UAV-based Receding Horizon Control for 3D Inspection Planning

Savvas Papaioannou, Panayiotis Kolios, Theocharis Theocharides,
Christos G. Panayiotou and Marios M. Polycarpou

Abstract— **Nowadays, unmanned aerial vehicles or UAVs are being used for a wide range of tasks, including infrastructure inspection, automated monitoring and coverage. This paper investigates the problem of 3D inspection planning with an autonomous UAV agent which is subject to dynamical and sensing constraints. We propose a receding horizon 3D inspection planning control approach for generating optimal trajectories which enable an autonomous UAV agent to inspect a finite number of feature-points scattered on the surface of a cuboid-like structure of interest. The inspection planning problem is formulated as a constrained open-loop optimal control problem and is solved using mixed integer programming (MIP) optimization. Quantitative and qualitative evaluation demonstrates the effectiveness of the proposed approach.**

I. Introduction

Recent technological and scientific advancements in aerospace, avionics and artificial intelligence, in conjunction with the cost reduction of electronic parts and equipment, have made increasingly popular the use of unmanned aerial vehicles (UAVs) in various application domains. UAVs have found widespread utilization in various tasks including emergency response and search-and-rescue missions [1]–[5], precision agriculture [6], [7], wildlife monitoring [8], [9], and security [10], [11].

One of the most important functionalities that will further enable the utilization of fully autonomous UAVs in the application domains discussed above is that of trajectory planning [12], also known in the literature as motion/path planning [13]. This technology is of crucial importance in designing and executing automated UAV-based flight missions (i.e., automated UAV guidance, navigation and control), according to the requirements of the task at hand. In trajectory planning we are interested in delivering a collision-free motion between an initial and a final location within a given environment. Moreover, for many tasks including UAV-based automated maintenance operations, target search and infrastructure inspection, there is the need for finding the optimal trajectory which allows an autonomous UAV to utilize its sensors in order to cover or inspect every point within a given area or structure of interest. This problem is known in the literature as inspection planning (IP) or coverage path planning (CPP) [14], and is the focus of this work. More specifically, during an automated inspection mission the UAV agent must autonomously plan its inspection trajectory which allows the efficient coverage/inspection of all points of interest on a given structure, while satisfying the vehicle's dynamic and sensing constraints.

The authors are with the KIOS Research and Innovation Centre of Excellence (KIOS CoE) and the Department of Electrical and Computer Engineering, University of Cyprus, Nicosia, 1678, Cyprus. {papaioannou.savvas, pkolios, ttheocharides, christosp, mpolycar}@ucy.ac.cy

Although a plethora of inspection planning approaches have been proposed in the literature, the technology has not yet reached the required level of maturity to fully support autonomous UAV operations. Towards this direction, in this paper we investigate the UAV-based inspection planning problem in 3D environments for cuboid-like structures (or objects) of interest, such as buildings, that need to be fully inspected. More specifically, we assume that a finite number of feature-points are scattered throughout the surface area of the structure of interest and must be inspected by an autonomous UAV agent. The UAV agent evolves in 3D space according to its dynamical model and is equipped with a camera sensor which exhibits a dynamic sensing range i.e., the size of the camera's projected field-of-view (FOV) on a given surface is a function of the distance between the UAV and that surface. Based on these assumptions, we propose a receding horizon mixed integer programming (MIP) inspection planning controller, for optimally determining the UAV's motion control inputs within a finite rolling planning horizon, subject to the UAV's sensing capabilities. The UAV's inspection trajectory is generated on-line by solving at each time-step a constrained open-loop optimal control problem until all feature-points are inspected (i.e., all feature-points are viewed through the UAV's camera). Specifically, the contributions of this work are the following:

- We propose a receding horizon inspection planning control approach which allows an autonomous UAV agent, governed by dynamical and sensing constraints, to inspect in 3D environments cuboid-like structures of interest (e.g., buildings).
- The inspection planning problem is formulated as a constrained optimal control problem and solved over a finite rolling planning horizon using mixed integer programming (MIP) optimization, allowing the on-line generation of the UAV's optimal inspection trajectory.
- Qualitative and quantitative evaluation demonstrates the performance of the proposed approach.

The rest of the paper is organized as follows. Section II summarizes the related work on inspection planning with ground and aerial vehicles. Section III discusses our assumptions and develops the system model, and Section IV formulates the problem tackled in this work. Then, Section V discusses the details of the proposed inspection planning control approach and Section VI evaluates the proposed approach. Finally, Section VII concludes the paper and discusses future work.

II. Related Work

Several approaches and algorithms can be found in the literature for the problem of autonomous inspection/coverage planning with single and multiple robots. In this section we give a brief overview of the most relevant techniques. A detailed survey regarding the various inspection/coverage planning techniques in the literature can be found in [14], [15]. Most notably, the problem of inspection/coverage planning with ground robots was investigated in [16] and [17]. Specifically, the authors in [16], propose a boustrophedon cellular decomposition coverage algorithm, in which the free-space of a 2D planar environment is a) first decomposed into non-intersecting regions or cells and b) the cells are then visited by the robot sequentially and covered with simple back-and-forth motions. The work in [17] proposes a two-stage approach for inspecting polygonal objects, with the main objective being the computation of a path such that each point on the object's boundary is observed by the robot. The algorithm in [17] first finds a set of sensing locations which allow full inspection of the polygonal object. In the second stage, the sensing locations found during the previous stage, are connected with the shortest path to generate the robot's inspection path. Several other works [18], [19] have also proposed the decomposition of the free space into several non-overlapping regions, which can be individually covered and inspected with sweeping motions. Extensions [20], [21] of these approaches have investigated the optimal region traversal order and the optimal sweeping direction.

The problem of 2D coverage planning was also investigated in the context of camera networks [22], [23], with the main objective being the optimal control and placement of cameras for full visual coverage of the monitoring space. The majority of the related work discussed so far, transforms the inspection/coverage planning problem to a path planning problem by firstly decomposing the area/object of interest into a number of non-overlapping cells which are then connected together with a path-finding algorithm to form the robot's path. Moreover, these approaches have mainly been tested in 2D environments and they do not consider the robot's dynamic and sensing behavior, i.e., they do not find the robot's control inputs which generate the inspection trajectory.

More related to the proposed approach is the work in [24] which proposes a coverage control approach for guiding a mobile robot to completely cover a bounded two-dimensional region. In [24] the 2D bounded surveillance environment is first covered with a minimum number of disks which exhibit a radius equal to the robots sensing range and then a neural network is used to plan the robot's coverage path. At a second stage, the generated path is adapted to the robot's kinematic constraints. Moreover, in [25], an energy optimized graph-based planner is proposed for UAV-based coverage in 2D discrete environments. Similarly, in [26] a UAV-based terrain coverage approach is proposed for computing a trajectory through a known environment with obstacles that ensures coverage of the terrain while minimizing path repetition. In [27] the terrain coverage problem with a UAV is investigated more realistically with the inclusion of photogrammetric constraints. The approach in [28] tackle the terrain-coverage problem for rectilinear polygonal environments with multiple UAV agents. In this case, the environment is partitioned into multiple sub-regions, which are assigned to the UAVs according to their coverage capabilities.

An off-line sampling-based 3D coverage planning approach for an underwater inspection robot is proposed in [29]. Specifically, the authors propose a redundant roadmap algorithm and a watchman route algorithm [30], to allow the construction of a discrete set of stationary robot views which allow full coverage of the object of interest. The generated view configurations are then connected by solving an instance of the traveling salesman problem (TSP), although the generated path might be infeasible for robots with dynamical constraints.

Finally, inspection planning approaches based on the next best view (NBV)/view-planning techniques [31]–[33] are concerned with the computation of sequence of viewpoints which results in complete scene coverage. These methods are usually applied in unknown environments and often provide only suboptimal results since they must solve instances of the set cover problem [34] and the traveling salesman problem (TSP) [35]. Moreover, these approaches focus on the selection of discrete sensor views rather than the construction of a continuous trajectory, which also accounts for the robots dynamical and/or sensing constraints.

III. Preliminaries

A. Agent Dynamical Model

In this work an autonomous UAV agent maneuvers inside a bounded surveillance region $\mathcal{W} \subset \mathbb{R}^3$, with dynamics governed according to the following discrete-time dynamical model:

$$x_{t+1} = Ax_t + Bu_t \tag{1}$$

where $x_t = [p_t, \nu_t] \in \mathbb{R}^{6,1}$ is the state of the agent at time-step t, which consists of position $p_t \in \mathbb{R}^{3,1}$ and velocity $\nu_t \in \mathbb{R}^{3,1}$ components in 3D cartesian coordinates. The vector $u_t \in \mathbb{R}^{3,1}$ denotes the input control force applied in each dimension, which allows the agent to change its direction and speed. The matrices A and B are given by:

$$A = \begin{bmatrix} \mathrm{I}_{3\times3} & \delta t\ \mathrm{I}_{3\times3} \\ 0_{3\times3} & \alpha\ \mathrm{I}_{3\times3} \end{bmatrix},\ B = \begin{bmatrix} 0_{3\times3} \\ \beta\ \mathrm{I}_{3\times3} \end{bmatrix} \tag{2}$$

where δt is the sampling interval, $\mathrm{I}_{3\times3}$ and $0_{3\times3}$ are the identity matrix and zero matrix respectively, both of dimension 3×3, and the parameters α and β are given by $\alpha = (1 - \eta)$ and $\beta = \frac{\delta t}{m}$. Moreover, the parameter η is used to model the air resistance and m denotes the agent's mass.

