2023 IEEE Conference on Control Technology and Applications (CCTA)
August 16-18, 2023. Bridgetown, Barbados

2023 IEEE Conference on Control Technology and Applications (CCTA) | 979-8-3503-3544-6/23/$31.00 ©2023 IEEE | DOI: 10.1109/CCTA54093.2023.10252592

An Optimization-Based Guidance System for Tactical Multi-Rotor UAVs Mapping Contested Environments

Julius A. Marshall[1], Paul Binder[2], and Andrea L'Afflitto[1]

***Abstract*— This paper presents an original guidance system for autonomous multi-rotor unmanned aerial vehicles (UAVs) tasked with creating maps of environments while operating in a tactical manner. An original algorithm is developed to produce a sequence of goal points whose locations enable exploration in a systematic or greedy manner, a path planner allows to interpolate goal points while exploiting obstacles for shelter, and a trajectory planner designed within the model predictive control framework enables collision-avoidance and accounts for the output feedback linearized dynamics, thrust saturation, and attitude constraints. This system assumes no prior information on the environment or threats, does not rely on external sources of information, and is equipped with forward-facing cameras to enable low altitude and indoor operations. Numerical simulations validate the proposed guidance system.**

I. Introduction

The problem of covering unknown environments using UAVs has been addressed for multiple purposes, including agriculture, mining, and civil engineering [1], [2]. However, the use of UAVs to cover potentially hostile environments has drawn less attention. These vehicles could provide useful support to first responders in perilous contexts [3] due to their payload capability and maneuverability. Of the publications on this topic, we recall [4], where a modified logistic map and a modulo tactic are used to generate unpredictable motion profiles, [5], where paths for teams of UAVs providing continuous coverage and sustained situational awareness are designed using an approach based on nonlinear programming, and [6], where genetic algorithms produce reference paths for heterogeneous fleets of UAVs flying at low altitude in dynamic and hostile environments. The systems in [4]–[6] propose path planners only, however, neglecting the UAV's dynamics reduces the vehicle's ability to operate in cluttered environments, and fast trajectory planners and collision avoidance algorithms should be included to succeed in support roles. However, many UAVs to be employed in potentially hostile environments leverage external information on both the location and the nature of the threats [5], [6], and these assumptions severely limit or prevent the use of these vehicles in completely unknown environments [1].

This paper presents an original guidance system for autonomous multi-rotor UAVs tasked with mapping potentially hostile environments, while flying at low altitudes or indoors, comprises both a fast path planner and a fast trajectory planner, does not rely on *a priori* information about the environment, and relies on forward-facing cameras for navigation. The proposed guidance system implements an explore-then-exploit approach [7], since its path planner outlines a suitable strategy to cover a user-defined set, and its trajectory planner enables the mapping process while exploiting the vehicle's dynamics [2].

In this paper, an original algorithm is presented for deducing a sequence of goal points to visit in order to cover an environment. This algorithm relies on Octrees [8], and allows the user to choose from a range of exploration behaviors. The user may choose between a *greedy* behavior, where goals lie in areas such that large amounts of new information can be gained, or a *systematic* behavior, where goals lie in areas so that exploration is enabled in a breadth-first manner. In a *systematic* approach, however, full coverage can be slow, and many back-and-forth motions are necessary, which may undermine the system's stealth objective [2]. An optimization-based path planning algorithm produces sequences of waypoints connecting consecutive goal points. This method of selecting goals and interpolating them is designed to avoid traversing all or almost all voxels in the voxel map, which is typically required by the boustrophedon, rectangular spiral, or Hilbert paths popular in the coverage literature [2], [9]. This approach may save time and effort especially in emergency situations.

The trajectory planner is designed within the model predictive control (MPC) framework and produces reference trajectories that interpolate the planned path. This planner's cost function captures the effort to control the UAV, the importance of intercepting any waypoint in the planned path, and the need to coast obstacles and seek shelter for tactical purposes. The UAV's dynamics are captured by the UAV's discrete-time output-feedback linearized equations of motion. The trajectory planner enables collision avoidance and accounts for the UAV's nonlinear dynamics. In contrast with the trajectory planner presented in [10], by employing barrier functions, constraints on the attitude and each propeller's maximum thrust are imposed. A quadratic programming framework is applied to solve the MPC problem in real time [10].

To enable reliable operations in dangerous environments, the path planner and trajectory planner allow the UAV to exploit obstacles for shelter and regulate its acceleration in response to its proximity to obstacles so that the chance of being detected by potential threats is reduced; these abilities enable *tactical* or *stealthy* motion planning. Conversely, when the planners disregard proximity to known obstacles

[1]Julius A. Marshall and Andrea L'Afflitto are with the Grado Department of Industrial and Systems Engineering, Virginia Tech, Blacksburg, VA 24061, USA ({mjulius,a.lafflitto}@vt.edu).

[2]Paul Binder is with the Kevin T. Crofton Department of Aerospace and Ocean Engineering, Virginia Tech, Blacksburg, VA 24061, USA (paulbinder@vt.edu).

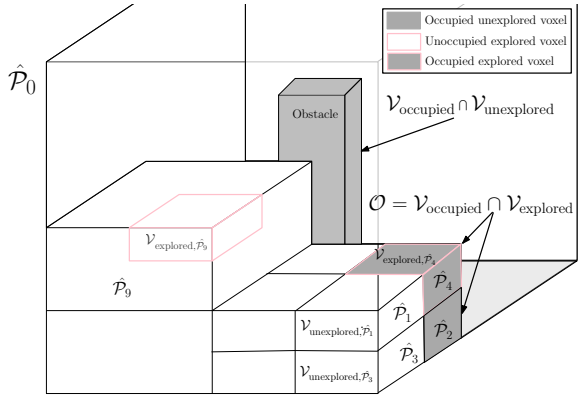

Fig. 1. A voxel map and its partitions $\mathcal{P}_i$, $i = 0, \ldots, 9$. Occupied unexplored voxels are marked in gray, unoccupied and explored voxels are marked with a pink boundary, and occupied explored voxels $\mathcal{O}$ are marked in gray with a pink boundary.

(beyond collision-avoidance), it may enable *reckless* motion planning. The need to attain a tactical, reckless, or mixed motion plan can be set by tuning parameters.

