Next-Best-View planning for surface reconstruction of large-scale 3D environments with multiple UAVs

Guillaume Hardouin[1,2], Julien Moras[1], Fabio Morbidi[2], Julien Marzat[1], El Mustapha Mouaddib[2]

***Abstract*— In this paper, we propose a novel cluster-based Next-Best-View path planning algorithm to simultaneously explore and inspect large-scale unknown environments with multiple Unmanned Aerial Vehicles (UAVs). In the majority of existing informative path-planning methods, a volumetric criterion is used for the exploration of unknown areas, and the presence of surfaces is only taken into account indirectly. Unfortunately, this approach may lead to inaccurate 3D models, with no guarantee of global surface coverage. To perform accurate 3D reconstructions and minimize runtime, we extend our previous online planner based on TSDF (Truncated Signed Distance Function) mapping, to a fleet of UAVs. Sensor configurations to be visited are directly extracted from the map and assigned greedily to the aerial vehicles, in order to maximize the global utility at the fleet level. The performances of the proposed TSGA (TSP-Greedy Allocation) planner and of a nearest neighbor planner have been compared via realistic numerical experiments in two challenging environments (a power plant and the Statue of Liberty) with up to five quadrotor UAVs equipped with stereo cameras.**

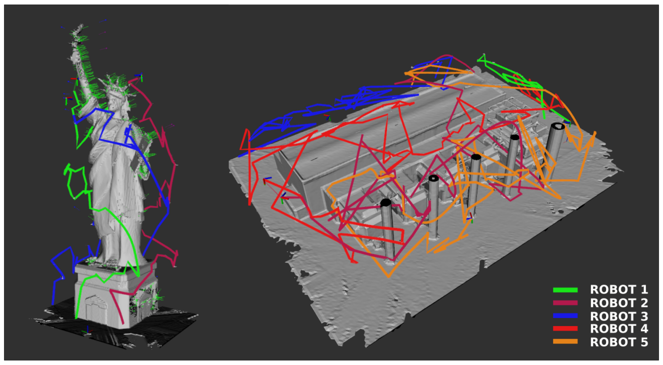

Fig. 1. Example outputs from our multi-UAV surface reconstruction algorithm. [left] The *Statue of Liberty* reconstructed by 3 UAVs; [Right] *Powerplant* reconstructed by 5 UAVs.

I. Introduction

Autonomous robots are being increasingly used today for time-consuming and dangerous tasks usually performed by human operators. For instance, aerial robots equipped with different on-board sensors (RGB-D, stereo cameras, laser range finders, etc.) hold great potential for modeling large-scale 3D structures. Recent applications include digital cultural heritage, exploration of confined and cluttered environments, and structural inspection for preventive maintenance [1]–[4]. Next-Best-View (NBV) planning [5], is a planning method in which a robot iteratively computes the best viewpoint configurations to fully reconstruct the 3D environment, which can be (partially) known in advance or completely unknown. In the latter case, the robot is forced to dynamically discover the surrounding environment in the course of the mission. The focus can be either on the exploration of the 3D volume (*exploration problem*) or on the consistency and completeness of the reconstructed surface (*inspection problem*), for which different path-planning strategies have been developed in the literature. When a single aerial robot is used, the time necessary to cover large-scale environments may be prohibitively long, and incompatible with limited on-board power resources and flight autonomy (15-20 minutes for a standard battery-powered quadrotor). This problem can be alleviated by performing the requested task with a team of cooperating robots.

In this paper, we extend our previous work [1] on incremental exploration and surface reconstruction of an unknown 3D environment, to a fleet of Unmanned Aerial Vehicles (UAVs). Differently from the classical surface frontier-based approaches [6], [7], we identify the incomplete areas directly from the online map and generate candidate sensor viewpoints in the free space, to complete them. The proposed NBV planning method greedily assigns these configurations to the robots, according to the "utility" (i.e. information gain) they locally provide to the fleet and known map (see Fig. 1). Optimal paths are computed by solving a variant of the Travelling Salesman Problem (TSP) [8], and updated when unknown areas are completed. In summary, the original contributions of this paper are threefold:

- A multi-UAV NBV planner is designed to visit viewpoint configurations for surface reconstruction. It relies on a greedy allocation of configurations and on the successive (approximate) resolution of the TSP.
- A single volumetric representation of surfaces, the Truncated Signed Distance Function (TSDF) [9], is used to cautiously navigate, identify incomplete areas, and evaluate the information gain and the quality of 3D reconstruction.
- The proposed method has been extensively validated in two realistic simulation environments.

The remainder of this paper is organized as follows. In Sect. II, we review the existing literature on multi-robot 3D reconstruction and NBV planning. Sect. III is devoted to the problem formulation, and in Sect. IV the

[1]DTIS, ONERA, Université Paris-Saclay, 91123 Palaiseau, France. Emails: firstname.lastname@onera.fr

[2]MIS laboratory, Université de Picardie Jules Verne, 80000 Amiens, France. Emails: {fabio.morbidi, mouaddib}@u-picardie.fr

This work has been supported by ONERA DTIS and by the Hauts-de-France region, through the research project ScanBot, "Robotic Scanners for Automatic 3D Modeling of Cultural Heritage" (2018-2021).

proposed method is described. The results of our numerical experiments, including a comparative study, are discussed in Sect. V. Finally, Sect. VI outlines the main contributions of the paper, and presents some possible directions for future research.

II. Related Work

The ability to plan informative paths for online exploration and modeling of 3D environments, is a fundamental skill that highly increases the level of autonomy of a robot. Numerous strategies for 3D modeling of an unknown environment with a sensor-equipped robotic platform, are available in the literature. In this paper, we focus on NBV methods which determine the best viewpoints to visit depending on the nature of the robot's mission. *Surface inspection* methods analyze the reconstructed surface for viewpoint definition. The goal is to ensure that the reconstruction is accurate and complete. Among them, *frontier-based* methods generate viewpoint configurations on the frontier of the observed surface, which satisfy some orientation, positioning, and sensing constraints. These configurations, visited by the sensor-equipped robot, provide new information about the surface, with some overlapping to ensure continuity [5], [10], [11]. These methods have been mostly applied for object reconstruction with robot manipulators, under strong assumptions on the navigable free space. However, several works have extended NBV to mobile robots by considering a volumetric representation [10], [12] and more recently the TSDF [1], [13], [14]. On the other hand, *volumetric exploration* aims at exploring a predefined volume containing the object of interest. It usually proceeds by building a map of a large unknown environment by using a 3D grid model (such as Octomap [15]) to identify known, unknown and occupied areas. The majority of recent exploration methods rely on *sampling-based* planning [16]–[18]. They expand a random tree (e.g. RRT/RRT* [19]) of sampled sensor configurations in the free space, and select the NBV trajectory that guarantees the maximum coverage of the volume. Such receding horizon methods are efficient, but the accuracy of the reconstructed surface is not explicitly taken into account. In order to guarantee fast exploration and reconstruction accuracy, some recent methods have tried to combine the two approaches. In [4], the authors subdivided the reconstruction into two phases: in the first one, a coarse model of the environment is obtained, while in the second phase the reconstructed surface is refined. Other methods try to cover the whole surface model by reasoning on the occupancy map [20], and the exploration path is refined by accounting for surface incompleteness [3]. However, a drawback of these strategies is that they rely on multiple volumetric and surface representations at the same time, and only a few methods deal directly with the surface inspection problem [1], [14].