B. Agent Camera Model

The UAV agent is equipped with an onboard forward facing camera which is used for inspecting cuboid-like structures. Without loss of generality, we assume that the camera's horizontal and vertical field-of-view (FOV) angles are equal, and thus the projection of the camera's FOV ($\mathcal{F}$) on a planar surface has a square footprint with side length ℓ. This is depicted in Fig. 1(a), where the camera view is represented by a regular square pyramid whose apex (i.e., the center of the camera located at the lens) is directly above the centroid of its square base, and whose sides are 4 triangular faces meeting at the apex. We assume that the size of the projected FOV footprint is a linear function of the distance between the agent and the object of interest and thus:

$$\ell = g(d) = z_1 d + z_0 \tag{3}$$

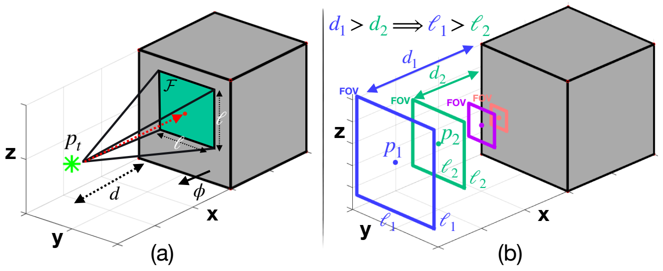

Fig. 1. The figure illustrates the UAV's sensing model. (a) The UAV is equipped with a camera that has a view which is represented by a regular square pyramid whose apex (i.e., p_t) is directly above the centroid of its square base, and whose sides are 4 triangular faces meeting at the apex. $\mathcal{F}$ denotes the camera's projection, on the surface of the cuboid-like structure, which is represented by a square with length ℓ. (b) The size of the projected FOV on a planar surface is a function of the distance d between the agent position p_t and the cuboid-like structure.

where d denotes the distance between the location of the agent and the structure that needs to be inspected, and the pair (z_0, z_1) are the model's parameters. This is shown in Fig. 1(b), where the size of projected square camera footprint decreases as the agent approaches the structure (i.e., the distance between the agent and the structure decreases and so does the size of camera FOV). Moreover, in this work it is assumed that the camera principal axis (depicted with the red line from the camera center perpendicular to the image plane in Fig. 1(a)) is always parallel with the outward normal vector (ϕ) of the plane which contains the face that is being viewed. In other words it is assumed that the agent automatically adjusts the onboard camera so that the viewing direction is parallel with the normal vector ϕ at all times.

C. Structure to be inspected

The 3D structure (or object) to be inspected by the UAV agent is represented in this work by a rectangular cuboid $\mathcal{C}$ of arbitrary size. A rectangular cuboid is convex polyhedron which exhibits six faces $f_i, i = [1,..,6]$, where opposite faces are equal and parallel, and each pair of adjacent faces meets in a right angle. The equation of the plane which contains the i_{th} face of the cuboid is given by:

$$\phi_i^\top \cdot x = \gamma_i \tag{4}$$

where $\phi_i^\top \cdot x$ denotes the dot product between $\phi_i^\top$ and x, $\phi_i \in \mathbb{R}^{3,1}$ is the outward normal vector to the plane which contains the i_{th} face, $\gamma_i \in \mathbb{R}$ is a constant derived by the dot product of ϕ_i with a known point on the plane, and $x \in \mathbb{R}^{3,1}$ is an arbitrary point in 3D space. Thus, all points x which satisfy the equality in Eqn. (4) belong to the plane which contains the i_{th} face of the cuboid. Consequently, a point $x \in \mathbb{R}^{3,1}$ resides inside the cuboid $\mathcal{C}$ when it satisfies all six inequalities:

$$x \in \mathcal{C} \iff \phi_i^\top \cdot x \le \gamma_i, \ \forall i = [1,..,6] \tag{5}$$

Equivalently, Eqn. (5) can be written more compactly in matrix form as $\Phi x \le \Gamma$, where Φ is a matrix of dimensions 6×3, where the i_{th} row corresponds to ϕ_i and Γ is a column vector of dimensions 6×1, where the i_{th} element corresponds to γ_i.

The goal of the UAV agent is to inspect all faces of cuboid $\mathcal{C}$. More specifically, we consider the existence of a finite

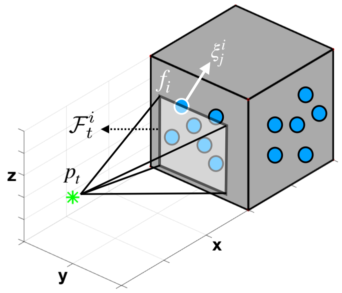

Fig. 2. The UAV's objective is to inspect all N_i feature-points $\xi_j^i, j = [1,..,N_i]$ scattered on face $f_i, i \in L$, for all $|L|$ faces which contain feature-points. The feature-point ξ_j^i is inspected at time t if it resides within the camera's projected FOV i.e., $\xi_j^i \in \mathcal{F}_t^i$ as illustrated in this figure.

number of feature-points scattered on the cuboid's surface that must be viewed through the agent's camera in order for $\mathcal{C}$ to be visually inspected by the UAV agent. Let $\xi_j^i \in \mathbb{R}^{3,1}$ to represent the j_{th} feature-point on the i_{th} face of the cuboid $\mathcal{C}$. We say that cuboid $\mathcal{C}$ has been visually inspected by the UAV agent *iff*:

$$\xi_j^i \in \bigcup_{t \le \mathcal{T}_{\text{max}} \ \wedge \ i \in L} \mathcal{F}_t^i, \ \forall j \tag{6}$$

where $\mathcal{T}_{\text{max}}$ denotes the total mission inspection time, $\mathcal{F}_t^i$ denotes the projected camera FOV on the i_{th} face of the cuboid at time-step t, and finally L denotes the set of cuboid faces which contain feature-points that need to be inspected. In this work we assume that the three-dimensional map of the environment is readily available for the UAV agent to use during its mission, which was obtained through a 3D mapping procedure [36] executed prior to the inspection mission. During this 3D mapping procedure, the structure to be inspected has been reconstructed as a rectangular cuboid $\mathcal{C}$, and a fixed number of feature-points $\xi_j^i \in \mathbb{R}^{3,1}$ have been identified and extracted from its 3D point cloud. The UAV agent uses this 3D map to acquire the number of feature-points along with their location in order to plan its inspection mission.

IV. Problem Statement

Let an autonomous UAV agent with initial state x_t at time-step t, to evolve inside a bounded surveillance area $\mathcal{W} \subset \mathbb{R}^3$. Let a cuboid-like structure $\mathcal{C} \in \mathcal{W}$ to contain a number of feature-points $\xi_j^i, \ i \in L, \ j = [1,..,N_i]$, where L denotes set of cuboid faces which contain feature-points and N_i denotes the total number of feature-points on the i_{th} face of the cuboid. The inspection planning problem tackled in this work can be stated as follows: *Given a sufficiently large mission inspection time $\mathcal{T}_{max}$, find the optimal UAV control inputs $U_t = \{u_{t|t},..,u_{t+T-1|t}\}, t \le \mathcal{T}_{max}$, over a rolling finite planning horizon of length T time-steps, such that all feature-points on the cuboid's surface are visually inspected by the UAV agent at some point during the mission time i.e., $\xi_j^i \in \bigcup_{t \le \mathcal{T}_{max} \ \wedge \ i \in L} \mathcal{F}_t^i, \forall j$.* In a high level form, the problem tackled in this work can be formulated as shown in Problem (P1).