II. A Path Planner for Tactical Coverage

A. Notation

A *voxel* is a cube of user-defined dimensions. Let the connected set $\mathcal{V} \subset \mathbb{R}^3$ denote the union of $n_{\mathcal{V}} \in \mathbb{N}$ congruent voxels that cover the environment to be mapped. Each voxel in $\mathcal{V}$ is denoted by its center located at $\hat{r}_p \in \mathcal{V}$, $p \in \{1, \ldots, n_{\mathcal{V}}\}$. Let $\mathcal{V}_{\text{unexplored}}$ and $\mathcal{V}_{\text{explored}}$ capture the *unexplored subset* and the *explored subset* of $\mathcal{V}$, respectively. Let $\mathcal{V}_{\text{occupied}}$ capture the *occupied subset* of $\mathcal{V}$, respectively. The *obstacles' set* $\mathcal{O} \triangleq \mathcal{V}_{\text{occupied}} \cap \mathcal{V}_{\text{explored}}$ comprises voxels that are occupied and explored; the cardinality of $\mathcal{O}$ is denoted by $n_{\mathcal{O}}$. The proposed system is only aware of obstacles in $\mathcal{O}$, and this set is produced only when the UAV's camera observes a voxel.

The set $\mathcal{V}$ is partitioned in $n_{\mathcal{P}}$ rectangular parallelepipeds $\mathcal{P}_i$, $i \in \{1, \ldots, n_{\mathcal{P}}\}$, such that $\mathcal{P}_i$ is the union of voxels, $\bigcup_{i=1}^{n_{\mathcal{P}}} \mathcal{P}_i = \mathcal{V}$, and $\mathcal{P}_i \cap \mathcal{P}_j = \{\emptyset\}$ for all $i, j \in \{1, \ldots, n_{\mathcal{P}}\}$ such that $i \neq j$. The *barycenter of* $\mathcal{P}_i$, $i \in \{1, \ldots, n_{\mathcal{P}}\}$ is denoted by $\hat{r}_{\mathcal{P}_i} \in \mathcal{P}_i$. Lastly, the smallest parallelepiped containing $\mathcal{V}$ is denoted by $\mathcal{P}_0$.

The number of explored voxels in $\mathcal{P}_i$, $i = 0 \ldots, n_{\mathcal{P}}$, is denoted by $n_{\text{explored},\mathcal{P}_i}$, the number of occupied voxels in $\mathcal{P}_i$ is denoted by $n_{\text{occupied},\mathcal{P}_i}$, the number of unexplored voxels in $\mathcal{P}_i$ is denoted by $n_{\text{unexplored},\mathcal{P}_i}$, and the length of the smallest edge of $\mathcal{P}_i$ is denoted by $\ell_{\mathcal{P}_i,\min}$. See Figure 1 for an illustration of a partitioned voxel map.

The number of goal points is denoted by $n_{\text{g}} \in \mathbb{N}$, and is incremented everytime a new goal is generated. The number of waypoints generated by the path planner is denoted by $n_{\text{w}} \in \mathbb{N}$. In general, n_{w} is different for every pair of initial positions and goal points, and depends on the resolution of the voxel map. In the remainder of this paper, the integer q counts the goal points and spans the interval $\{0, \ldots, n_{\text{g}}\}$, the integer k counts the waypoints outlined by the path planner and spans the interval $\{0, \ldots, n_{\text{w}} - 1\}$ unless explicitly stated otherwise.

Algorithm 1 Find goal points $\hat{r}_{\widetilde{\mathcal{P}},q}$

1: **Set** $q = 0$
2: **while** $\exists \tilde{\mathcal{P}}_i$ s.t. $\frac{n_{\text{explored},\tilde{\mathcal{P}}_i}}{n_{\tilde{\mathcal{P}}_i}} < \mu_1$ **do**
3: **Execute** Algorithm 2 to partition $\mathcal{P}_0$ in $\{\mathcal{P}_1, \ldots, \mathcal{P}_{n_{\mathcal{P}}}\}$
4: **Determine** $\left\{\widetilde{\mathcal{P}}_1, \ldots, \widetilde{\mathcal{P}}_{\tilde{n}_{\mathcal{P}}}\right\}$
5: **Compute** $\hat{r}_{\mathcal{P}_{\text{prox}}}, \hat{r}_{\mathcal{P}_{\max}}$
6: **Set** $\hat{r}_{\widetilde{\mathcal{P}},q}$ according to (1)
7: **Wait** until $\hat{r}_{\widetilde{\mathcal{P}},q}$ is detected
8: $q \leftarrow q + 1$
9: **end while**

B. Algorithms to Select Goal Points

The UAV's reference path is computed iteratively after identifying a sequence of goal points according to Algorithm 1. The first step of Algorithm 1 is to partition $\mathcal{P}_0$ by employing Algorithm 2, which will be discussed later. The next step is to find sufficiently unexplored partitions $\left\{\widetilde{\mathcal{P}}_1, \ldots, \widetilde{\mathcal{P}}_{\tilde{n}_{\mathcal{P}}}\right\} \subseteq \{\mathcal{P}_1, \ldots, \mathcal{P}_{n_{\mathcal{P}}}\}$ such that $\frac{n_{\text{unexplored},\tilde{\mathcal{P}}_i}}{n_{\tilde{\mathcal{P}}_i}} \geq 1 - \mu_1$, $i = 1, \ldots, \tilde{n}_{\mathcal{P}}$, where $\mu_1 \in (0, 1)$ is user-defined. Finally, the UAV's *goal point* $\hat{r}_{\widetilde{\mathcal{P}},q}$, is defined as the barycenter of the partition $\widetilde{\mathcal{P}} \in \left\{\widetilde{\mathcal{P}}_1, \ldots, \widetilde{\mathcal{P}}_{\tilde{n}_{\mathcal{P}}}\right\}$ that is accessible to the UAV and contains the point

$$\mu_2 \hat{r}_{\mathcal{P}_{\text{prox}}} + (1 - \mu_2) \hat{r}_{\mathcal{P}_{\max}}, \tag{1}$$

where $\mu_2 \in [0, 1]$ is user-defined, $\hat{r}_{\mathcal{P}_{\text{prox}}}$ denotes the barycenter of the sufficiently unexplored partition that is closest to the UAV along its direction of motion, $\hat{r}_{\mathcal{P}_{\max}}$ denotes the sufficiently unexplored partition containing the largest number of unexplored voxels. Partitions and goal points are recomputed as soon as the UAV's navigation system maps the voxel containing $\hat{r}_{\widetilde{\mathcal{P}},q}$. If there is no partition whose ratio of explored voxels to total voxels is smaller than μ_1, then the voxel map is considered sufficiently covered.