The aforementioned works only feature a *single robot*, and online incremental reconstruction with multiple cooperating robots is an open problem in the literature [21]. The authors in [22] were the first to propose how to compute frontier cells, and to define the trade-off between their distance cost and utility for a multi-agent exploration mission, with robots equipped with laser scanners in a 2D environment. A cooperative frontier-based approach for a team of UAVs flying in uncluttered (outdoor) areas, has been also recently proposed in [23]. It is one of the first works to consider a 3D space representation for multi-robot exploration, with a centralized Octomap built on a ground station, which is also used to coordinate the motion of the UAVs (RRT*-based path planning). The method was evaluated using realistic numerical experiments (ROS/Gazebo) with simulated stereo sensors. The approximation of the mutual information of range sensors [24] also led to the development of exploration approaches based on probabilistic occupancy maps with entropy reduction, such as decMCTS [25] or SGA [26]. Recently, in [27], [28], a finite-horizon decentralized planner (DGSA) has been designed using sampling-based Monte Carlo Tree Search (MCTS) [29]. The trajectories of the robots are assigned by solving a submodular maximization problem over matroid constraints with greedy assignment heuristics, for which polynomial-time algorithms and suboptimality bounds can be established [30]. However, these mutual-information methods rely on occupancy mapping, and they do not exploit any surface representation. Therefore, it might be problematic, in practice, to assess the quality of 3D reconstruction.

In this paper, we explicitly address the *surface inspection problem* by considering a NBV frontier-based planning strategy for multiple UAVs in large-scale environments. Viewpoint configurations are clustered according to their location in space to evaluate the interest of visiting specific regions and find a suitable path. We use the property of submodular set functions to formalize the assignment problem [30], [31], and to allocate sensor configurations using a TSP-based greedy local search. Successive approximate resolutions allow to cover the unknown environment and complete the reconstruction.

III. Problem formulation

This paper builds upon our previous work [1], and extends it to a multi-robot setting. A fleet of N identical UAVs with 4 degrees of freedom (the 3D position $[x, y, z]^T \in \mathbb{R}^3$ and the yaw angle $\psi \in \mathcal{S}^1$) is considered. Each UAV is equipped with a forward-facing depth sensor with limited field of view (FOV) and sensing range, extrinsically calibrated with respect to the body frame of the aerial robot. The UAVs should scan an unknown 3D environment (for instance, a building), characterized by its surface. A mapping algorithm is required in order to build a representation of the reconstructed surface as a collection of *voxels*, and to estimate *unknown*, *occupied* and *empty* space (for this, a TSDF representation [9] was considered). A voxel is said to be *scanned* if it has been seen by a UAV. We assume that the UAVs are equipped with an accurate localization system which allows to estimate their pose with respect to a global reference frame, and that a robust lower-level trajectory-tracking algorithm is available (Model Predictive Control was used in our implementation [32]).

The incompleteness of the surface model is defined as follows:

Definition 1 (Incomplete surface element). We call *Incomplete Surface Element (ISE)*, a voxel $\mathbf{v}$ lying on the surface at a frontier, near both the unknown and empty space. We denote by C the set of all ISEs.

For a rigorous definition of unknown, occupied and empty space, the reader is referred to Sect. IV-B.

Definition 2 (Remaining incomplete surface). Let Q be the set of all collision-free configurations $\mathbf{q} = [x,\, y,\, z,\, \psi]^T$ of a UAV, and let $Q_c \subseteq Q$ be the set of all configurations $\mathbf{q}$ from which an ISE $\mathbf{v} \in C$ can be scanned. The *remaining incomplete surface* is then defined as $C_{\text{rem}} = \bigcup_{\mathbf{v}\,\in\, C} \{\mathbf{v} \mid Q_c = \emptyset\}$.

The vector-valued function $\mathbf{p}^i_{j,k}(s) : [0,\, 1] \to \mathbb{R}^3 \times \mathcal{S}^1$ defines the path of UAV i from configuration j to configuration k, where $\mathbf{p}^i_{j,k}(0) = \mathbf{q}^i_j$ and $\mathbf{p}^i_{j,k}(1) = \mathbf{q}^i_k$, $i \in \{1, \ldots, N\}$. We assume that $\mathbf{p}^i_{j,k}(s)$ is collision-free and feasible for UAV i (i.e. the kinematic/dynamic constraints of the aerial robot are satisfied along the path). We are now ready to state the problem studied in this paper.

Problem 1 (Inspection with a fleet of UAVs). Given a fleet of N identical UAVs with initial configurations $\mathbf{q}^i_0 \in Q$, $i \in \{1, \ldots, N\}$, find collision-free paths $\mathbf{p}^i_{0,f}(s)$ between $\mathbf{q}^i_0$ and the final configurations $\mathbf{q}^i_f$, which allow the aerial vehicles to scan the set $C_{\text{ins}} = C \setminus C_{\text{rem}}$ of all ISEs.

IV. Proposed Approach

In this paper, we consider a centralized architecture where the mapping and planning information is shared among all the UAVs, and the communication is assumed to be perfect. The general flowchart of our algorithm is depicted in Fig. 2. During the exploration, the poses and depth maps of each UAV are sent to a ground station and the volumetric map is incrementally built (Sect. IV-A). Then, the ISE extractor is used to identify sets of ISEs. Configurations which allow to complete them are generated and clustered, depending on their location in the 3D space (Sect. IV-B). The planner generates and continuously expands a directed graph which represents the travel utility between clusters in the free space. In order to maximize a cumulative utility function for the robots, paths are extracted from the graph ensuring collision-free navigation and they are broadcast to the UAVs. As the map is updated, new ISEs are uncovered and the path to complete them is updated (Sect. IV-C). The 3D model is considered complete when no ISEs are left.