In this work the notation $x_{t'|t}$ denotes the agent's future predicted state at time-step t' which is computed at time-step t. In essence we are looking to find the optimal future

Problem (P1): `High-level Controller`

$$\underset{\mathbf{U}_t}{\arg\min}\ \mathcal{J}_{\text{Inspection}},\ t \le \mathcal{T}_{\max} \tag{7a}$$

subject to: $\tau \in \{0, \ldots, T-1\}$

$$x_{t+\tau+1|t} = Ax_{t+\tau|t} + Bu_{t+\tau|t} \qquad \forall \tau \tag{7b}$$

$$x_{t|t} = x_{t|t-1} \tag{7c}$$

$$\xi^i_j \in \bigcup_{t \le \mathcal{T}_{\max}\ \wedge\ i \in L} \mathcal{F}^i_t, \qquad \forall j = [1, .., N_i] \tag{7d}$$

$$p_{t+\tau+1|t} \notin \mathcal{C} \qquad \forall \tau \tag{7e}$$

$$x_{t+\tau+1|t} \in \mathcal{X},\ u_{t+\tau|t} \in \mathcal{U} \qquad \forall \tau \tag{7f}$$

UAV control inputs $U_t = \{u_{t|t}, .., u_{t+T-1|t}\}$, which optimize the inspection objective function $\mathcal{J}_{\text{Inspection}}$ and which satisfy a certain set of constraints i.e., Eqn. (7b) - Eqn. (7e). The constraints in Eqn. (7b) and Eqn. (7c) are due to the agent's dynamical model as discussed in Sec. III-A, and the constraint in Eqn. (7d) makes sure that all feature-points ξ^i_j on the cuboid $\mathcal{C}$ will be viewed by the agent's camera at least once during the mission i.e., all feature-points must be included inside the agent's projected camera FOV. Then the collision avoidance constraint in Eqn. (7e) makes sure that the UAV agent avoids collisions with the cuboid under inspection $\mathcal{C}$ at all times, and finally, the constraint in Eqn. (7f) place the agent's state and control inputs within the desired operating bounds $\mathcal{X}$ and $\mathcal{U}$ respectively.

V. UAV-based Receding Horizon Inspection Planning Control

Essentially, the inspection planning problem discussed in the previous section is posed in this work as a receding horizon constrained optimal control problem, in which the future optimal UAV control inputs $U_t = \{u_{t|t}, .., u_{t+T-1|t}\}, t \le \mathcal{T}_{\max}$ are computed at each time-step t, over a finite moving planning horizon of length T. The first control input of the sequence is then applied to the UAV and the problem is solved again for the next time-step. In the proposed approach the trajectory planning decisions are optimized based on the mission objective in an on-line fashion, and according to a set of mission constraints including a) the UAV's dynamical and sensing model, b) collision avoidance constraints and c) duplication of effort constraints.

Next we discuss the details of the proposed receding horizon inspection planning controller for a single cuboid-like structure that needs to be inspected. Specifically, we have transformed the optimal control inspection planning problem shown in (P1), into a mixed integer quadratic program (MIQP), as shown in detail in problem (P2), which can be solved using readily available optimization solvers [37]. We will begin the analysis of the proposed approach by first discussing the mission constraints i.e., Eqn. (8b) - Eqn. (8t). Then, we discuss in detail how we have designed the multi-objective cost function i.e., Eqn. (8a) that drives the inspection planning mission.

A. Inspection Constraints

The first two constraints in Eqn. (8b) and Eqn. (8c) are due to the agent's dynamical model as already discussed in Sec III-A. The next constraint shown in Eqn. (8d) computes

Problem (P2): `Inspection Controller`

$$\underset{\mathbf{U}_t}{\arg\min}\ \mathcal{J}_{\text{Inspection}},\ t \le \mathcal{T}_{\max} \tag{8a}$$

subject to $\tau \in \{0, \ldots, T-1\}$**:**

$$x_{t+\tau+1|t} = Ax_{t+\tau|t} + Bu_{t+\tau|t} \qquad \forall \tau \tag{8b}$$

$$x_{t|t} = x_{t|t-1} \tag{8c}$$

$$d^{\text{face}}_{\tau,i} = |H_i p_{t+\tau+1|t} - C_i| \qquad \forall \tau, i \tag{8d}$$

$$\ell_{\tau,i} = g(d^{\text{face}}_{\tau,i}) \qquad \forall \tau, i \tag{8e}$$

$$J_{i,c} p_{t+\tau+1|t} + (M - K_{i,c}) b^1_{\tau,i,c} \le M \qquad \forall \tau, i, c = [1, .., 5] \tag{8f}$$

$$5 b^2_{\tau,i} - \sum_{c=1}^{5} b^1_{\tau,i,c} \le 0 \qquad \forall \tau, i \tag{8g}$$

$$\sum_{i=1}^{L} b^2_{\tau,i} \le 1 \qquad \forall \tau \tag{8h}$$

$$\Omega_{i,c} \xi^i_j b^3_{\tau,i,j,c} - \Omega_{i,c} p_{t+\tau+1|t} b^3_{\tau,i,j,c} - \frac{\ell_{\tau,i}}{2} \le 0 \qquad \forall \tau, i, j, c = [1, .., 4] \tag{8i}$$

$$4 b^4_{\tau,i,j} - \sum_{c=1}^{4} b^3_{\tau,i,j,c} \le 0 \qquad \forall \tau, i, j \tag{8j}$$

$$\kappa^1_{\tau,i,j} = b^2_{\tau,i} \wedge b^4_{\tau,i,j} \qquad \forall \tau, i, j \tag{8k}$$

$$\kappa^2_{\tau,i,j} \le \kappa^1_{\tau,i,j} + \mathcal{Q}_{i,j} \qquad \forall \tau, i, j \tag{8l}$$

$$\sum_{\tau=0}^{T-1} \kappa^2_{\tau,i,j} \le 1 \qquad \forall i, j \tag{8m}$$

$$\kappa^2_{\tau,i,j} * d^{\text{face}}_{\tau,i} \le D_{\max} \qquad \forall \tau, i, j \tag{8n}$$

$$\Phi_l p_{t+\tau+1|t} \ge \Gamma_l - M o_{\tau,l} \qquad \forall \tau, l \tag{8o}$$

$$\sum_{l=1}^{6} o_{\tau,l} \le 5 \qquad \forall \tau \tag{8p}$$

$$x_{t+\tau+1|t} \in \mathcal{X},\ u_{t+\tau|t} \in \mathcal{U} \qquad \forall \tau \tag{8q}$$

$$b^1_{\tau,i,c}, b^2_{\tau,i}, b^3_{\tau,i,j,c}, b^4_{\tau,i,j} \in \{0,1\} \qquad \forall \tau, i, j, c \tag{8r}$$

$$\kappa^1_{\tau,i,j}, \kappa^2_{\tau,i,j}, \mathcal{Q}_{i,j}, o_{\tau,l} \in \{0,1\} \qquad \forall \tau, i, j, l \tag{8s}$$

$$i = [1, .., |L|],\ j = [1, .., N_i],\ l = [1, .., 6] \tag{8t}$$

the distance between the agent's position $p_{t+\tau+1|t}$ (also abbreviated as p_τ) and every face $f_i, i \in L$ of the cuboid-like structure $\mathcal{C}$ for all future time-steps $\tau \in \{0, \ldots, T-1\}$ inside the planning horizon, where for brevity we have used the notation $d^{\text{face}}_{\tau,i}$ to mean $d^{\text{face}}_{t+\tau+1|t,i}$. In Eqn. (8d), H is a matrix with dimensions $|L|$-by-3, where $|L|$ denotes the number of cuboid faces which need to be inspected, and C is a $|L|$-by-1 column vector. The distance $d^{\text{face}}_{\tau,i}$ between the agent with position p_τ and the i_{th} face is defined here as the 1-norm distance between p_τ and its orthogonal projection on the plane which contains f_i, i.e., the perpendicular distance to the nearest point on the plane. For instance, let the plane (with equation $x = a$) which is parallel to the zy-axis to contain the i_{th} face of the cuboid to be inspected, and the agent to be located in front of f_i as shown in Fig. 1, with $p_\tau = [p_\tau(x), p_\tau(y), p_\tau(z)]^\top$. In this example the i_{th} row of the matrix H is given by $H_i = [1, 0, 0]$ and $C_i = a$. Thus

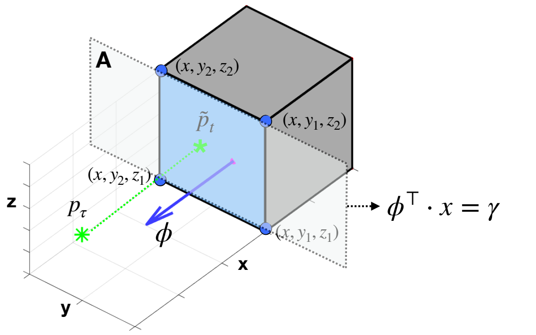

Fig. 3. The UAV agent with position p_τ is viewing the blue face of the cuboid-like structure when a) the projection $\tilde{p}_\tau$ of its position on the that face resides inside the face's rectangular region (i.e., blue shaded area) and b) p_τ resides inside the positive half-space created by the plane A ($\phi^\top \cdot x = \gamma$) which contains the blue face as shown above, where ϕ denotes the outward normal vector on A.

the distance between the agent and the face f_i is computed as $d^{\text{face}}_{\tau,i} = |H_i p_{t+\tau+1|t} - C_i| = |p_\tau(x) - a|$. In a similar fashion, H and C are populated for all faces $|L|$ that need to be inspected and the distance between the agent and all the cuboid faces is computed for all time-steps inside the planning horizon using Eqn. (8d).