At each iteration of Algorithm 1, $\mathcal{P}_0 \subseteq \mathcal{V}$ is partitioned in $n_{\mathcal{P}}$ rectangular parallelepipeds according to Algorithm 2. In Algorithm 2, if two sets of conditions are verified, then $\mathcal{P}_0$ or any of its partitions are divided into smaller parallelepipeds, whose aspect ratios are the same as the aspect ratio of $\mathcal{P}_0$. The first set of these conditions requires that both the ratio of explored voxels over the total number of voxels in $\mathcal{P}_i$, $i = 0 \ldots, n_{\mathcal{P}}$, is smaller than the user-defined parameter $\mu_1 \in (0, 1)$ and the length of the smallest side of $\mathcal{P}_i$ is larger than $2\mu_3$, where $\mu_3 > 0$ is user-defined. The second set of conditions requires that the number of explored voxels in $\mathcal{P}_i$, $i = 0 \ldots, n_{\mathcal{P}}$, is larger than the user-defined parameter $\mu_4 \in \mathbb{N}$ or the ratio of occupied voxels over the total number of voxels in $\mathcal{P}_i$ is larger than the parameter $\mu_5 \in (0, 1)$.

C. Numerical Solution of the Path Planning Problem

Given the sequence of goal points $\{\hat{r}_{\widetilde{\mathcal{P}},q}\}_{q=0}^{n_{\text{g}}}$, the proposed path planning algorithm generates the *UAV's reference path*

Algorithm 2 Octree iterative algorithm to partition $\mathcal{P}_0$

Initialize $\mathcal{P}_0 =$ minimum bounding box, $n_{\mathcal{P}} = 1$
if $\frac{n_{\text{explored},\mathcal{P}_0}}{n_{\mathcal{P}_0}} < \mu_1$ **then**
  Divide$(\mathcal{P}_0)$
end if
procedure DIVIDE$(\mathcal{P}_i)$
  Compute child parallelipipeds
  $n_{\mathcal{P}} \leftarrow n_{\mathcal{P}} + 8$
  for $j = (n_{\mathcal{P}} - 7)$:$n_{\mathcal{P}}$ **do**
    if $\frac{n_{\text{explored},\mathcal{P}_j}}{n_{\mathcal{P}_j}} < \mu_1$ **and** $\frac{\ell_{\mathcal{P}_j,\min}}{2} \geq \mu_3$ **then**
      if $n_{\text{explored},\mathcal{P}_j} \geq \mu_4$ **or** $\frac{n_{\text{occupied},\mathcal{P}_j}}{n_{\mathcal{P}_j}} \geq \mu_5$ **then**
        Divide$(\mathcal{P}_j)$
      end if
    end if
  end for
end procedure

as the sequence $\{\hat{r}_k\}_{k=0}^{n_{\text{w}}} \subset \mathbb{R}^3 \backslash \mathcal{O}$ such that $\hat{r}_0$ is the UAV's position when the planner is initialized, and $\hat{r}_{n_{\text{w}}} = \hat{r}_{\widetilde{\mathcal{P}},q+1}$. The sequence $\{\hat{r}_k\}_{k=0}^{n_{\text{w}}}$ minimizes the cost function

$$f_{k,q} \triangleq g_k + h_{k,q}, \quad k \in \{1, \ldots, n_{\text{w}}\}, \tag{2}$$

where

$$g_k \triangleq \sum_{p=1}^{k} [\kappa(\text{d}_2(\hat{r}_p, \mathcal{O}))\text{d}_2(\hat{r}_p, \hat{r}_{p-1})] \tag{3}$$

denotes the *cost-to-come function*,

$$h_{k,q} \triangleq \mu_8 \text{d}_2\left(\hat{r}_k, \hat{r}_{\widetilde{\mathcal{P}},q}\right) \tag{4}$$

denotes the *heuristic function*,

$$\kappa(\alpha) \triangleq \begin{cases} \mu_8 + 0.5(1-\mu_8)\left[1 + \cos\frac{2\pi(\alpha-\mu_6)}{\mu_7-\mu_6}\right], \alpha \in [\mu_6, \mu_7], \\ 1, \qquad \alpha \in [0, \mu_6) \cup (\mu_7, \infty), \end{cases} \tag{5}$$

denotes the *weighing function*, and $\mu_7 > \mu_6 > 0$ and $\mu_8 \in [0, 1)$ are user-defined parameters, and $d_2 : \mathbb{R}^3 \times \mathbb{R}^3 \to \mathbb{R}$ captures the distance between two points in $\mathbb{R}^3$. The cost function (2) is the weighted sum of the length of the UAV's reference path, namely (3), and an under-estimate of the Euclidean distance between the voxel occupied by the UAV and the goal point $\hat{r}_{\widetilde{\mathcal{P}},q}$, namely (4).

The role of the function $\kappa(\cdot)$ is to encourage tactical behaviors by rewarding paths that are closer to the obstacles' set $\mathcal{O}$. The minimum of this continuous function is equal to μ_8 and its minimizer is given by $\alpha = (\mu_6 + \mu_7)/2$. Consequently, smaller values of μ_8 increase the attractiveness of the obstacles' set and induce a more tactical behavior. Additionally, $\kappa(\alpha) < 1$, $\alpha \in [\mu_6, \mu_7]$, therefore, $[\mu_6, \mu_7]$ is a measure of the region of influence of the obstacles' set. Additionally, the parameter μ_6 can be used to impose a safety margin.

The UAV's reference path $\{\hat{r}_k\}_{k=0}^{n_{\text{w}}}$ can be deduced by any heuristics-based search algorithm over graphs, such as, for instance, LPA^* [11], by finding minimizers of (2) subject to the constraint of allowing the UAV to move across adjacent unoccupied voxels. Since μ_8 scales the distance between the UAV's current position and the next goal point $\hat{r}_{\widetilde{\mathcal{P}},q}$, in (4), and $\kappa(\alpha) \geq \mu_8$ for all $\alpha \geq 0$, by the triangle inequality, $h_{k,q}$ is a consistent heuristic function. In this paper, the LPA^* algorithm is selected to solve the path planning problem. See [12, Ch. 3] for a discussion and analysis of the benefits of LPA^*.

III. A Trajectory Planner for Tactical Coverage

A. Notation

Time is denoted by $t \geq 0$. Third-order derivatives with respect to time are denoted by $(\cdot)^{(3)}$. We assume that the UAV is able to fly from $\hat{r}_k$ to $\hat{r}_{k+1}$ in $n_{\text{t}}\Delta T$ time units, where both $n_{\text{t}} \in \mathbb{N}$ and $\Delta T > 0$ are user-defined parameters capturing the length of the planning horizon. Let $\nu_{\text{s},k} \in \{1, \ldots, n_{\text{w}} - k\}$ denote the user-defined *path stride*. The integer $i \in \{0, \ldots, n_{\text{t}}\nu_{\text{s},k}\}$ is employed to count segments of MPC's time horizon, the integer $j \in \{i, \ldots, n_{\text{t}}\nu_{\text{s},k}\}$ is employed to capture the time step $j\Delta T$ in the interval $[i\Delta T, n_t\Delta T]$ unless otherwise stated.