A. Surface-based mapping

Surface-based reconstruction is performed over time to allow detection of incomplete areas. The depth maps sensed by the UAVs are integrated in a TSDF volumetric map M, which consists of a voxel grid, where each voxel contains a truncated signed distance value $\phi \in \mathbb{R}$ and a weight $w \geq 0$. The model is progressively built by integrating the

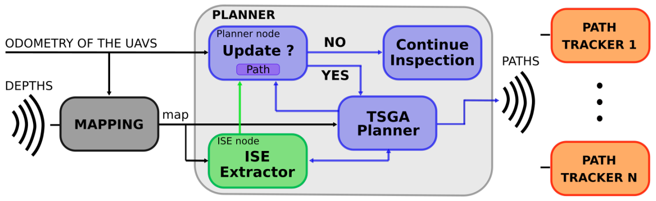

Fig. 2. General flowchart of the reconstruction algorithm: the architecture of the planner is shown inside the shaded box.

depth measurements via an inverse-squared distance increment $1/z^2_{\mathbf{q}}(\mathbf{v})$ for the weight w, where $z_{\mathbf{q}}(\mathbf{v})$ is the distance between voxel $\mathbf{v}$ and the current sensor pose $\mathbf{q}$. With this choice, the sensing error is minimized [33], [34]. This model implicitly represents a surface, which corresponds to the zero level set of the distance field: hence, the TSDF volume is a *volumetric representation of a surface*. Depending on the values of ϕ and w, one can determine whether a voxel is *unknown* or *known*, i.e. *occupied* or *empty*. A voxel $\mathbf{v}$ is considered *known* if $w(\mathbf{v}) \geq W_{\text{th}}$ and *unknown* if $w(\mathbf{v}) < W_{\text{th}}$, where the threshold W_{th} depends on the sensing range of the depth sensor.

B. ISE extractor and viewpoint generation

Using the proposed surface model and following [13], a voxel $\mathbf{v} \in M$ is an ISE, i.e. $\mathbf{v} \in C$, if the following three conditions are fulfilled:

a) $w(\mathbf{v}) \geq W_{\text{th}} \wedge \phi(\mathbf{v}) > 0,$ (empty)

b) $\exists\, \mathbf{u} \in \mathcal{N}^6_{\mathbf{v}}$ s.t. $w(\mathbf{u}) < W_{\text{th}},$ (unknown)

c) $\exists\, \mathbf{o} \in \mathcal{N}^{18}_{\mathbf{v}}$ s.t. $w(\mathbf{o}) \geq W_{\text{th}} \wedge \phi(\mathbf{o}) \leq 0,$ (occupied)

where $\mathcal{N}^6_{\mathbf{v}}$ and $\mathcal{N}^{18}_{\mathbf{v}}$ denote the 6- and 18-connected voxel neighborhoods of $\mathbf{v}$, respectively. With reference to Definitions 1 and 2, we introduce the notion of *scanned element*:

Definition 3 (Scanned element). A voxel $\mathbf{v} \in M$ which satisfies $w(\mathbf{u}) \geq W_{\text{th}}$, $\forall\, \mathbf{u} \in \mathcal{N}^6_{\mathbf{v}}$, is called a *scanned element*.

The determination of a direction $\mathbf{n}_{\mathbf{v}}$ to observe the ISE $\mathbf{v}$, is based on the gradient of the weight function $\nabla w(x, y, z)$, which can be computed as,

$$\mathbf{n}_{\mathbf{v}} = \sum_{\mathbf{c}\,\in\,\mathcal{N}^{26}_{\mathbf{v}}} w'(\mathbf{c})\, \frac{\mathbf{c} - \mathbf{v}}{\|\mathbf{c} - \mathbf{v}\|}\,,$$

where $\mathcal{N}^{26}_{\mathbf{v}}$ is the 26-connected neighborhood of $\mathbf{v}$ and the weight function w' is

$$w'(\mathbf{c}) = \begin{cases} -W_{\text{th}} & \text{if voxel } \mathbf{c} \text{ is occupied,} \\ W_{\text{th}} & \text{otherwise.} \end{cases}$$

Note that $\mathbf{n}_{\mathbf{v}}$ is not a unit vector. A new sensor configuration is generated along the direction $\mathbf{n}_{\mathbf{v}}$ at a distance δ_{pose} from voxel $\mathbf{v}$ (see Fig. 3). The sensor points towards $\mathbf{v}$ along $-\mathbf{n}_{\mathbf{v}}$ and the value of δ_{pose} depends on the sensing range of the depth camera. Two poses $\mathbf{q}_j, \mathbf{q}_k \in Q$ generated from the ISEs $\mathbf{v}_j, \mathbf{v}_k \in C$, respectively, may be very close, i.e.

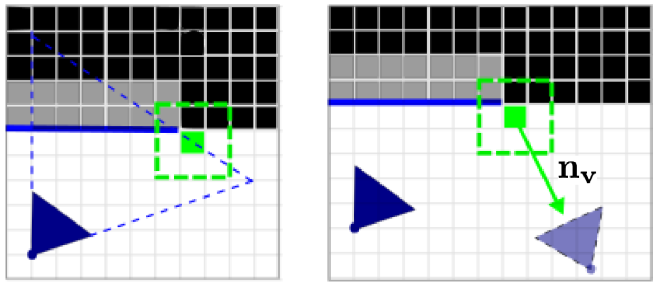

Fig. 3. [left] Two-dimensional example of ISE $\mathbf{v}$ (filled green square). Its 2D neighborhood is represented by a dashed green square. Unknown voxels are black, occupied are gray, and empty voxels are white. The reconstructed surface is depicted as a blue segment, and the sensor configuration and its frustum as dark blue triangles; [right] Direction from the contour, $\mathbf{n_v}$, and corresponding viewpoint configuration at distance δ_{pose} from $\mathbf{v}$ (light blue triangle).

$\text{dist}(\mathbf{q}_j, \mathbf{q}_k) < \epsilon$ for a small $\epsilon > 0$, and the viewing directions $\mathbf{n}_{\mathbf{v}_j}$, $\mathbf{n}_{\mathbf{v}_k}$, almost parallel, i.e. $|\mathbf{n}_{\mathbf{v}_j} \cdot \mathbf{n}_{\mathbf{v}_k}| \simeq 1$. These configurations are then aggregated into a single viewpoint by averaging their positions and orientations to reduce the number of overall poses.