The next constraint shown in Eqn. (8e) computes the size of the projected camera FOV on every face i of the cuboid to be inspected for all time-steps τ inside the planning horizon. Specifically, the side length $\ell_{\tau,i}$ of the projected square camera FOV is computed as a function of the distance $d^{\text{face}}_{\tau,i}$ between the agent and each face f_i i.e.:,

$$\ell_{\tau,i} = g(d^{\text{face}}_{\tau,i}), \quad \forall \tau, i \tag{9}$$

where $g(.)$ is a linear or piecewise linear function with respect to the input (i.e., Eqn. (3)), and again $\ell_{t+\tau+1|t,i}$ has been abbreviated as $\ell_{\tau,i}$ for notational clarity.

The constraints in Eqn. (8f) - Eqn. (8h) use the binary variables $b^1_{\tau,i,c}$ and $b^2_{\tau,i}$ to determine whether the agent's camera is viewing any of the cuboid's faces and if it does, identify on which face the camera's FOV is being projected. This is illustrated in Fig. 3, where the agent with position p_τ is viewing the i_{th} face of the cuboid (shown in blue color), with vertices $V = [v_1 \ v_2 \ v_3 \ v_4]$ where $v_1 = [x, y_1, z_1]^\top$, $v_2 = [x, y_1, z_2]^\top$, $v_3 = [x, y_2, z_2]^\top$ and $v_4 = [x, y_2, z_1]^\top$. As shown in the figure, the face f_i is contained within the plane A with equation $\phi^\top \cdot x = \gamma$, where ϕ is the outward normal to the plane, x is an arbitrary point on the plane and γ is a constant. Let us denote the projection of p_τ on the plane A as $\tilde{p}_\tau = [p_\tau(y), p_\tau(z)]^\top \in \mathbb{R}^{2,1}$. Now we can determine whether the agent is viewing face f_i with the following two conditions which must be satisfied simultaneously: (a) the agent's projected position $\tilde{p}_\tau$ resides inside face f_i i.e., $y_1 \le p_\tau(y) \le y_2$ and $z_1 \le p_\tau(z) \le z_2$ and (b) the agent must be located in front of face f_i or equivalently the agent must reside inside the positive half-space formed by the plane A which contains face f_i i.e., $\phi^\top \cdot x \ge \gamma$ as illustrated in Fig. 3. Condition (a) can be written more compactly in matrix form as $\tilde{J}_i \cdot p_\tau \le \tilde{K}_i$ where:

$$\tilde{J}_i = \begin{bmatrix} 0 & -1 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & -1 \\ 0 & 0 & 1 \end{bmatrix}, \quad \text{and} \quad \tilde{K}_i = \begin{bmatrix} -y_1 \\ y_2 \\ -z_1 \\ z_2 \end{bmatrix} \tag{10}$$

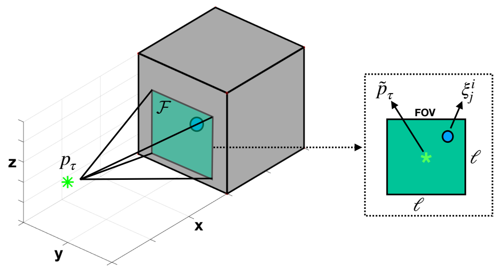

Fig. 4. The 3D feature-point $\xi^i_j = [\xi^i_j(x), \xi^i_j(y), \xi^i_j(z)]^\top$ is planned to be inspected by the UAV agent p_τ at time-step τ, when it resides inside the agent's camera projected FOV at time τ i.e., $\xi^i_j \in \mathcal{F}^i_\tau$. In the example illustrated above this is equivalent to the following constraints: $p_\tau(y) - \frac{\ell}{2} \le \xi^i_j(y) \le p_\tau(y) + \frac{\ell}{2}$ and $p_\tau(z) - \frac{\ell}{2} \le \xi^i_j(z) \le p_\tau(z) + \frac{\ell}{2}$.

Subsequently, the matrix J_i and the column vector K_i in Eqn. (8f) are defined for face f_i as $J_i = [\tilde{J}_i \ \phi^\top]$ and $K_i = [\tilde{K}_i \ \gamma]$. Thus J_i and K_i have dimensions 5×3 and 5×1 respectively. Thus, the agent is viewing face f_i if all 5 constraints discussed above are being satisfied. The constraints in Eqn. (8f) - Eqn. (8h), also shown below, can now be explained as follows:

$$J_{i,c} p_{t+\tau+1|t} + (M - K_{i,c}) b^1_{\tau,i,c} \le M, \quad \forall \tau, i, c = [1, ..5]$$

$$5 b^2_{\tau,i} - \sum_{c=1}^{5} b^1_{\tau,i,c} \le 0, \quad \forall \tau, i$$

$$\sum_{i=1}^{L} b^2_{\tau,i} \le 1, \quad \forall \tau$$

First, the binary variable $b^1_{\tau,i,c}$ is used to indicate whether at time τ inside the planning horizon (or equivalently at time $t+\tau+1|t$), the constraint c (out of 5) is true for face f_i. If the c_{th} constraint is true then $b^1_{\tau,i,c}$ is activated i.e., $b^1_{\tau,i,c} = 1$. Otherwise, $b^1_{\tau,i,c} = 0$ and the inequality in Eqn. (8f) becomes $J_{i,c} p_{t+\tau+1|t} \le M$ which is always true for a large constant M. Then, the constraint in Eqn. (8g) uses the binary variable $b^2_{\tau,i}$ to determine whether $b^1_{\tau,i,c} = 1, \forall c = [1, ..5]$, in which case $b^2_{\tau,i}$ is activated i.e., $b^2_{\tau,i} = 1$. Thus the binary variable $b^2_{\tau,i}$ is used to determine whether face f_i is being viewed by the agent at some point in time τ. Observe that, Eqn. (8g) is also satisfied when $b^2_{\tau,i} = 0$. Finally, the constraint in Eqn. (8h) makes sure that at any point in time τ at most one face is being viewed by the agent, which is added here for numerical stability purposes.

Moving forward with the analysis of the proposed receding horizon 3D inspection controller, the constraints in Eqn. (8i) - Eqn. (8j) (also shown below), use the binary variables $b^3_{\tau,i,j,c}$ and $b^4_{\tau,i,j}$ to determine whether the j_{th} feature-point, on the i_{th} face of the cuboid resides inside the agent's camera FOV projection $\tilde{\mathcal{F}}^i_\tau$ at time τ.

$$\Omega_{i,c} \xi^i_j b^3_{\tau,i,j,c} - \Omega_{i,c} p_{t+\tau+1|t} b^3_{\tau,i,j,c} - \frac{\ell_{\tau,i}}{2} \le 0, \quad \forall \tau, i, j, c$$

$$4 b^4_{\tau,i,j} - \sum_{c=1}^{4} b^3_{\tau,i,j,c} \le 0, \quad \forall \tau, i, j$$

The constraint in Eqn. (8i) is illustrated more clearly with an example in Fig. 4. First observe that the agent's position

p_τ is determined by the constraints in Eqn. (8b) and Eqn. (8c). The camera FOV projection $\tilde{\mathcal{F}}^i_\tau$ (colored green) on the i_{th} face at time τ is a square, centered at $\tilde{p}_\tau = [p_\tau(y) p_\tau(z)]$ (i.e., the projection of the agent's position on the face f_i), with side length $\ell_{\tau,i}$ given by Eqn. (8e), governed by the distance between the agent and face f_i as determined by Eqn. (8d). The notation $\tilde{\mathcal{F}}^i_\tau, \forall i$ is used here to indicate the hypothetical FOV projection on the i_{th} face of the cuboid at time τ, assuming that the i_{th} face is actually being observed. Thus, the procedure described in this paragraph computes the hypothetical FOV projection on all $|L|$ cuboid faces to determine which feature-points are included in each hypothetical FOV projection. As we discuss in the next paragraph at every time instance τ only one FOV projection is active which is instead denoted by $\mathcal{F}^i_\tau$, and determined as explained next.