The *UAV's position* is captured by $r_k : [0, n_{\text{t}}\Delta T] \to \mathbb{R}^3 \setminus \mathcal{O}$, expressed in a conveniently located inertial reference frame $\mathbb{I}$. The *UAV's roll angle* is denoted by $\phi_k : [0, n_{\text{t}}\Delta T] \to \left(-\frac{\pi}{2}, \frac{\pi}{2}\right)$, $k \in \{0, \ldots, n_{\text{w}} - 1\}$, the *UAV's pitch angle* is denoted by $\theta_k : [0, n_{\text{t}}\Delta T] \to \left(-\frac{\pi}{2}, \frac{\pi}{2}\right)$, the *UAV's yaw angle* is denoted by $\psi_k : [0, n_{\text{t}}\Delta T] \to [0, 2\pi)$, the *UAV's velocity* with respect to $\mathbb{I}$ is denoted by $v_k : [0, n_{\text{t}}\Delta T] \to \mathbb{R}^3$, the UAV's *angular velocity* with respect to $\mathbb{I}$ is denoted by $\omega_k : [0, n_{\text{t}}\Delta T] \to \mathbb{R}^3$, and the UAV's *state vector* is denoted by $x_k(j\Delta T) \triangleq \left[r_k^{\text{T}}(j\Delta T), \phi(j\Delta T), \theta(j\Delta T), \psi(j\Delta T), v_k^{\text{T}}(j\Delta T), \omega_k^{\text{T}}(j\Delta T)\right]^{\text{T}}$.

Let $\eta_{1,k} : [0, n_{\text{t}}\Delta T] \to \mathbb{R}$, denote the *total thrust force's virtual control input*, the *total thrust force* produced by the UAV's propellers is defined as $u_{1,k}(t)$, $t \in [0, n_{\text{t}}\Delta T]$, and

$$\begin{bmatrix} \dot{u}_{1,k}(t) \\ \ddot{u}_{1,k}(t) \end{bmatrix} = \begin{bmatrix} 0 & 1 \\ 0 & 0 \end{bmatrix} \begin{bmatrix} u_{1,k}(t) \\ \dot{u}_{1,k}(t) \end{bmatrix} + \begin{bmatrix} 0 \\ 1 \end{bmatrix} \eta_{1,k}(t),$$
$$\begin{bmatrix} u_{1,k}(0) \\ u_{1,k}(n_{\text{t}}\Delta T) \end{bmatrix} = \begin{bmatrix} u_{1,0,k} \\ u_{1,\text{f},k} \end{bmatrix}, \quad t \in [0, n_{\text{t}}\Delta T]. \tag{6}$$

The *roll moment* produced by the UAV's propellers is denoted by $u_{2,k}(\cdot)$, the *pitch moment* produced by the UAV's propellers is denoted by $u_{3,k}(\cdot)$, and the *yaw moment* produced by the UAV's propellers is denoted by $u_{4,k}(\cdot)$. The *UAV's control input* is defined as $u_k(t) \triangleq [u_{1,k}(t), u_{2,k}(t), u_{3,k}(t), u_{4,k}(t)]^{\text{T}}$, $t \in [0, n_{\text{t}}\Delta T]$. The *vector of thrust forces* produced by each propeller is defined as

$$T_k(t) \triangleq M_{T,u} u_k(t), \qquad t \in [0, n_{\text{t}}\Delta T], \tag{7}$$

where the ith component of $T_k(\cdot)$, $i = 1, \ldots, 4$, namely $T_{i,k} : [0, n_{\text{t}}\Delta T] \to [0, \infty)$, denotes the thrust force produced by the ith propeller, and $M_{T,u}$ is the UAV's mixer.

B. Output-Feedback Linearized Equations of Motion

Neglecting the aerodynamic drag, the inertial counter-torque, and the gyroscopic effect [13], the UAV's equations of motion are given by

$$\dot{r}_k(t) = v_k(t),\ r_k(0) = r_{\mathrm{init},k},\ t \in [0, n_{\mathrm{t}}\nu_{\mathrm{s},k}\Delta T], \quad (8a)$$

$$\dot{v}_k(t) = m^{-1}R(\phi_k(t), \theta_k(t), \psi_k(t))[0, 0, u_{1,k}(t)]^{\mathrm{T}} - [0, 0, g]^{\mathrm{T}},\ \ r_k(n_{\mathrm{t}}\nu_{\mathrm{s},k}\Delta T) = r_{\mathrm{end},k}, \quad (8b)$$

$$\begin{bmatrix}\dot{\phi}_k(t)\\ \dot{\theta}_k(t)\\ \dot{\psi}_k(t)\end{bmatrix} = \Gamma^{-1}(\phi_k(t), \theta_k(t))\omega_k(t),\ \ \begin{bmatrix}\phi_k(0)\\ \theta_k(0)\\ \psi_k(0)\end{bmatrix} = \begin{bmatrix}\phi_{0,k}\\ \theta_{0,k}\\ \psi_{0,k}\end{bmatrix}, \quad (8c)$$

$$\dot{\omega}_k(t) = I^{-1}\left(\begin{bmatrix}u_{2,k}(t)\\ u_{3,k}(t)\\ u_{4,k}(t)\end{bmatrix} - \omega_k^{\times}(t)I\omega_k(t)\right),\ \ \begin{bmatrix}\phi_k(n_{\mathrm{t}}\nu_{\mathrm{s},k}\Delta T)\\ \theta_k(n_{\mathrm{t}}\nu_{\mathrm{s},k}\Delta T)\\ \psi_k(n_{\mathrm{t}}\nu_{\mathrm{s},k}\Delta T)\end{bmatrix} = \begin{bmatrix}\phi_{\mathrm{f},k}\\ \theta_{\mathrm{f},k}\\ \psi_{\mathrm{f},k}\end{bmatrix}, \quad (8d)$$

where $R(\phi_k, \theta_k, \psi_k)$, $(\phi_k, \theta_k, \psi_k) \in \left(-\frac{\pi}{2}, \frac{\pi}{2}\right) \times \left(-\frac{\pi}{2}, \frac{\pi}{2}\right) \times [0, 2\pi)$, is a 321 rotation sequence of Tait-Bryan angles capturing the UAV's attitude relative to the inertial reference frame $\mathbb{I}$, and $\Gamma(\phi_k, \theta_k)$, $(\phi_k, \theta_k) \in \left(-\frac{\pi}{2}, \frac{\pi}{2}\right) \times \left(-\frac{\pi}{2}, \frac{\pi}{2}\right)$ is the Jacobian of the rotational kinematics [14, Ch. 2].