Large-scale environments may result in a prohibitive number of possible viewpoint configurations, which are impractical for planning purposes. To overcome this issue and easily identify the most promising areas to scan, the viewpoints are grouped into *clusters* u_j, $j \in \{1, 2, \ldots, N_c\}$, depending on their location in space. We denote by $U = \{u_1, u_2, \ldots, u_{N_c}\}$ the set of all clusters. A configuration $\mathbf{q}_l$ belongs to a generic cluster u if $\exists\, \mathbf{q}_j \in u$ such that $\text{d}(\tau_l^j) < d_\nu$, where $\text{d}(\tau_l^j)$ denotes the length of the path τ_l^j between $\mathbf{q}_l$ and $\mathbf{q}_j$ on a directed graph we will define in Sect. IV-C, and d_ν is an upper bound on the distance. If no neighbors are found, d_ν is increased up to a maximum value $d_\nu^{\max}$.

Once the clusters have been defined, their level of informativeness should be quantified. To evaluate a configuration, we use the ray-tracing method [35] from a frontier-based perspective, i.e. we count the number of ISEs that can be seen. Let $C_\mathbf{q}$ be the set of all ISEs seen from viewpoint $\mathbf{q}$ and let $C_u = \bigcup_{\mathbf{q} \in u} C_\mathbf{q}$. The *gain* $g(u)$ of cluster u is defined as,

$$g(u) = \frac{|C_u|}{|C|}, \tag{1}$$

where $|C_u|$ denotes the cardinality of the set C_u.

C. Next-Best-View planning

The planner globally allocates clusters to the UAVs and schedules their visit according to a given common TSDF map. To formalize this idea, let us introduce the weighted directed graph $\mathcal{G} = (U, E, \{a_{uv}\}_{(u,v) \in E})$, where U is the set of nodes (in our case, the clusters), E is the set of edges, and $\{a_{uv}\}_{(u,v) \in E}$ is the collection of weights for the edges. Each edge $e_{uv} \in E$ is directed and links cluster u to v, with $u, v \in U$. Let us assume that the initial configuration of UAV i belongs to one of the clusters of $\mathcal{G}$, i.e. $\mathbf{q}_0^i \in U$. Let $\mathbf{q}_k^i$ be a configuration of cluster u, and $\mathbf{q}_l^i$, $\mathbf{q}_m^i$ two configurations of cluster v. Then, the weight a_{uv} between cluster u and v is the 6-tuple defined as,

$$a_{uv} = \{\tau_k^l,\, \tau_l^m,\, g(v),\, \text{d}(\tau_k^l),\, \text{d}(\tau_l^m),\, f_{uv}\}, \tag{2}$$

where

- τ_k^l denotes the path from $\mathbf{q}_k^i \in u$ to $\mathbf{q}_l^i \in v$, i.e. the path between cluster u and cluster v. We select $\mathbf{q}_l^i$ among all the configurations of v, so that τ_k^l is the shortest possible path,
- τ_l^m denotes the shortest Hamiltonian path [36] including configurations of v, which starts at $\mathbf{q}_l^i$ and ends at $\mathbf{q}_m^i$,
- $g(v)$ is the gain of cluster v, as defined in (1),
- $\text{d}(\tau_k^l)$ is the cost associated with the inter-cluster path τ_k^l, i.e. the length of τ_k^l,
- $\text{d}(\tau_l^m)$ is the cost associated with the intra-cluster path τ_l^m, i.e. the length of τ_l^m,
- f_{uv} is the *utility function* defined as,

$$f_{uv} = g(v) \exp\left(-\lambda_{\text{tc}}\, \text{d}(\tau_k^l) - \lambda_{\text{ic}}\, \text{d}(\tau_l^m)\right), \tag{3}$$

  where λ_{tc} and λ_{ic} are positive penalty terms for the inter-cluster and intra-cluster costs, respectively, which can be used to promote the visit of clusters far apart or large clusters. A similar utility function was originally proposed in [37].

The weights on the directed graph $\mathcal{G}$ in (2), quantify the potential benefit of choosing a path to pursue the 3D reconstruction: the higher the value of the function f_{uv}, the more beneficial is the path. Since the formulation of the inspection problem depends on the number of robots, we will separately analyze the case of a single UAV and of multiple UAVs.

1) Single UAV: If we consider a single UAV as in [1], we can formalize the inspection problem as a *maximum Asymmetric Travelling Salesman Problem* (maxATSP), i.e. as the problem of finding a maximum-utility Hamiltonian path $\mathbf{p}$ on $\mathcal{G}$. In what follows, we will denote by $\text{maxATSP}(U)$ the set function that takes the set of clusters U and outputs its utility value p, from which path $\mathbf{p}$ is computed. In practice, the ATSP is solved by first converting it into a symmetric TSP (i.e. a standard TSP) and then using the well-known Lin-Kernighan heuristic [38].

2) Multiple UAVs: In the multi-robot case, the clusters should be suitably allocated to the UAVs, and a cluster assignment problem should be solved. Let U^i be the set of clusters assigned to robot i, such that $\bigcup_{i=1}^N U^i = U$. Then, the assignment problem can be stated as follows,

$$\max_{U^1, \ldots, U^N \subset U} \left\{ \sum_{i=1}^{N} \text{maxATSP}(U^i) \;\middle|\; U^i \cap U^\ell = \emptyset, \right. \\ \left. i \neq \ell,\, \bigcup_{i=1}^{N} U^i = U \right\}, \tag{4}$$

where $\sum_{i=1}^N \text{maxATSP}(U^i)$ is a non-decreasing submodular set function [30], and the space of feasible paths has the structure of a simple partition matroid. The submodular maximisation problem (4) can be approximately solved using local greedy heuristics [31], such as the TSP-Greedy Allocation (TSGA) procedure reported in **Algorithm 1**. Note that

Algorithm 1: TSP-greedy allocation (TSGA)

```
Set $U^i = \emptyset$ and $p^i = 0, \forall\, i \in \{1, \dots, N\}$;
**foreach** *cluster* $v \in U$ **do**
    $i \leftarrow \underset{k \in \{1,\dots,N\}}{\arg\max} \{\text{maxATSP}(U^k \cup v) - p^k\}$;
    $U^i \leftarrow U^i \cup v$ ;
    $p^i \leftarrow \text{maxATSP}(U^i)$;
    $\mathbf{p}^i_{U^i} \leftarrow \{p^i, U^i\}$;
Broadcast the paths $\{\mathbf{p}^1_{U^1}, \dots, \mathbf{p}^N_{U^N}\}$ to the UAVs;
```

while provable worst-case bounds on the suboptimality can be established [39], a detailed mathematical analysis is beyond the scope of this paper and is not reported here. The TSGA algorithm greedily assigns a cluster to a UAV, when it locally maximizes the overall utility of the fleet of N robots. Unlike classical insertion methods where a cluster is added to a robot's path [39], the maxATSP problem is solved for the extended cluster set $U^i \cup v$, which results in a more efficient path for UAV i. Once the clusters have been assigned and the corresponding paths $\mathbf{p}^1_{U^1}, \dots, \mathbf{p}^N_{U^N}$ computed, they are broadcast to each UAV. This strategy maximizes the individual utility of the UAVs over disjoint sets, to maximize fleet-wise utility, and it is amenable to a distributed implementation in which only local information is used (such as, local free space, ISEs, U^i related to the local map of UAV i).