To continue our discussion observe from Fig. 4, that an arbitrary feature-point $\xi^i_j = [\xi^i_j(x), \xi^i_j(y), \xi^i_j(z)]^\top \in \mathbb{R}^{3,1}$ resides inside the hypothetical projected FOV $\tilde{\mathcal{F}}^i_\tau$ when:

$$p_\tau(y) - \frac{\ell}{2} \leq \xi^i_j(y) \leq p_\tau(y) + \frac{\ell}{2} \quad (11a)$$

$$p_\tau(z) - \frac{\ell}{2} \leq \xi^i_j(z) \leq p_\tau(z) + \frac{\ell}{2} \quad (11b)$$

The constraints in Eqn. (11) can be written in matrix form as:

$$\begin{bmatrix} 0 & 1 & 0 \\ 0 & -1 & 0 \\ 0 & 0 & 1 \\ 0 & 0 & -1 \end{bmatrix} \begin{bmatrix} \xi^i_j(x) \\ \xi^i_j(y) \\ \xi^i_j(z) \end{bmatrix} \leq \begin{bmatrix} 0 & 1 & 0 \\ 0 & -1 & 0 \\ 0 & 0 & 1 \\ 0 & 0 & -1 \end{bmatrix} \begin{bmatrix} p_\tau(x) \\ p_\tau(y) \\ p_\tau(z) \end{bmatrix} + \frac{\ell_{\tau,i}}{2}$$

or more compactly as $\Omega_i \xi^i_j \leq \Omega_i p_\tau + 0.5\ell_{\tau,i}$. Therefore, Ω_i is a matrix with size 4-by-3 which encodes the 4 constraints that need to be satisfied in order for feature-point ξ^i_j on the i_{th} face of the cuboid to be included inside the agent's hypothetical FOV $\tilde{\mathcal{F}}^i_\tau$ at time τ when the agent is located at p_τ. This is achieved with the binary variable $b^3_{\tau,i,j,c}$ which is activated when constraint c (out of 4) is true. Otherwise $b^3_{\tau,i,j,c} = 0$. Subsequently, the binary variable $b^4_{\tau,i,j}$ in Eqn. (8j) is activated when all 4 constraints are true i.e., $b^3_{\tau,i,j,c} = 1, \forall c = [1,..4]$. Thus, $b^4_{\tau,i,j}$ allows us to determine if at time τ the feature-point j which is on the i_{th} face of the cuboid, resides inside the hypothetical FOV projection $\tilde{\mathcal{F}}^i_\tau$.

Observe however, that with the binary variable $b^4_{\tau,i,j}$ alone we cannot determine whether feature-point ξ^i_j is viewed by the agent. This is because $b^4_{\tau,i,j}$ does not encode which of the cuboid faces (if any) is the agent actually viewing at time τ. Therefore, in order to actually determine if the feature-point ξ^i_j is being observed by the UAV agent at time τ i.e., $\xi^i_j \in \mathcal{F}^i_\tau$, we use the constraint in Eqn. (8k) as shown below:

$$\kappa^1_{\tau,i,j} = b^2_{\tau,i} \wedge b^4_{\tau,i,j}, \quad \forall \tau, i, j$$

where we have combined the binary variables $b^2_{\tau,i}$ and $b^4_{\tau,i,j}$ with a logical conjunction. The resulting binary variable $\kappa^1_{\tau,i,j}$ is activated only when both $b^2_{\tau,i}$ and $b^4_{\tau,i,j}$ are true. Thus, the UAV agent views feature-point ξ^i_j at time τ, iff $\xi^i_j \in \tilde{\mathcal{F}}^i_\tau$ indicated by $b^4_{\tau,i,j}$ and the UAV agent views at the same time the i_{th} face of the cuboid which is given by $b^2_{\tau,i}$.

As we have already mentioned, the UAV agent plans its inspection trajectory at every time-step t over a moving finite planning horizon. Since in the majority of scenarios the inspection mission cannot be completed inside a single planning horizon, the agent needs to be equipped with some form of memory or record in order to keep track the mission progress (i.e., which feature-points are left to be inspected), and minimize the duplication of work (i.e., avoid inspecting feature-points that have already been inspected). To enable this functionality we use the constraint in Eqn. (8l) i.e.,:

$$\kappa^2_{\tau,i,j} \leq \kappa^1_{\tau,i,j} + \mathcal{Q}_{i,j}, \quad \forall \tau, i, j$$

where the agent's memory is being realized with the 2D matrix $\mathcal{Q}_{i,j} \in \{0,1\}$, $i = [1,..|L|]$, $j = [1,..,N_i]$. When the UAV agent inspects a particular feature-point ξ^i_j the (i,j)-element of $\mathcal{Q}_{i,j}$ is activated. Specifically,

$$\mathcal{Q}_{i,j} = \begin{cases} 1, & \text{iff } \exists t \leq \mathcal{T}_{\text{max}} : \xi^i_j \in \mathcal{F}^i_t \\ 0, & \text{o.w} \end{cases} \quad (12)$$

where as we have previously mentioned $\mathcal{F}^i_t$ is the agent's projected FOV on the i_{th} face of the cuboid at the current time-step t. Because $\kappa^2_{\tau,i,j}$ is also a binary variable, observe that with the constraint in Eqn. (8l) the UAV agent has no incentive in inspecting (during the current and future planning horizons), a feature-point ξ^i_j which has already been inspected i.e., $\mathcal{Q}_{i,j} = 1$ (assuming Eqn. (8l) is maximized). This allows the agent to minimize the duplication of work and plan inspection trajectories towards new feature-points. Next, the constraint in Eqn. (8m) discourages the agent from planning trajectories which inspect more than once the same feature-point inside the current planning horizon.

The quality of the visual inspection task of the cuboid-like structure inversely decreases with the distance between the agent and the feature-points to be inspected. In particular, we assume that for distances beyond a certain cut-off threshold, the amount of detailed captured by the agent's camera in not sufficient for reliably performing its inspection task. For this reason, we use the constraint in Eqn. (8n) i.e.,

$$\kappa^2_{\tau,i,j} * d^{\text{face}}_{\tau,i} \leq D_{\text{max}}, \quad \forall \tau, i, j \quad (13)$$

where D_{max} denotes the cut-off distance beyond which the inspection task must not be performed. Consequently, the agent is forced to inspect feature-points ξ^i_j at time τ (indicated by $\kappa^2_{\tau,i,j}$) i.e., $\xi^i_j \in \mathcal{F}^i_\tau$, from distances less than or equal to the cut-off distance.

Finally, the constraints in Eqn. (8o) - Eqn. (8p) implement collision avoidance constraints with the cuboid-like structure to be inspected. The objective here is to make sure that the agent's position p_τ during the planning horizon does not resides inside the cuboid-like structure i.e., $p_\tau \notin \mathcal{C}$ which is accomplished as shown below:

$$\Phi_l p_{t+\tau+1|t} \geq \Gamma_l - M o_{\tau,l}, \quad \forall \tau, l$$

$$\sum_{l=1}^{6} o_{\tau,l} \leq 5, \quad \forall \tau$$

As we have already discussed in Sec. III-C, the agent is inside the cuboid-like structure at some time τ when $\Phi p_\tau \leq \Gamma$. Therefore, a collision can be avoided at time τ when $\exists\ l \in [1,..,6] : \phi_l^\top p_\tau \geq \gamma_l$, which is is implemented with the inequalities shown above. Specifically, we use the binary variable $o_{\tau,l}$ to determine whether the l_{th} constraint (i.e., $\phi_l^\top p_\tau \geq \gamma_l$) is false, in which case $o_{\tau,l}$ is activated, i.e., $o_{\tau,l} = 1$. Subsequently, the inequality in Eqn. (8p) makes sure that the number of times $o_{\tau,l}$ is activated at a particular time τ is less than or equal to 5, which indicates that the agent is outside the cuboid-like structure at time τ. The rest of the constraints in Eqn. (8q) - Eqn. (8t) restrict the agent's state and control inputs within the desired bounds and declare the various variables used.

B. Inspection Objective

To summarize, at each time-step $t \leq \mathcal{T}_{\text{max}}$ the autonomous UAV agent computes its future inspection trajectory $x_{t+\tau+1|t}, \forall \tau$ over a finite moving planning horizon i.e., $\tau \in \{0,..,T-1\}$ of length T, by taking into account the mission constraints in Eqn. (8b) - Eqn. (8t) and by optimizing the mission objective in Eqn. (8a). Specifically, the agent's control inputs $U_t = \{u_{t|t},..,u_{t+T-1|t}\}$ are chosen such that the multi-objective cost function $\mathcal{J}_{\text{Inspection}}$ is minimized, which is defined in this work as follows:

$$\mathcal{J}_{\text{Inspection}} = \left(-\sum_{\tau=0}^{T-1} \sum_{i=1}^{|L|} \sum_{j=1}^{N_i} \frac{\kappa^2_{\tau,i,j}(T-\tau)}{T} \right) + w \times d^2_{\text{target}} \tag{14}$$

The first term shown in Eqn. (14) is used here to maximize the number of unobserved feature-points over the planning horizon that reside inside the agent's projected FOV. As a reminder, the binary variable $\kappa^2_{\tau,i,j}$ indicates whether at time-step $t+\tau+1|t$, the feature-point ξ^i_j is included inside the agent's projected FOV $\mathcal{F}^i_{t+\tau+1|t}$. Therefore, by minimizing the term inside the parenthesis, the UAV agent generates inspection plans which allow at each time-step τ of the planning horizon the maximum number of feature-points to be included inside the camera's projected FOV. Also note here that the weight factor $(T-\tau)T^{-1}$ which multiplies $\kappa^2_{\tau,i,j}$ drives the UAV agent to want to inspect feature-points at the earliest possible time. Moreover, due to the constraint in Eqn. (8l), this sub-objective makes sure that the feature-points that are planned to be inspected have not been inspected in the past. Moreover, the constraint in Eqn. (8n) makes sure that the inspection planning will take place within the camera's working distance i.e., all feature-points will be inspected at a distance less than or equal to the cut-off distance D_{max}.