In this paper, we are interested in controlling directly the UAV's position $r_k(\cdot)$ and its yaw angle $\psi_k(\cdot)$ since the cameras' focal axes are aligned with the UAV's roll axis. Thus, setting $\left[r_k^{\mathrm{T}}(t), \psi_k(t)\right]^{\mathrm{T}}$, $t \in [0, n_{\mathrm{t}}\nu_{\mathrm{s},k}\Delta T]$, as the *measured output*, and proceeding as in [15, Prop. 5.1.2], we verify that the dynamical system given by (8) and (6) has vector relative degree $\{4, 4, 4, 2\}$, and the UAV's output-feedback linearized equations of motion are

$$\dot{\chi}_k(t) = \tilde{A}\chi_k(t) + \tilde{B}\lambda_k(t), \qquad t \in [0, n_{\mathrm{t}}\nu_{\mathrm{s},k}\Delta T], \quad (9)$$

where $\chi_k(t) \triangleq \left[r_k^{\mathrm{T}}(t), \dot{r}_k^{\mathrm{T}}(t), \ddot{r}_k^{\mathrm{T}}(t), r_k^{(3)\mathrm{T}}(t), \psi_k(t), \dot{\psi}_k(t)\right]^{\mathrm{T}}$, $\lambda_k \in \mathbb{R}^4$ denotes a *virtual control input*, $\tilde{A} \triangleq \mathrm{blockdiag}(\tilde{A}_r, \tilde{A}_\psi)$, $\tilde{B} \triangleq \left[\tilde{B}_r^{\mathrm{T}}, \tilde{B}_\psi^{\mathrm{T}}\right]^{\mathrm{T}}$,

$$\tilde{A}_r \triangleq \begin{bmatrix}0_{3\times3} & \mathbf{1}_3 & 0_{3\times3} & 0_{3\times3}\\ 0_{3\times3} & 0_{3\times3} & \mathbf{1}_3 & 0_{3\times3}\\ 0_{3\times3} & 0_{3\times3} & 0_{3\times3} & \mathbf{1}_3\\ A_{r,0} & A_{r,1} & A_{r,2} & A_{r,3}\end{bmatrix} \in \mathbb{R}^{12\times12}, \quad (10)$$

$$\tilde{A}_\psi \triangleq \begin{bmatrix}0 & 1\\ A_{\psi,0} & A_{\psi,1}\end{bmatrix} \in \mathbb{R}^{2\times2}, \quad (11)$$

are Hurwitz, $A_{r,i} \in \mathbb{R}^{3\times3}$, $i \in \{0, \ldots, 3\}$, $A_{\psi,0} \in \mathbb{R}$, and $A_{\psi,1} \in \mathbb{R}$ are user-defined,

$$\tilde{B}_r \triangleq \begin{bmatrix}0_{9\times4}\\ B_r\end{bmatrix} \in \mathbb{R}^{12\times4}, \qquad \tilde{B}_\psi \triangleq \begin{bmatrix}0\\ B_\psi\end{bmatrix} \in \mathbb{R}^{2\times4} \quad (12)$$

and the pairs $(\tilde{A}_r, \tilde{B}_r)$ and $(\tilde{A}_\psi, \tilde{B}_\psi)$ are controllable.

It follows from (9) that the discrete-time, linearized, zero-order hold [16], output-feedback linearized equations of motion of a quadcopter UAV are given by

$$\chi_k((j+1)\Delta T) = A\chi_k(j\Delta T) + B\lambda_k(j\Delta T), \quad (13)$$

where $A = e^{\tilde{A}\Delta T}$ and $B = \int_0^{\Delta T} e^{\tilde{A}\sigma}\mathrm{d}\sigma\tilde{B}$. Equation (13) captures equality constraints for the MPC algorithm employed to outline tactical reference trajectories for UAVs.

Given $\lambda_k(j\Delta T)$, the total thrust force's virtual control input $\eta_{1,k}(j\Delta T)$, the roll moment produced by the UAV's propellers $u_{2,k}(j\Delta T)$, the pitch moment $u_{3,k}(j\Delta T)$, and the yaw moment $u_{4,k}(j\Delta T)$ are computed as

$$[\eta_{1,k}(j\Delta T), u_{2,k}(j\Delta T), u_{3,k}(j\Delta T), u_{4,k}(j\Delta T)]^{\mathrm{T}} = \zeta_k(j\Delta T), \quad (14)$$

where

$$\zeta_k \triangleq G(r_k, \phi_k, \theta_k, \psi_k, \omega_k, u_{1,k})\Big(-f(r_k, \phi_k, \theta_k, \psi_k, \omega_k, u_{1,k}) + \begin{bmatrix}A_{r,0}r_k(t) + A_{r,1}\dot{r}_k(t) + A_{r,2}\ddot{r}_k(t) + A_{r,3}r_k^{(3)}(t)\\ A_{\psi,0}\psi_k(t) + A_{\psi,1}\dot{\psi}_k(t)\end{bmatrix} + \begin{bmatrix}B_r\\ B_\psi\end{bmatrix}\lambda_k\Big), \quad (15)$$

for all $(r_k, \phi_k, \theta_k, \psi_k, \omega_k, u_{1,k}, \lambda_k) \in \mathbb{R}^3 \times \left(-\frac{\pi}{2}, \frac{\pi}{2}\right) \times \left(-\frac{\pi}{2}, \frac{\pi}{2}\right) \times [0, 2\pi) \times \mathbb{R}^3 \times \mathbb{R} \times \mathbb{R}^4$; expressions for $f(\cdot)$ and $G(\cdot)$ are omitted for brevity and can be found by applying Proposition 5.1.2 of [15], and are recomputed at every time-step. Note that $\lambda_k(\cdot)$ is the input of the virtual system (9), and $\zeta_k(\cdot)$ is the input of the real system (8a)–(8d). The discrete-time zero-order hold counterpart of (6) is

$$\delta_k((j+1)\Delta T) = \begin{bmatrix}1 & 1\\ 0 & 1\end{bmatrix}\delta_k(j\Delta T) + \frac{1}{2}\begin{bmatrix}\Delta T^2\\ 2\Delta T\end{bmatrix}\eta_{1,k}(j\Delta T). \quad (16)$$

Finally, the thrust force produced by each propeller is computed by applying (7).

