, J. C. Metzger, R. Siegwart, and D. Rus. Voronoi coverage of non-convex environments with a group of networked robots. In *Proc. of the International Conference on Robotics and Automation (ICRA 10)*, pages 4982–4989, Anchorage, Alaska, USA, May 2010.

[4] Oliver Brock and Oussama Khatib. High-speed navigation using the global dynamic window approach. In *Robotics and Automation, 1999. Proceedings. 1999 IEEE International Conference on*, volume 1, pages 341–346, 1999.

[5] E.F. Camacho and C.B. Alba. *Model Predictive Control*. Advanced Textbooks in Control and Signal Processing. Springer, 2013.

[6] Jorge Cortés. Coverage optimization and spatial load balancing by robotic sensor networks. *Automatic Control, IEEE Transactions on*, 55(3):749–754, 2010.

[7] Jorge Cortes, Sonia Martinez, Timur Karatas, and Francesco Bullo. Coverage control for mobile sensing networks. *Robotics and Automation, IEEE Transactions on*, 20(2):243–255, 2004.



(a) (b) (c) (d)

Fig. 5: Stills from the experimental video illustrating the control strategy while an evader is in a no-fly zone. The pursuers (yellow squares) arrange themselves around the boundary, waiting for the evader to emerge in (d).

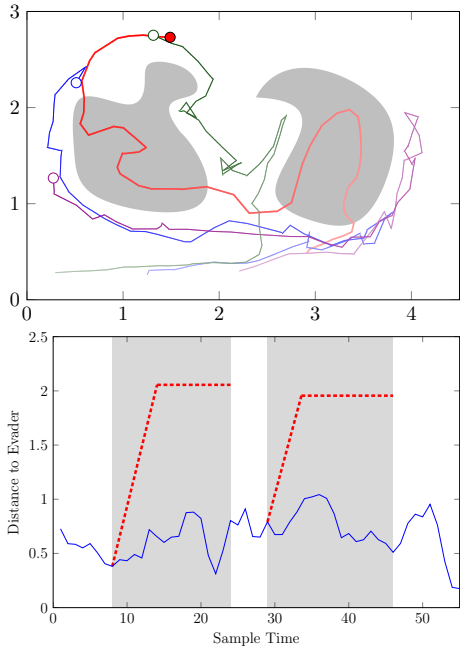
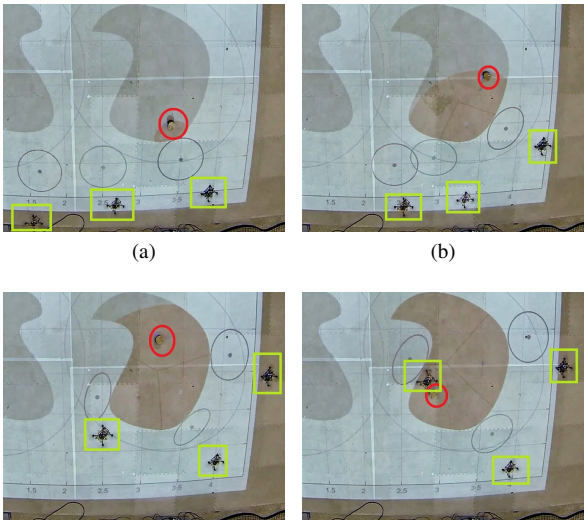


Fig. 6: (Top) Trajectories of the pursuers (green, blue purple) and evader (red) over time. (Bottom) Minimum distance from any pursuer to the evader over time. The red dashed line shows the maximum distance to the evader inside the no-fly zone. Our strategy keeps the distance well below the maximum.



(a) (b) (c) (d)

Fig. 7: Stills from the experimental video illustrating the no-fly-zone planning. The pursuers are denoted by the yellow squares, and the evader is circled in red.

[8] Z. Drezner. *Facility location: a survey of applications and methods*. Springer series in operations research. Springer, 1995.

[9] Qiang Du, Vance Faber, and Max Gunzburger. Centroidal voronoi tessellations: Applications and algorithms. *SIAM review*, 41(4):637–676, 1999.

[10] C. Goerzen, Z. Kong, and B. Mettler. A survey of motion planning algorithms from the perspective of autonomous UAV guidance. *Journal of Intelligent and Robotic Systems*, 57(1-4):65–100, 2010.

[11] Haomiao Huang, Wei Zhang, J. Ding, D.M. Stipanovic, and C.J. Tomlin. Guaranteed decentralized pursuit-evasion in the plane with multiple pursuers. In *Decision and Control and European Control Conference (CDC-ECC), 2011 50th IEEE Conference on*, pages 4835–4840, Dec 2011.

[12] Mayuresh V. Kothare, Venkataramanan Balakrishnan, and Manfred Morari. Robust constrained model predictive control using linear matrix inequalities. *Automatica*, 32(10):1361–1379, 1996.

[13] Jean-Claude Latombe. *Robot Motion Planning*. The Springer International Series in Engineering and Computer Science. Springer, 1991.

[14] S. G. Lee and M. Egerstedt. Controlled coverage using time-varying density functions. In *Proc. of the IFAC Workshop on Estimation and Control of Networked Systems*, 2013.

[15] Shih-Yuan Liu, Zhengyuan Zhou, Claire Tomlin, and Karl Hedrick. Evasion as a team against a faster pursuer. In *American Control Conference (ACC), 2013*, pages 5368–5373. IEEE, 2013.

[16] Javier Minguez and Luis Montano. Nearness diagram (nd) navigation: collision avoidance in troublesome scenarios. *Robotics and Automation, IEEE Transactions on*, 20(1):45–59, 2004.

[17] Petter Ogren and Naomi Ehrich Leonard. A convergent dynamic window approach to obstacle avoidance. *Robotics, IEEE Transactions on*, 21(2):188–195, 2005.

[18] S. Pan, Haomiao Huang, J. Ding, Wei Zhang, D.M. Stipanovic, and C.J. Tomlin. Pursuit, evasion and defense in the plane. In *American Control Conference (ACC), 2012*, pages 4167–4173, 2012.

[19] LucianoC.A. Pimenta, Mac Schwager, Quentin Lindsey, Vijay Kumar, Daniela Rus, RenatoC. Mesquita, and GuilhermeA.S. Pereira. Simultaneous coverage and tracking (scat) of moving targets with robot networks. In GregoryS. Chirikjian, Howie Choset, Marco Morales, and Todd Murphey, editors, *Algorithmic Foundation of Robotics VIII*, volume 57 of *Springer Tracts in Advanced Robotics*, pages 85–99. Springer Berlin Heidelberg, 2010.

[20] Christian Schlegel. Fast local obstacle avoidance under kinematic and dynamic constraints for a mobile robot. In *Intelligent Robots and Systems, 1998. Proceedings., 1998 IEEE/RSJ International Conference on*, volume 1, pages 594–599. IEEE, 1998.

[21] S. Susca, F. Bullo, and S. Martinez. Monitoring environmental boundaries with a robotic sensor network. *Control Systems Technology, IEEE Transactions on*, 16(2):288–296, March 2008.