When the length of the planning horizon is not sufficiently large, solving the inspection planning problem by solely optimizing the term in the parenthesis becomes infeasible. This is because without a sufficiently large planning horizon the generated inspection trajectory would not be able to reach all feature-points for inspection, and the mission will fail. In order to make the problem independent of the length of the planning horizon and to increase the robustness of the proposed approach we include in the objective function an additional term (i.e., cost-to-go factor) as shown in Eqn. (14), (where $d_{\text{target}} = \|p_{t+1|t} - \hat{\xi}^i_j\|_2$) which enables the agent to greedily move (inside each planning horizon) towards the nearest unobserved feature-point $\hat{\xi}^i_j$ i.e.,:

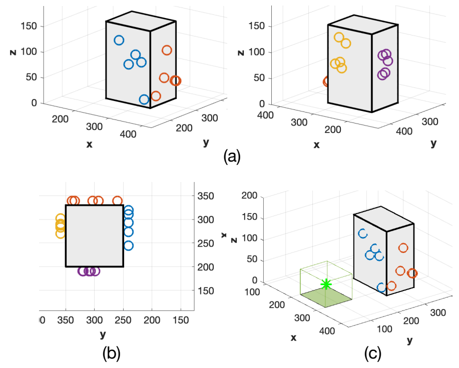

Fig. 5. The figure illustrates an object of interest that needs to be inspected by the UAV agent. (a)(b) The feature-points that must be observed by the UAV agent are scattered on the object's surface, marked with circles. (c) The $\star$ denotes the agent's initial location.

$$\hat{\xi}^i_j = \underset{\xi^i_j \notin \mathcal{Q}_{i,j}}{\arg\min} \|p_{t|t} - \xi^i_j\|_2 \tag{15}$$

where the notation $\xi^i_j \notin \mathcal{Q}_{i,j}$ is used to denote feature-points that have not be observed and $p_{t|t}$ is the current UAV position. Thus Eqn. (15) finds the nearest unobserved feature-point $\hat{\xi}^i_j$ with respect to the agent's current position and the minimization of the square of d_{target} drives the agent's future position $p_{t+1|t}$ towards $\hat{\xi}^i_j$. Therefore, the agent can always move towards the unobserved feature-points even when no feature-points can be observed during the current planning horizon. Finally, w is a tuning weight which determines the emphasis given to the two sub-objectives.

To summarize, we have presented a receding horizon inspection control approach which allows an autonomous UAV agent to inspect in 3D a cuboid-like structure. The proposed approach, based on the agent's dynamical and sensing model, generates inspection plans that enable the UAV agent to observe a finite number of feature-points scattered throughout the surface of a cuboid-like structure by appropriately selecting its control inputs inside a rolling planning horizon. The mission objective in Eqn. (14) aims at maximizing the number of unobserved future-points planned to be inspected inside the planning horizon and thus minimizing the mission time. Finally, the mission objective in Eqn. (14) also makes sure that a feasible solution can be obtained for planning horizons of arbitrary sizes.

VI. Evaluation

A. Simulation Setup

For the evaluation of the proposed approach we have used the following setup: The UAV's dynamical model is according to Eqn. (1) with $\delta t = 1$s. The UAV's mass m and the air resistance coefficient η are set to 3.35kg and respectively 0.2. The UAV's control input u_t is bounded in each dimension within the interval $[-20, 20]$N, the UAV's acceleration ν_t is limited in each dimension within the interval $[-15, 15]$ m/s and the parameters z_0 and z_1 are set to 10 and 0.5 respectively. We assume that the agent is maneuvering inside a bounded surveillance region of dimensions 500m by 500m by 250m, and its objective is to

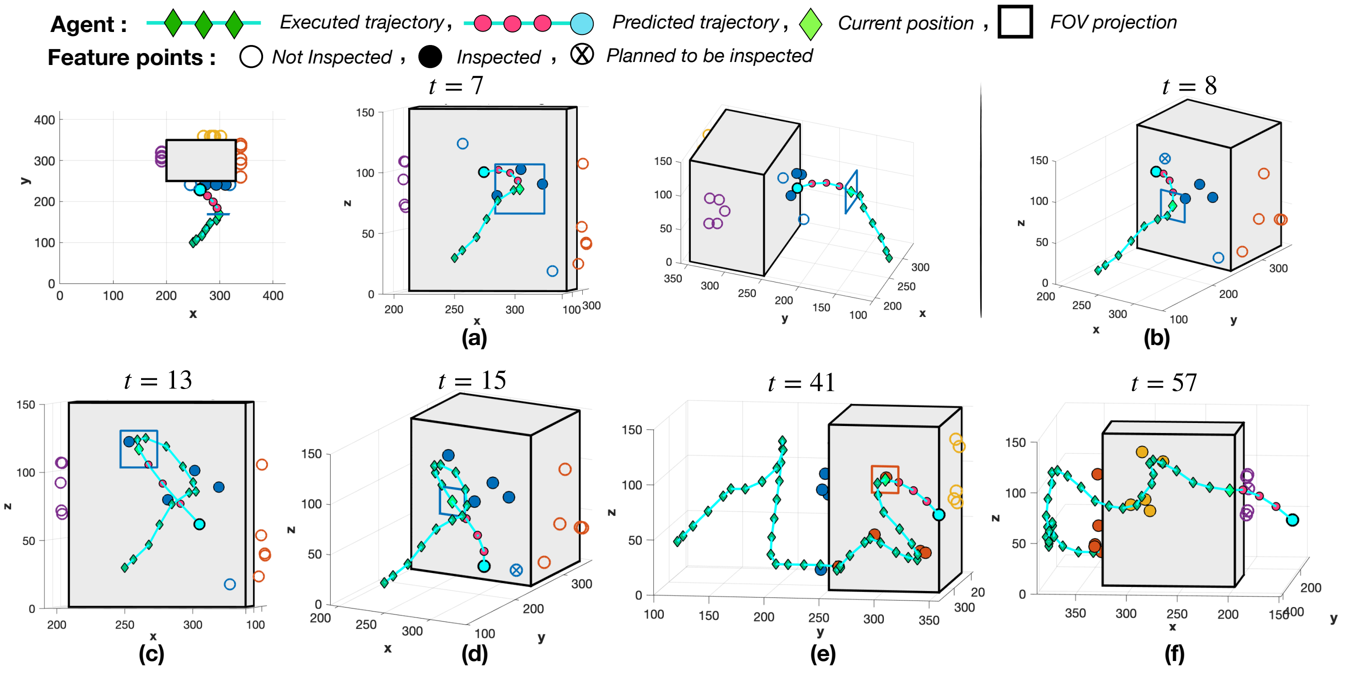

Fig. 6. The figure illustrated the UAV's generated inspection trajectory for a simulated scenario with a cuboid-like object of interest containing 20 feature-points.

inspect a cuboid-like object of interest of dimensions 130m by 100m by 150m as shown in Fig. 5. Specifically, the UAV agent needs to inspect all feature-points scattered on the object's surface (sampled uniformly) denoted by the colored circles (using different color per face) in Fig. 5(a)(b). As shown, each of the object's lateral faces contains 5 feature-points that need to be inspected by the agent. In this example the top and bottom face do not contain any feature-points, thus making the total number of feature-points that need to be inspected 20. Finally, the planning horizon T is set to 5 time-steps, the maximum mission time $\mathcal{T}_{\max}$ is set to 100 time-steps, the tuning weight w in Eqn. (14) is set to $w = 0.01$ and $D_{\max} = 100$m. Our evaluation has been conducted on a 2GHz laptop computer, with 8GB of RAM, running the Gurobi v9 solver.