The maximization of the utility function over the long run, prompts the UAVs to explore the areas which appear to be the most valuable in terms of completeness and quality. This might induce a UAV to visit a configuration that covers an area containing multiple ISEs and scan them all (cf. equ. (1)). It is then pointless to visit configurations associated to ISEs that have already been scanned. To avoid unnecessary visits, the planner computes the remaining ISEs of paths since the last iteration, and possibly updates the plan. In practice, a path is updated when 50% of the ISEs whose visit is currently scheduled, has been completed. This update rule ensures a reactive visit of revealed ISEs as the map grows.

Note that **Algorithm 1** may allocate paths of various length: hence, the UAVs may conclude their tours at different time instants. To reduce the idle time of the UAVs finishing first, we asynchronously assign new paths to them in order to visit clusters which are not currently allocated to any robot (e.g. if UAV i has not completed its exploration round, the set of clusters assigned to the fleet becomes $U \setminus U^i$). The reconstruction procedure stops when no more ISEs are left.

V. Numerical experiments

The proposed surface-driven method has been validated via realistic numerical experiments. We compared the TSGA planner with another multi-robot planner, the Nearest Neighbor (NNB) greedy algorithm, for a fleet of 3 and 5 UAVs. In NBB, only one cluster is allocated to each robot by locally computing $\max_{v \in U} f_{uv}$ for the updated map. Replanning is thus very fast compared to TSGA, but only one cluster at a time is set to be visited. We also evaluated the 3D reconstruction accuracy achieved by the fleet and by a single UAV, using the strategy proposed in [1].

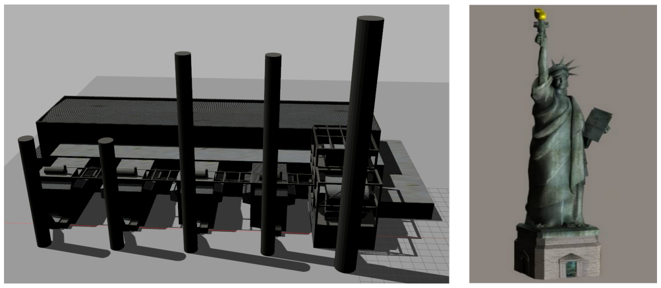

Fig. 4. Gazebo simulation environments: [left] **Scenario 1**, Powerplant; [right] **Scenario 2**, the Statue of Liberty.

1) Experimental setup: An industrial plant benchmark, which is widely used in the volumetric exploration literature, has been chosen (**Scenario 1**) [3]. In order to study the impact of the two penalty terms in the utility function (3), on the reconstruction accuracy/completeness, we also considered a monumental statue (**Scenario 2**) [20], see Fig. 4. The simulation parameters used in the two scenarios are reported in Table I. We used the RotorS simulator [40] to model quadrotor UAVs equipped with a stereo camera, in the ROS-Gazebo[1] environment. To represent depth map uncertainty, we considered a Gaussian noise model. The standard deviation associated with a pixel corresponds to the depth-value sensing error of the corresponding point located at a distance z, i.e. $\sigma(z) = \frac{|e_d|}{fB} z^2$, where $|e_d|$ is the magnitude of the disparity error, f the focal length in pixels, and B the baseline of the stereo camera in meters. Following [33], [41], the raw depth map was smoothed out by using a 3×3 kernel. The TSDF volume was generated with the algorithm proposed in [42], where the reconstruction is performed with MarchingCubes [43], and the weight increment has been modified to be quadratic, as reported in Sect. IV-A. Collision-free UAV paths have been computed using the Lazy

[1]`https://ros.org/, http://gazebosim.org/`

TABLE I
Parameters used in the numerical experiments.

Parameter	**Scenario 1**	**Scenario 2**
Voxel resolution $r_{\mathbf{v}}$ [m]	0.3	0.15
Threshold W_{th}	0.3	0.3
$e_{\max}$ [m]	0.2598	0.1299
Camera range [m]	[1.6, 8]	[1, 5]
Camera FOV [deg.] (H, V)	90 × 60	90 × 60
e_d [pixels]	0.1	0.1
f [pixels]	376	376
B [m]	0.11	0.11
Collision radius [m]	1	1
UAV nominal speed [m/s]	0.5	0.5
δ_{pose} [m]	4.7	3.6
d_ν [m]	2.0	2.5
$d_\nu^{\max}$ [m]	5	5
Penalty term λ_{tc}	0.3	0.17
Penalty term λ_{ic}	0.03	0.15