C. Constraints on Collision Avoidance and Yaw Angle

An affine constraint set, which contains the UAV and excludes all obstacle points, is captured by

$$F_{r,k}(i\Delta T)r_k(i\Delta T) \leq\leq f_{r,k}(i\Delta T), \quad (17)$$

where $\leq\leq$ denotes the component-wise inequality, and $F_{r,k} : \mathbb{R} \to \mathbb{R}^{n_{\mathrm{coll}}\times3}$ and $f_{r,k} : \mathbb{R} \to \mathbb{R}^{n_{\mathrm{coll}}}$ can be deduced from a voxel map of the environment by proceeding as in [10]. Constraints on the UAV's yaw angle are captured by

$$F_\psi(i\Delta T)\psi_k(j\Delta T) \leq\leq f_{\psi,k}(i\Delta T), \quad (18)$$

where $F_\psi(i\Delta T) \triangleq [-1, 1]^{\mathrm{T}}$, $f_{\psi,k}(i\Delta T) \triangleq [\psi_{\max} - \psi_{\mathrm{f},k}, \psi_{\max} + \psi_{\mathrm{f},k}]^{\mathrm{T}}$. The MPC algorithm searches reference trajectories that verify (13), (17), and (18).

D. Cost Function Definition

The proposed trajectory planner produces reference trajectories that minimize the *cost function*

$$\begin{aligned}&\tilde{J}[r_k(0), \psi_k(0), \lambda_k(\cdot)]\\ &\triangleq \sum_{l=k}^{k+\nu_{\mathrm{s},k}-1}\nu_{\mathrm{dis}}^{l-k}\ell_{\mathrm{f}}(r_l(n_{\mathrm{t}}\Delta T), \psi_l(n_{\mathrm{t}}\Delta T))\\ &+ \sum_{l=k}^{k+\nu_{\mathrm{s},k}-1}\sum_{i=0}^{n_{\mathrm{t}}-1}\nu_{\mathrm{dis}}^{l-k}\tilde{\ell}(i\Delta T, r_l(i\Delta T), \psi_l(i\Delta T), \lambda_l(i\Delta T)),\end{aligned} \quad (19)$$

where $\nu_{\mathrm{dis}} \in (0,1)$,

$$\tilde{\ell}(\tau, r_k, \psi_k, \lambda_k) \triangleq \begin{bmatrix} \tilde{r}_k \\ \lambda_k \end{bmatrix}^{\mathrm{T}} \tilde{R}_k(\tau) \begin{bmatrix} \tilde{r}_k \\ \lambda_k \end{bmatrix} + q_\psi(\psi_k - \psi_{\mathrm{f},k})^2 + \tilde{q}_r^{\mathrm{T}} \tilde{r}_k + \tilde{q}_\lambda^{\mathrm{T}} \lambda_k,$$
$$(\tau, r_k, \psi_k, \lambda_k) \in [0, n_{\mathrm{t}} \nu_{\mathrm{s},k} \Delta T] \times \mathbb{R}^3 \times \mathbb{R} \times \mathbb{R}^4, \quad (20)$$

$$\ell_{\mathrm{f}}(r_k, \psi_k) \triangleq \left(r_k - \hat{r}_{k+\nu_{\mathrm{s},k}}\right)^{\mathrm{T}} R_{r,\mathrm{f}} \left(r_k - \hat{r}_{k+\nu_{\mathrm{s},k}}\right) + q_\psi(\psi_k - \psi_{\mathrm{f},k})^2, \quad (21)$$

$\tilde{R}_k(\tau) \triangleq g_{\mathrm{barrier},k}^{-1}(\tau) \begin{bmatrix} \tilde{R}_r & \tilde{R}_{r,\lambda} \\ \tilde{R}_{r,\lambda}^{\mathrm{T}} & R_\lambda \end{bmatrix}$, $\tilde{R}_r \in \mathbb{R}^{3\times 3}$ is symmetric, $\tilde{R}_{r,\lambda} \in \mathbb{R}^{3\times 4}$, and $R_\lambda \in \mathbb{R}^{4\times 4}$ is positive-definite and such that $\tilde{R}_r - 2\tilde{R}_{r,\lambda}^{\mathrm{T}} R_\lambda^{-1} \tilde{R}_{r,\lambda} > 0$, $\tilde{q}_r \in \mathbb{R}^3$, $\tilde{q}_\lambda \in \mathbb{R}^4$, $q_\psi > 0$,

$$\tilde{r}_k(i\Delta T) \triangleq (1 - \mu_{13}) f_{\mathrm{sat}}\left(\mu_{14}(\hat{r}_k - r_{\mathcal{O}})\right) \cdot [r_k(i\Delta T) - r_{\mathcal{O}}] + \mu_{13}\left[r_k(i\Delta T) - \hat{r}_{k+\nu_{\mathrm{s},k}}\right], \quad (22)$$

$\mu_{13} \in (0,1]$ and $\mu_{14} > 0$ are user-defined, $f_{\mathrm{sat}}(w) \triangleq \dfrac{\mathrm{sat}\left(\|w\|\right)}{\|w\|}$, $w \in \mathbb{R}^n$, $r_{\mathcal{O}} \triangleq \mathrm{d}_2(\hat{r}_k, \mathcal{O})$, and $R_{r,\mathrm{f}} \in \mathbb{R}^{3\times 3}$ is symmetric and nonnegative-definite,

$$g_{\mathrm{barrier},k}(j\Delta T) \triangleq \left[\phi_{\max}^2 - \phi_k^2(j\Delta T)\right]\left[\theta_{\max}^2 - \theta_k^2(j\Delta T)\right] \cdot \prod_{p=1}^{4} [T_{p,k}(j\Delta T) - T_{\min}] \prod_{p=1}^{4} \left[T_{\max} - T_{p,k}(j\Delta T)\right], \quad (23)$$

and $\phi_{\max} \in \left(0, \frac{\pi}{2}\right)$ is the maximum roll angle, $\theta_{\max} \in \left(0, \frac{\pi}{2}\right)$ is the maximum pitch angle, and $0 < T_{\min} < T_{\max}$ are user-defined minimum and maximum thrust forces.

As discussed in Section III-E, boundary conditions are enforced for the trajectory planning problem at the waypoint $\hat{r}_{k+\nu_{\mathrm{s},k}}$. Thus, the reference trajectory is not imposed to traverse the waypoints $\hat{r}_{k+1}, \ldots, \hat{r}_{k+\nu_{\mathrm{s},k}-1}$. The cost function (19) captures the need to outline a reference trajectory that approximates the waypoints $\hat{r}_{k+1}, \ldots, \hat{r}_{k+\nu_{\mathrm{s},k}-1}$. The term ν_{dis}^{l-k}, in (19) is a *discount factor* applied to the objective function so that, while the UAV approaches $\hat{r}_{k+1}$, the influence of $\hat{r}_{k+2}, \ldots, \hat{r}_{k+\nu_{\mathrm{s},k}}$ on the cost function is less marked.