B. Results

The UAV agent starts its mission from the (x, y)-coordinates $(250, 100)$, hovering 30m above the ground as shown in Fig. 5(c). As depicted in Fig. 6, the UAV's executed trajectory is denoted with $-\diamond-$ whereas the UAV's predicted trajectory (i.e., planned trajectory) is denoted by $-\circ-$. In Fig. 6, the feature-points which have not yet been inspected are shown with clear circles, whereas feature-points which have been inspected by the UAV agent are shown in solid circles. Moreover, feature-points which have been planned to be inspected at some point in the future inside the planning horizon will be shown as $\otimes$. As shown in Fig. 6, initially the UAV agent begins by approaching the face containing the blue feature-points. More specifically, between time-steps 1 and 6 the UAV agent is approaching the object of interest in order to place itself within the specified working distance $D_{\max}$ according to the constraint in Eqn. (8n) i.e., all feature-points must be inspected from a distance less or equal to $D_{\max}$. At each time-step the UAV agent solves a finite-horizon optimal control problem which seeks to find the control inputs inside the planning horizon which will maximize the number of feature-points that can be inspected, as discussed in detail in Sec. V-B. This is shown in Fig. 6(a), where at time-step $t = 7$, the agent manages to inspect simultaneously 3 feature-points from a single location. As it is shown, the 3 feature-points which are marked with solid blue circles reside inside the UAV's projected FOV denoted with a blue square. At the subsequent time-step, also observe that the agent's prediction plan is headed towards the nearest unobserved feature-point according to the objective function defined in Eqn. (14). As it is shown in Fig. 6(b) the feature-point with (x, y, z) coordinates $[245, 250, 120]$ has been marked for inspection as illustrated by the $\otimes$ symbol. Then, at time-step $t = 13$ the same feature-point is inspected as shown in Fig. 6(c). Observe that at $t = 13$ this feature-point resides within the agent's projected FOV. Subsequently, the UAV agent moves towards the nearest unobserved feature-point as shown in Fig. 6(d). The figure also illustrates the UAV's executed trajectory so far, along with its current prediction plan. Once all blue feature-points have been inspected, the UAV agent moves to the next face as depicted in Fig. 6(e). Here we can observe the generated UAV's trajectory for inspecting the orange feature-points (time-step $t = 41$). Finally, at time-step $t = 57$, the UAV agent has completed the inspection of the yellow feature-points as shown in Fig. 6(f), and is moving towards the object's final face which contains the purple feature-points.

Figure 7 shows the final inspection trajectory along with the applied control inputs. In this figure, the UAV's start and stop locations are marked with $\star$ and $\times$ respectively, and the inspection trajectory is color-coded according to the mission elapsed time. As it is shown in the figure all feature-points are inspected within 66 time-steps. Finally, Figure 8 shows the UAVs projected FOV along its inspection

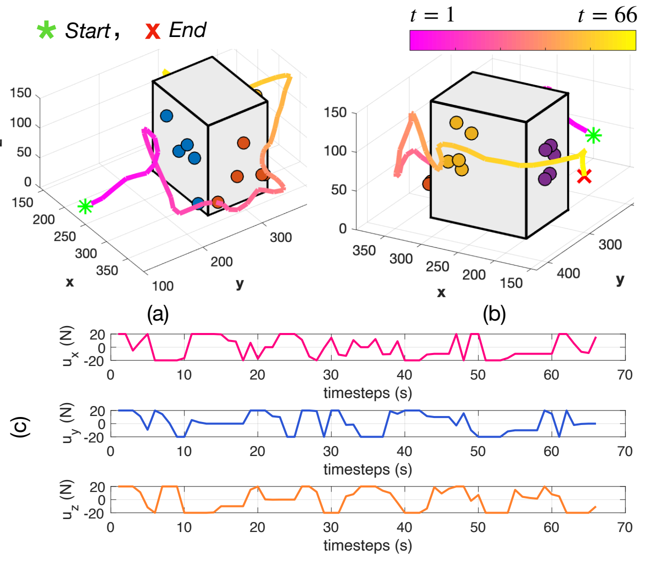

Fig. 7. The figure illustrates: (a)(b) The UAV's inspection plan in 3D used to inspect 20 feature-points scattered on the surface of the object of interest. (c) The control inputs used for generating the inspection trajectory.

trajectory. Observe, that the UAV's control inputs are selected in such a way so that resulting FOV projections contain all the feature-points that need to be inspected, as indicated by the colored solid circles.

We conclude our evaluation by discussing the computational complexity of the proposed approach. In general, the main optimization algorithm which is employed in order to tackle mixed integer programs (MIP) is that of branch-and-bound [38], and its complexity is usually due to the number of integral variables that are being utilized. Specifically, a branch-and-bound algorithm constructs a search tree by enumerating in systematic and consistent way candidate solutions for the MIP problem. Each node of this tree includes the original MIP problem constraints plus additional constraints on the bounds of the integer variables. The algorithm, proceeds by exploring nodes of the tree i.e., by solving a linear programming relaxation problem after dropping all integrality constraints. When the solution to the linear program consists of an integer constrained variable with a fractional value (i.e., $x = f$), the algorithm generates a new branch for this variable consisting of two sub-problems (i.e., nodes) where new integrality constraints are imposed (i.e., $x \leq \lfloor f \rfloor$ and $x \geq \lceil f \rceil$). Therefore, the size of this search tree grows with the number of integral variables and as a consequence the computational complexity grows as well.

As we have discussed in Sec. V the proposed approach uses a number of binary variables in order to implement the desired inspection planning behavior. As shown by the constraints in Eqn. (8r)-(8s) the number of required binary variables depends mainly on the number of feature-points that need to be inspected and on the length of the planning horizon. To better understand the computational complexity of the proposed controller, a Monte-Carlo simulation was conducted, where for the same object of interest we have varied the number of feature-points that need to be inspected and the length of the planning horizon, running 20 trials for each configuration and measuring the average runtime until the optimal solution is found. More specifically, we have run

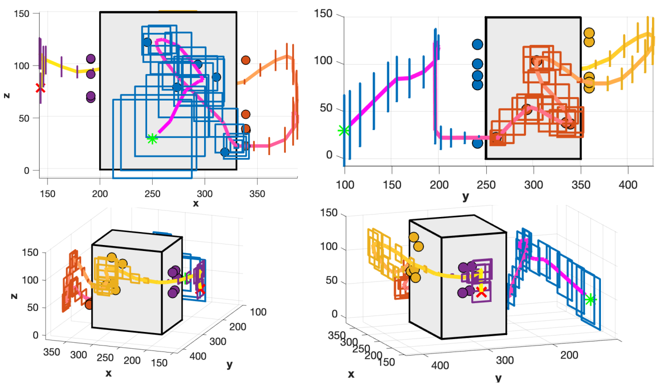

Fig. 8. The figure illustrates the UAV's projected FOV along its inspection trajectory, and its size in terms of side length.

Avg. Execution Time (sec)		
# feature-points	Horizon Length	Runtime
12	3	0.645
12	8	1.040
12	15	4.073
20	3	0.621
20	8	1.376
20	15	6.143
32	3	0.653
32	8	1.552
32	15	9.427

the proposed inspection planner with 12, 20 and 32 feature-points (in each trial the future points are randomly scattered on the object's surface area, with equal number of points per face), and with planning horizons of length 3, 8 and 15 time-steps. For this experiment the agent is always initialized from the same location, and the rest of the simulation parameters are set according Sec. VI-A.

Table VI-B summarizes the results of this experiment in terms of the average execution time (i.e., the time required by the solver to find the optimal solution). In particular, Table VI-B shows the average time (taken over the 20 trials) for each combination of the parameters. The results verify that the computational complexity increases as the number of feature-points and the length of the planning horizon increase. As it is shown, the length of the planning horizon has the largest impact on the performance of the proposed approach in terms of runtime (observe that the planning horizon is involved in every constraint listed in (P2)). Nevertheless, for some of the configurations of the parameters the proposed approach shows potential for real-time execution, considering that these results have been obtained on a 2GHz laptop computer. It is also worth noting that additional computational savings can be obtained through various heuristics and approximations [39]–[41] which can provide adequate near-optimal MIP solutions in real-time. A more thorough investigation of the real-time performance of the proposed approach and its real-world implemetation will be investigated in future works.

VII. Conclusion

In this work, we have tackled the problem of UAV-based automated planning, guidance and control for 3D inspection missions. We have formulated the inspection planning problem as a constrained receding horizon optimal control problem, in where the UAV's control inputs are optimally determined to enable the generation of efficient inspection trajectories of cuboid-like structures. We have derived a mixed-integer mathematical program which can be solved using off-the-shelf optimization solvers, and we have demonstrated its effectiveness through synthetic qualitative and quantitative experiments. In the future, we plan to extend the proposed approach to multiple UAV agents, and investigate its real-world performance.

Acknowledgments

This work is supported by the European Union's Horizon 2020 research and innovation programme under grant agreement No 739551 (KIOS CoE), and from the Republic of Cyprus through the Directorate General for European Programmes, Coordination and Development.