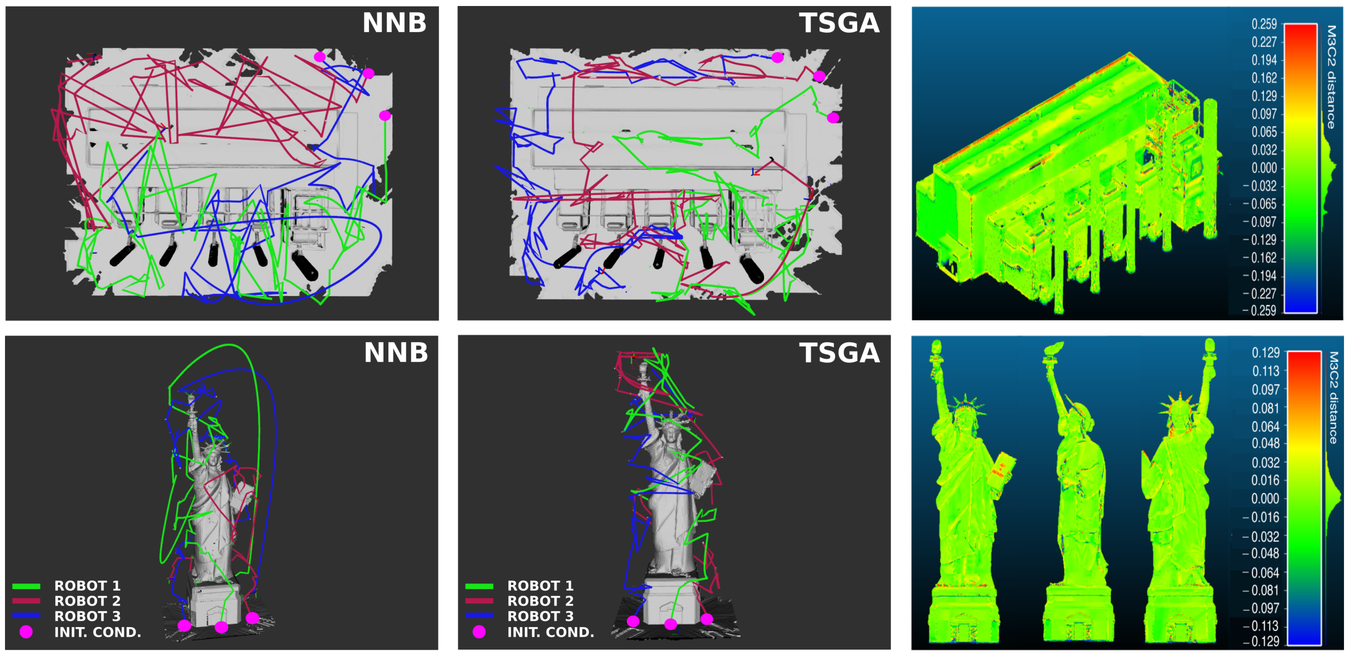

Fig. 5. [top] **Scenario 1** and [bottom] **Scenario 2**: Reconstructed mesh and 3D exploration paths $\mathbf{p}^1_{0,f}$, $\mathbf{p}^2_{0,f}$, $\mathbf{p}^3_{0,f}$ (green, red, blue) of 3 UAVs (the initial locations are marked with magenta dots); [left] NNB planner; [middle] TSGA planner; [right] Signed distance error: the color coding shows the error in meters with respect to the ground truth, computed with CloudCompare's M3C2 algorithm.

PRM* planner from the Open Motion Planning Library [44]: in this way, we can find the shortest path between two configurations by taking the structure of the TSDF map and the current location of other UAVs into account (the collision radius was set to 1 m). Lazy PRM* allows multi-query path planning to all destination points, which is useful for reachable-path checking and distance evaluation, because of the reduced computational complexity compared to RRT (the average runtime is below 1 s). The UAVs track the generated paths using Model Predictive Control [32]: the reference translational velocity was fixed at 0.5 m/s. In **Scenario 1**, the popular Powerplant model[2] was scaled to fit in a box of size $65 \times 42 \times 15\,\mathrm{m}^3$ (as a consequence, the five flues have the same height, see Fig. 4). Because of its narrow passages, high walls and roof, large flues and thin gantries, Powerplant is challenging for both navigation and reconstruction (occlusion problem). In **Scenario 2**, we considered a model of the Statue of Liberty[3] ($20 \times 20 \times 60\,\mathrm{m}^3$), which contains multiple sharp edges and fine details. In both scenarios, the UAVs are initially located in the same area, around a ground station (magenta dots in Fig. 5) The results of our numerical experiments are reported in Table II. The single-UAV planner with perfect and noisy depth measurements (denoted by [1] and [1]*, respectively), is compared here with the TSGA and NNB planners for 3 and 5 UAVs. To obtain statistically-significant values, 10 trials per scenario were carried out. We ran all tests on a Dell Precision 7520 laptop with 2.90 GHz Intel Core i7 processor, 16 GB RAM and Quadro M2200 graphics card.

[2]http://models.gazebosim.org/
[3]https://free3D.com/

2) Metrics: The single-UAV planner proposed in [1] is used as a baseline to evaluate our new multi-robot strategy, in terms of cumulated path length and completion time (to cover the entire 3D environments). This includes travel time and sensing time (e.g. one depth map integration and map update). The reconstructed 3D surface has been evaluated with CloudCompare[4] using the M3C2 (Multiscale Model to Model Cloud Comparison) algorithm [45]. To quantify how well the surface has been recovered, dense point clouds were sampled on the reconstructed and ground truth (GT) meshes, and their deviation was measured by performing a cloud-to-cloud comparison (see Fig. 5 [right]). For a fair evaluation, all the invisible surfaces of the GT mesh were pruned beforehand (e.g. the interior floor and walls), and the analysis was restricted to the exterior surface mesh only. A point belonging to the GT point cloud was considered to be covered by a corresponding one in the reconstructed cloud, if their absolute distance was less than the length of the half diagonal of a voxel, i.e. $e_{\max} = r_{\mathbf{v}}\sqrt{3}/2$, where $r_{\mathbf{v}}$ is voxel's resolution. The quality of the recovered surface is evaluated in Table II, by reporting the average and standard deviation of the signed distance error with respect to the GT point cloud, and the root-mean-square error (RMSE).

3) Penalty terms: The choice of the penalty terms λ_{tc} and λ_{ic} appearing in the utility function (3), depends on the nature of the 3D environment explored by the UAVs. **Scenario 1** and **2** are, in this respect, two representative examples. In a wide, box-like environment as **Scenario 1**, the ISEs tend to appear in the proximity of occluded regions and sharp edges,

[4]https://danielgm.net/cc/

TABLE II

Results of the numerical experiments (statistics over 10 trials).