The *Mayer's function* (21) captures the UAV's need to reach the waypoint $\hat{r}_{k+\nu_{\mathrm{s},k}}$, and point its roll axis toward this waypoint. The first term on the right-hand side of (22) captures the UAV's distance from $\hat{r}_{k+\nu_{\mathrm{s},k}}$, and the second term on the right-hand side of (22) captures the UAV's distance from the obstacles.

The *Lagrangian function* (20) captures the UAV's competing needs of reaching the waypoint $\hat{r}_{k+\nu_{\mathrm{s},k}}$, and coasting the obstacles' set, while pointing the onboard cameras toward $\hat{r}_{k+\nu_{\mathrm{s},k}}$. Indeed, if $\mu_{13} = 1$, then $\tilde{r}_k(i\Delta T) = r_k(i\Delta T) - \hat{r}_{k+\nu_{\mathrm{s},k}}$, $i \in \{0, \ldots, n_{\mathrm{t}}\nu_{\mathrm{s},k} - 1\}$, and minimizing (19) induces a reckless behavior in the UAV, since its only goal is to reach the waypoint, and if smaller values of $\mu_{13} \in (0,1)$ make coasting the obstacles' set a higher priority. The function $f_{\mathrm{sat}}(\cdot)$ in (22) reduces the attractive effect of obstacles at a distance from the waypoint $\hat{r}_k$, that is larger than μ_{14}^{-1}.

See [12, Ch. 3] for a discussion on the impact of $g_{\mathrm{barrier}}(\cdot)$.

E. Boundary Conditions

The conditions on the UAV's final position must be set to ensure that the reference trajectory traverses the waypoint $\hat{r}_{k+\nu_{\mathrm{s},k}}$. The condition on the UAV's final acceleration are defined so that $\ddot{r}_{\mathrm{end},k} = \hat{a}_{k+\nu_{\mathrm{s},k}}$ and

$$\hat{a}_{k+\nu_{\mathrm{s},k}} \triangleq \hat{a}\left(\mathrm{d}_2(\hat{r}_{k+\nu_{\mathrm{s},k}}, \mathcal{O})\right) \hat{d}_k, \quad (24)$$

where

$$\hat{a}(\alpha) \triangleq \begin{cases} 0, & \text{if } \alpha \in [0, \mu_{15}], \\ \frac{\mu_{17}}{\mu_{16} - \mu_{15}}(\alpha - \mu_{15}), & \text{if } \alpha \in [\mu_{15}, \mu_{16}], \\ \mu_{17}, & \text{otherwise}, \end{cases} \quad (25)$$

$$\hat{d}_k \triangleq \mu_{18} \bar{d}(\hat{r}_{k+\nu_{\mathrm{s},k}}) + (1 - \mu_{18}) \bar{d}(\hat{r}_{k+\nu_{\mathrm{s},k}+1}), \quad (26)$$

and $\mu_{15}, \mu_{16}, \mu_{17} > 0$ and $\mu_{18} \in [0,1]$ are user-defined, and $\bar{d}(\hat{r}_n)$ captures the direction of the line joining $\hat{r}_n$ and $\hat{r}_{n-1}$.

If $\mathrm{d}_2(\hat{r}_{k+\nu_{\mathrm{s},k}}, \mathcal{O}) \leq \mu_{15}$, that is, if the UAV is sheltered by an obstacle at a distance smaller than μ_{15}, then $\|\ddot{r}_{\mathrm{end},k}\| = 0$, since it is desirable for the UAV to maintain its speed while operating near shelter as it passes through the waypoint. Alternatively, if $\mathrm{d}_2(\hat{r}_{k+\nu_{\mathrm{s},k}}, \mathcal{O}) \in [\mu_{15}, \mu_{16}]$, then $\|\ddot{r}_{\mathrm{end},k}\|$ is set to increase as a linear function of $\mathrm{d}_2(\hat{r}_{k+\nu_{\mathrm{s},k}}, \mathcal{O})$ from 0 to μ_{17}. Finally, if $\mathrm{d}_2(\hat{r}_{k+\nu_{\mathrm{s},k}}, \mathcal{O}) > \mu_{16}$, that is, if the UAV is too far from any obstacle to be considered as sheltered, then $\|\ddot{r}_{\mathrm{end},k}\| = \mu_{17}$, so the UAV will accelerate and use its speed to avoid detection or interception. If $\mu_{18} = 1$, then $\ddot{r}_{\mathrm{end},k}$, is set to point in the direction joining the waypoint $\hat{r}_{k+\nu_{\mathrm{s},k}}$ and $\hat{r}_{k+\nu_{\mathrm{s},k}-1}$. Alternatively, if $\mu_{18} = 0$, then $\ddot{r}_{\mathrm{end},k}$, is set to point in the direction joining the waypoint $\hat{r}_{k+\nu_{\mathrm{s},k}}$, which is the trajectory's endpoint, and $\hat{r}_{k+\nu_{\mathrm{s},k}+1}$ the next trajectory's endpoint.

F. Numerical Solution of the Trajectory Planning Problem

In this paper, the pair $(\chi_k(j\Delta T), \lambda_k(j\Delta T))$ is computed as minimizers of the cost function (19) subject to (13), (17), and (18) by applying the framework presented in [10]. A difference with [10] is that in this work, the cost function underlying the trajectory planner is a function of time due to the barrier function used to impose constraints on the thrust, pitch and roll angles. Thus, the quadratic programming algorithm to compute reference trajectories needs to be updated at each time step.

IV. Numerical Simulations Results

Two simulations were performed to validate the system with different settings in a simulated indoor environment; see Figures 2 and 3 for details. A simulated camera fixed to the UAV's position and aligned with its roll axis observes the environment. In the first simulation, the proposed guidance system is set to induce a tactical behavior and greedy goal selection. Furthermore, the proposed goal selection algorithm was set to instill a greedy behavior, and less backand- forth motion is evident. In the second simulation, the proposed guidance system is set to induce reckless guidance and systematic goal selection. Furthermore, the goal selection

TABLE I

TABLE LISTING QUANTITIES FROM THE TWO SIMULATIONS.