References

[1] S. Papaioannou, P. Kolios, T. Theocharides, C. G. Panayiotou, and M. M. Polycarpou, "Jointly-optimized searching and tracking with random finite sets," *IEEE Transactions on Mobile Computing*, vol. 19, no. 10, pp. 2374–2391, 2020.

[2] ——, "Decentralized search and track with multiple autonomous agents," in *2019 IEEE 58th Conference on Decision and Control (CDC)*. IEEE, 2019, pp. 909–915.

[3] ——, "A Cooperative Multi-Agent Probabilistic Framework for Search and Track Missions," *IEEE Transactions on Control of Network Systems*, 2020.

[4] ——, "Towards Automated 3D Search Planning for Emergency Response Missions," *Journal of Intelligent & Robotic Systems*, vol. 103, no. 1, pp. 1–19, 2021.

[5] ——, "3D Trajectory Planning for UAV-based Search Missions: An Integrated Assessment and Search Planning Approach," in *2021 International Conference on Unmanned Aircraft Systems (ICUAS)*. IEEE, 2021, pp. 517–526.

[6] D. C. Tsouros, S. Bibi, and P. G. Sarigiannidis, "A review on uav-based applications for precision agriculture," *Information*, vol. 10, no. 11, p. 349, 2019.

[7] P. Radoglou-Grammatikis, P. Sarigiannidis, T. Lagkas, and I. Moscholios, "A compilation of uav applications for precision agriculture," *Computer Networks*, vol. 172, p. 107148, 2020.

[8] J. C. Hodgson, S. M. Baylis, R. Mott, A. Herrod, and R. H. Clarke, "Precision wildlife monitoring using unmanned aerial vehicles," *Scientific reports*, vol. 6, no. 1, pp. 1–7, 2016.

[9] L. F. Gonzalez, G. A. Montes, E. Puig, S. Johnson, K. Mengersen, and K. J. Gaston, "Unmanned aerial vehicles (uavs) and artificial intelligence revolutionizing wildlife monitoring and conservation," *Sensors*, vol. 16, no. 1, p. 97, 2016.

[10] S. Papaioannou, P. Kolios, C. G. Panayiotou, and M. M. Polycarpou, "Cooperative simultaneous tracking and jamming for disabling a rogue drone," in *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, pp. 7919–7926.

[11] S. Papaioannou, P. Kolios, and G. Ellinas, "Downing a rogue drone with a team of aerial radio signal jammers," in *2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, pp. 2555–2562.

[12] A. Gasparetto, P. Boscariol, A. Lanzutti, and R. Vidoni, "Path planning and trajectory planning algorithms: A general overview," *Motion and operation planning of robotic systems*, pp. 3–27, 2015.

[13] 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*, vol. 57, no. 1, pp. 65–100, 2010.

[14] E. Galceran and M. Carreras, "A survey on coverage path planning for robotics," *Robotics and Autonomous Systems*, vol. 61, no. 12, pp. 1258–1276, 2013.

[15] T. M. Cabreira, L. B. Brisolara, and P. R. Ferreira Jr., "Survey on coverage path planning with unmanned aerial vehicles," *Drones*, vol. 3, no. 1, 2019.

[16] H. Choset and P. Pignon, "Coverage path planning: The boustrophedon cellular decomposition," in *Field and service robotics*. Springer, 1998, pp. 203–209.

[17] T. Danner and L. Kavraki, "Randomized planning for short inspection paths," in *Proceedings 2000 ICRA. Millennium Conference. IEEE International Conference on Robotics and Automation. Symposia Proceedings (Cat. No.00CH37065)*, vol. 2, 2000, pp. 971–976 vol.2.

[18] Y. Gabriely and E. Rimon, "Spanning-tree based coverage of continuous areas by a mobile robot," *Annals of mathematics and artificial intelligence*, vol. 31, no. 1, pp. 77–98, 2001.

[19] E. U. Acar, H. Choset, A. A. Rizzi, P. N. Atkar, and D. Hull, "Morse decompositions for coverage tasks," *The international journal of robotics research*, vol. 21, no. 4, pp. 331–344, 2002.

[20] W. H. Huang, "Optimal line-sweep-based decompositions for coverage algorithms," in *Proceedings 2001 ICRA. IEEE International Conference on Robotics and Automation (Cat. No. 01CH37164)*, vol. 1. IEEE, 2001, pp. 27–32.

[21] R. Mannadiar and I. Rekleitis, "Optimal coverage of a known arbitrary environment," in *2010 IEEE International conference on robotics and automation*. IEEE, 2010, pp. 5525–5530.

[22] A. Mavrinac and X. Chen, "Modeling coverage in camera networks: A survey," *International journal of computer vision*, vol. 101, no. 1, pp. 205–226, 2013.

[23] V. P. Munishwar and N. B. Abu-Ghazaleh, "Coverage algorithms for visual sensor networks," *ACM Transactions on Sensor Networks (TOSN)*, vol. 9, no. 4, pp. 1–36, 2013.

[24] Y. Guo and M. Balakrishnan, "Complete coverage control for nonholonomic mobile robots in dynamic environments," in *Proceedings 2006 IEEE International Conference on Robotics and Automation, 2006. ICRA 2006.*, 2006, pp. 1704–1709.

[25] C. Di Franco and G. Buttazzo, "Energy-aware coverage path planning of uavs," in *2015 IEEE International Conference on Autonomous Robot Systems and Competitions*, 2015, pp. 111–117.

[26] A. Xu, C. Viriyasuthee, and I. Rekleitis, "Optimal complete terrain coverage using an unmanned aerial vehicle," in *2011 IEEE International Conference on Robotics and Automation*, 2011, pp. 2513–2519.

[27] H. Wang, H. Li, C. Zhang, S. He, and J. Liu, "A 3d coverage path planning approach for flying cameras in nature environment under photogrammetric constraints," in *2017 36th Chinese Control Conference (CCC)*. IEEE, 2017, pp. 6761–6766.

[28] I. Maza and A. Ollero, "Multiple uav cooperative searching operation using polygon area decomposition and efficient coverage algorithms," in *Distributed Autonomous Robotic Systems 6*. Springer, 2007, pp. 221–230.

[29] B. Englot and F. S. Hover, "Three-dimensional coverage planning for an underwater inspection robot," *The International Journal of Robotics Research*, vol. 32, no. 9-10, pp. 1048–1073, 2013.

[30] J. O'Rourke, *Art gallery theorems and algorithms*. Oxford University Press Oxford, 1987, vol. 57.

[31] H. González-Baños, E. Mao, J.-C. Latombe, T. Murali, and A. Efrat, "Planning robot motion strategies for efficient model construction," in *Robotics Research*. Springer, 2000, pp. 345–352.

[32] C. Dornhege, A. Kleiner, and A. Kolling, "Coverage search in 3d," in *2013 IEEE International Symposium on Safety, Security, and Rescue Robotics (SSRR)*, 2013, pp. 1–8.

[33] W. Jing, J. Polden, W. Lin, and K. Shimada, "Sampling-based view planning for 3d visual coverage task with unmanned aerial vehicle," in *2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2016, pp. 1808–1815.

[34] Z. Zhang, X. Gao, and W. Wu, "Algorithms for connected set cover problem and fault-tolerant connected set cover problem," *Theoretical Computer Science*, vol. 410, no. 8-10, pp. 812–817, 2009.

[35] G. Laporte, A. Asef-Vaziri, and C. Sriskandarajah, "Some applications of the generalized travelling salesman problem," *Journal of the Operational Research Society*, vol. 47, no. 12, pp. 1461–1467, 1996.

[36] J. Xiao, J. Zhang, B. Adler, H. Zhang, and J. Zhang, "Three-dimensional point cloud plane segmentation in both structured and unstructured environments," *Robotics and Autonomous Systems*, vol. 61, no. 12, pp. 1641–1652, 2013.

[37] R. Anand, D. Aggarwal, and V. Kumar, "A comparative analysis of optimization solvers," *Journal of Statistics and Management Systems*, vol. 20, no. 4, pp. 623–635, 2017.

[38] J. E. Mitchell, "Branch-and-cut algorithms for combinatorial optimization problems," *Handbook of Applied Optimization*, vol. 1, pp. 65–77, 2002.

[39] E. Klotz and A. M. Newman, "Practical guidelines for solving difficult mixed integer linear programs," *Surveys in Operations Research and Management Science*, vol. 18, no. 1, pp. 18–32, 2013.

[40] V. V. Naik and A. Bemporad, "Exact and heuristic methods with warm-start for embedded mixed-integer quadratic programming based on accelerated dual gradient projection," *arXiv preprint arXiv:2101.09264*, 2021.

[41] G. Hendel, "Adaptive large neighborhood search for mixed integer programming," *Mathematical Programming Computation*, pp. 1–37, 2021.