	Scenario 1						Scenario 2					
Number of UAVs	1		3		5		1		3		5	
Algorithm	[1]	[1]*	NNB	TSGA	NNB	TSGA	[1]	[1]*	NNB	TSGA	NNB	TSGA
Path length [m]	780	785	1038	790	1113	879	547	550	733	721	580	574.2
Completion time [min.]	32	33′10″	11′09″	10′20″	6′51″	6′22″	36′	37′	13′10″	10′18″	7′30″	6′45″
Time gain [%] w.r.t. [1]*	–	–	66.4	68.8	79.4	80.8	–	–	64.4	72.2	79.7	81.8
Surface coverage [%]	91.5	90.4	90	91	90.5	90.6	92.3	91.2	91.1	91	90.9	91.1
M3C2 avg. error [cm]	0.14	−0.13	−0.15	−0.26	−0.11	−0.3	0.29	−0.02	−0.8	−0.03	0.06	−0.01
M3C2 std. error [cm]	5.85	7.51	7.52	7.54	7.5	7.52	3.41	3.67	3.61	3.69	3.65	3.66
RMSE [cm]	5.86	7.51	7.52	7.55	7.5	7.52	3.43	3.67	3.69	3.69	3.65	3.66

and large stretches of known surface may separate these sites. To minimize the total distance traveled, inter-cluster utility should then take priority over intra-cluster utility, i.e. $\lambda_{\text{tc}} \gg \lambda_{\text{ic}}$. On the other hand, the pedestal of the statue excluded, **Scenario 2** mainly consists of round surfaces and the average distance between two clusters is much smaller than in **Scenario 1**. Similar penalty terms should be then selected this time (i.e. $\lambda_{\text{tc}} \simeq \lambda_{\text{ic}}$, see Table I).

4) Single UAV: The method in [1] and the algorithms described in [3], [20], exhibit similar completion times for **Scenario 1**. The deviation is more pronounced with the Statue of Liberty: in fact, the algorithm in [20] takes twice as long to finish the exploration. However, the trajectory of the quadrotor UAV is longer (twice as much, in **Scenario 1**), and more jagged with the planner in [1]. This is not surprising, since the viewpoint configurations have been generated for accurate 3D reconstruction and not for navigation purposes as in [3]. Noisy depth measurements have a negligible effect on the trajectory of the UAV and on the completion time, but a degradation of the reconstruction and coverage quality can be observed.

5) Multi-UAV vs. single-UAV: The use of multiple UAVs has a beneficial effect on the completion time. In particular, the TSGA planner significantly reduces it, as shown in Table II. In **Scenario 1**, the fleet of three (five) UAVs performs the 3D reconstruction 68.8% (80.8%) faster than a single UAV. With **Scenario 2** we had a similar outcome, the gain on the completion time being of 72.2% (81.8%). On the other hand, the cumulated path length of the fleet is bigger than that of a single UAV in both scenarios, and the number of quadrotors has little impact on the reconstruction quality and completeness.

6) TSGA vs. NNB planner: Overall, the TSGA planner is more effective than the NNB planner for multi-robot navigation (see the comparative results reported in Table III). TSGA outperforms NNB in terms of cumulated path length in wide and large environments (**Scenario 1**), but the results are comparable in medium-size structures (**Scenario 2**). Indeed, the surface properties play an important role on the reconstruction, and a planner favoring fast local updates tends to be more successful in small environments containing close occluded areas, sharp edges and fine details, where many ISEs can be revealed after a scan. On the other hand, over a long horizon, a planner is more effective at finding the shortest path in a large planar environment where each viewpoint covers less ISEs. By comparing the completion times for **Scenario 2**, we can notice that TSGA is much faster than NNB, but that the UAVs need to travel long distances before completing the same number of ISEs. Nevertheless, NNB performs many more scans which are close to each other, and long paths are computed to complete the reconstruction (see Fig. 5 [bottom left]).

TABLE III

Comparison between the TSGA and NBB planner.

	Scenario 1		Scenario 2	
Number of UAVs	3	5	3	5
Path length gain [%]	23.9	21	1.64	1
Completion-time gain [%]	7.32	7.06	21.8	10

Our method guarantees that all the regions that are accessible to the UAVs are covered. Moreover, in keeping with the recent qualitative analysis for single-robot exploration in [14], it turns out to be competitive with the state-of-the-art approaches in terms of overall 3D reconstruction quality. In fact, the quality of 3D reconstruction is resolution-dependent: in fact, it is inversely proportional to the size of the TSDF voxels. A small resolution amounts to a large number of voxels to be integrated in the TSDF map, which is a resource-intensive process. Therefore, if multiple UAVs explore a large environment using on-board sensing and processing, a trade-off between reconstruction accuracy and computational efficiency should be found.

VI. Conclusions and future work

In this paper, we have presented a new Next-Best-View planning algorithm for online surface reconstruction of large-scale environments with a fleet of aerial robots. In particular, a novel cluster-based 3D reconstruction gain and cost-utility formulation, and a local TSP-greedy allocation planner have been proposed. Realistic numerical experiments in ROS-Gazebo, have successfully validated the proposed strategy in two challenging outdoor environments. A significant reduction of completion time has been observed with respect to the single-UAV case and a baseline multi-robot path planner (NNB).

There are several promising directions for further research we would like to explore in the future. First of all, we plan to test our method in the presence of localization uncertainty, network communication delays and data dropout, and to modify our centralized architecture (and notably our mapping module) to make it amenable to a decentralized implementation. We would also like to study the case of multiple heterogeneous robots (e.g. ground and aerial vehicles). Finally, work is in progress to validate our approach on hardware platforms in real-world environments.

References

[1] G. Hardouin, F. Morbidi, J. Moras, J. Marzat, and E. Mouaddib. Surface-driven Next-Best-View planning for exploration of large-scale 3D environments. In *Proc. 21st IFAC World Congress*, 2020, to appear.

[2] W. Tabib, M. Corah, N. Michael, and R. Whittaker. Computationally efficient information-theoretic exploration of pits and caves. In *Proc. IEEE/RSJ Int. Conf. Intel. Robots Syst.*, pages 3722–3727, 2016.

[3] S. Song and S. Jo. Surface-based Exploration for Autonomous 3D Modeling. In *Proc. IEEE Int. Conf. Robot. Automat.*, pages 4319–4326, 2018.

[4] A. Bircher, M. Kamel, K. Alexis, H. Oleynikova, and R. Siegwart. Receding horizon path planning for 3D exploration and surface inspection. *Auton. Robot.*, 42(2):291–306, 2018.

[5] C. Connolly. The Determination of Next Best Views. In *Proc. IEEE Int. Conf. Robot. Automat.*, volume 2, pages 432–435, 1985.

[6] R. Pito. A Sensor-Based Solution to the "Next Best View" Problem. In *Proc. 13th Int. Conf. Pattern Recogn.*, volume 1, pages 941–945, 1996.

[7] S. Kriegel, C. Rink, T. Bodenmüller, and M. Suppa. Efficient next-best-scan planning for autonomous 3D surface reconstruction of unknown objects. *J. Real-Time Image Pr.*, 10(4):611–631, 2015.

[8] A.P. Punnen. The Traveling Salesman Problem: Applications, Formulations and Variations. In G. Gutin and A.P. Punnen, editors, *The Traveling Salesman Problem and Its Variations*, volume 12, pages 1–28. Springer, 2007.