Parameter Sets	Sim. 1	Sim. 2
Avg. dist. from $\mathcal{O}$ [m]	1.02 $\pm$ 0.94	3.92 $\pm$ 2.78
Dist. traveled [m]	221.85	129.57
Time [s]	363.65	306.52
Expl. rate [voxels/s]	445.87	531.58

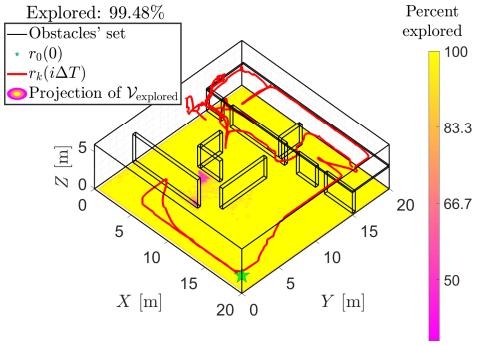

Fig. 2. Results from simulation 1 with tactical guidance and greedy goal selection parameter sets.

algorithm was set to instill a systematic behavior, and back-and-forth motion is prevalent which may exacerbate the vehicle's vulnerability to detection. Notably, in both simulations, the UAV must map at least 97% of the assigned volume, that is, we set $\mu_1 = 0.97$.

Table I lists key information about the two simulations. Tactical and greedy settings induce longer trajectories, longer mission times, and slower mapping rates than reckless and systematic settings, However, tactical and greedy settings induce the UAV to travel considerably closer to the obstacles' set $\mathcal{O}$, and, hence, the UAV shows a safer behavior.

Both in Figure 2 and in Figure 3, a color map indicates the projection of $\mathcal{V}_{\text{explored}}$ onto the horizontal plane of the three-dimensional plot. It is apparent how, following a tactical trajectory, the assigned volume is almost completely covered. Following a reckless trajectory, the UAV only needs to travel from the South-East quadrant, to the center, then to South-West, then to North-East, and finally to North-West to complete its mission.

V. CONCLUSION

This paper presented a novel guidance system for autonomous multi-rotor UAVs employed to cover unknown areas, while implementing several tactics to minimize the risk of exposure to unknown, potential threats. These tactics include exploring obstacles to seek shelter and proceeding more slowly in safer areas, such as in proximity of obstacles; which are enabled through a path planner and trajectory planner. Numerical simulations showed the applicability of the proposed system in a simulated indoor space with no *a priori* knowledge of the environment.

ACKNOWLEDGEMENT

This work was supported in part by DARPA under the Grant no. D18AP00069.

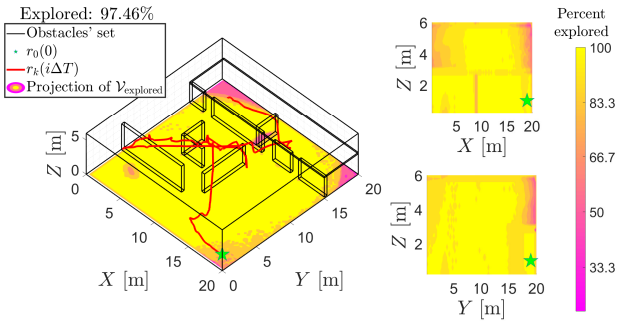

Fig. 3. Results from simulation 2 with reckless guidance and systematic goal selection parameter sets.

REFERENCES

[1] J. A. Marshall, W. Sun, and A. L'Afflitto, "A survey of guidance, navigation, and control systems for autonomous multi-rotor small unmanned aerial systems," *Annual Reviews in Control*, vol. 52, pp. 390–427, 2021.

[2] X. Zhou, Z. Yi, Y. Liu, K. Huang, and H. Huang, "Survey on path and view planning for UAVs," *Virtual Reality & Intelligent Hardware*, vol. 2, no. 1, pp. 56–69, 2020.

[3] R. J. Wallace and J. M. Loffi, "How law enforcement unmanned aircraft systems (UAS) could improve tactical response to active shooter situations: The case of the 2017 Las Vegas shooting," *International Journal of Aviation, Aeronautics, and Aerospace*, vol. 4, no. 4, p. 7, 2017.

[4] L. Moysis, E. Petavratzis, C. Volos, H. Nistazakis, I. Stouboulos, and K. Valavanis, "A chaotic path planning method for 3D area coverage using modified logistic map and a modulo tactic," in *International Conference on Unmanned Aircraft Systems*. IEEE, 2020, pp. 220–227.

[5] A. Wallar, E. Plaku, and D. A. Sofge, "A planner for autonomous risk-sensitive coverage (PARCov) by a team of unmanned aerial vehicles," in *IEEE Symposium on Swarm Intelligence*. IEEE, 2014, pp. 1–7.

[6] P. He and S. Dai, "Stealth coverage multi-path corridors planning for UAV fleet," in *International Conference on Mechatronic Sciences, Electric Engineering and Computer*. IEEE, 2013, pp. 2922–2926.

[7] C. Peng and V. Isler, "Adaptive view planning for aerial 3D reconstruction," in *International Conference on Robotics and Automation*. IEEE, 2019, pp. 2981–2987.

[8] K. Yamaguchi, T. Kunii, K. Fujimura, and H. Toriya, "Octree-related data structures and algorithms," *IEEE Computer Graphics and Applications*, vol. 4, no. 01, pp. 53–59, 1984.

[9] T. Li, C. Wang, C. W. de Silva *et al.*, "Coverage sampling planner for UAV-enabled environmental exploration and field mapping," in *International Conference on Intelligent Robots and Systems*. IEEE, 2019, pp. 2509–2516.

[10] J. A. Marshall, R. B. Anderson, W.-Y. Chien, E. N. Johnson, and A. L'Afflitto, "A guidance system for tactical autonomous unmanned aerial vehicles," *Journal of Intelligent & Robotic Systems*, vol. 103, no. 4, pp. 1–36, 2021.

[11] S. Koenig, M. Likhachev, and D. Furcy, "Lifelong planning A^*," *Artificial Intelligence*, vol. 155, no. 1, pp. 93 – 146, 2004.

[12] A. L'Afflitto, G. Inalhan, and S. Hyo-Sang, *Control of Autonomous Aerial Vehicles: Advances in Autopilot Design for Civilian UAVsl*. London, UK: Springer, 2023.

[13] A. L'Afflitto, R. B. Anderson, and K. Mohammadi, "An introduction to nonlinear robust control for unmanned quadrotor aircraft," *IEEE Control Systems Magazine*, vol. 38, no. 3, pp. 102–121, 2018.

[14] A. L'Afflitto, *A Mathematical Perspective on Flight Dynamics and Control*. London, UK: Springer, 2017.

[15] A. Isidori, *Nonlinear Control Systems*. New York, NY: Springer, 1995.

[16] J. S.-H. Tsai, C.-C. Huang, S.-M. Guo, and L.-S. Shieh, "Continuous to discrete model conversion for the system with a singular system matrix based on matrix sign function," *Applied Mathematical Modelling*, vol. 35, no. 8, pp. 3893–3904, 2011.