[9] B. Curless and M. Levoy. A volumetric method for building complex models from range images. In *Proc. 23rd Annual Conf. Computer Graph. Interact. Tech.*, pages 303–312, 1996.

[10] J.I. Vasquez-Gomez, L.E. Sucar, R. Murrieta-Cid, and E. Lopez-Damian. Volumetric Next-best-view Planning for 3D Object Reconstruction with Positioning Error. *Int. J. Adv. Robot. Syst.*, 11(10):159, 2014.

[11] R. Border, J.D. Gammell, and P. Newman. Surface Edge Explorer (SEE): Planning Next Best Views Directly from 3D Observations. In *Proc. IEEE Int. Conf. Robot. Automat.*, pages 6116–6123, 2018.

[12] L. Yoder and S. Scherer. Autonomous Exploration for Infrastructure Modeling with a Micro Aerial Vehicle. In D.S. Wettergreen and T.D. Barfoot, editors, *Field and Service Robotics: Results of the 10th Int. Conf.*, pages 427–440. Springer, 2016.

[13] R. Monica and J. Aleotti. Contour-based next-best view planning from point cloud segmentation of unknown objects. *Auton. Robot.*, 42(2):443–458, 2018.

[14] L.M. Schmid, M. Pantic, R. Khanna, L. Ott, R. Siegwart, and J. Nieto. An Efficient Sampling-based Method for Online Informative Path Planning in Unknown Environments. *IEEE Robot. Autonom. Lett.*, 5(2):1500–1507, 2020.

[15] A. Hornung, K.M. Wurm, M. Bennewitz, C. Stachniss, and W. Burgard. OctoMap: An efficient probabilistic 3D mapping framework based on octrees. *Auton. Robot.*, 34(3):189–206, 2013.

[16] A. Bircher, M. Kamel, K. Alexis, H. Oleynikova, and R. Siegwart. Receding Horizon "Next-Best-View" Planner for 3D Exploration. In *Proc. IEEE Int. Conf. Robot. Automat.*, pages 1462–1468, 2016.

[17] C. Papachristos, M. Kamel, M. Popović, S. Khattak, A. Bircher, H. Oleynikova, T. Dang, F. Mascarich, K. Alexis, and R. Siegwart. Autonomous Exploration and Inspection Path Planning for Aerial Robots Using the Robot Operating System. In A. Koubaa, editor, *Robot Operating System (ROS)*, pages 67–111. Springer, 2019.

[18] M. Selin, M. Tiger, D. Duberg, F. Heintz, and P. Jensfelt. Efficient Autonomous Exploration Planning of Large-Scale 3-D Environments. *IEEE Robot. Autonom. Lett.*, 4(2):1699–1706, 2019.

[19] S. Karaman and E. Frazzoli. Sampling-based algorithms for optimal motion planning. *Int. J. Robot. Res.*, 30(7):846–894, 2011.

[20] S. Song and S. Jo. Online Inspection Path Planning for Autonomous 3D Modeling using a Micro-Aerial Vehicle. In *Proc. IEEE Int. Conf. Robot. Automat.*, pages 6217–6224, 2017.

[21] F. Amigoni and A. Gallo. A Multi-Objective Exploration Strategy for Mobile Robots. In *Proc. IEEE Int. Conf. Robot. Automat.*, pages 3850–3855, 2005.

[22] W. Burgard, M. Moors, C. Stachniss, and F.E. Schneider. Coordinated multi-robot exploration. *IEEE Trans. Robot.*, 21(3):376–386, 2005.

[23] A. Mannucci, S. Nardi, and L. Pallottino. Autonomous 3D Exploration of Large Areas: A Cooperative Frontier-Based Approach. In *Proc. Int. Conf. Model. Simul. Auton. Systems*, pages 18–39, 2017.

[24] B. Charrow, S. Liu, V. Kumar, and N. Michael. Information-theoretic mapping using Cauchy-Schwarz Quadratic Mutual Information. In *Proc. IEEE Int. Conf. Robot. Automat.*, pages 4791–4798, 2015.

[25] G. Best, O. Cliff, T. Patten, R. Mettu, and R. Fitch. Decentralised Monte Carlo Tree Search for Active Perception. In *Int. Work. Algor. Found. Robot. (WAFR)*, 2016. Paper 50.

[26] N. Atanasov, J. Le Ny, K. Daniilidis, and G.J. Pappas. Decentralized active information acquisition: Theory and application to multi-robot SLAM. In *Proc. IEEE Int. Conf. Robot. Automat.*, pages 4775–4782, 2015.

[27] M. Corah, C. O'Meadhra, K. Goel, and N. Michael. Communication-Efficient Planning and Mapping for Multi-Robot Exploration in Large Environments. *IEEE Robot. Autonom. Lett.*, 4(2):1715–1721, 2019.

[28] M. Corah and N. Michael. Distributed matroid-constrained submodular maximization for multi-robot exploration: Theory and practice. *Auton. Robot.*, 43(2):485–501, 2019.

[29] G.M.J.-B. Chaslot. *Monte-Carlo Tree Search*. PhD thesis, Maastricht University, 2010.

[30] M.L. Fisher, G.L. Nemhauser, and L.A. Wolsey. An Analysis of Approximations for Maximizing Submodular Set Functions — II. *Math. Program. Stud.*, 8:73–87, 1978.

[31] G.L. Nemhauser, L.A. Wolsey, and M.A. Fisher. An Analysis of Approximations for Maximizing Submodular Set Functions — I. *Math. Program.*, 14:265–294, 1978.

[32] M. Kamel, T. Stastny, K. Alexis, and R. Siegwart. Model Predictive Control for Trajectory Tracking of Unmanned Aerial Vehicles Using Robot Operating System. In A. Koubaa, editor, *Robot Operating System (ROS) The Complete Reference*, volume 2, pages 3–29. Springer, 2017.

[33] C.V. Nguyen, S. Izadi, and D. Lovell. Modeling Kinect Sensor Noise for Improved 3D Reconstruction and Tracking. In *2nd IEEE Int. Conf. 3D Imaging, Model. Proc. Visual. & Transm.*, pages 524–530, 2012.

[34] H. Oleynikova, C. Lanegger, Z. Taylor, M. Pantic, A. Millane, R. Siegwart, and J. Nieto. An open-source system for vision-based micro-aerial vehicle mapping, planning, and flight in cluttered environments. *J. Field Robot.*, 37(4):642–666, 2020.

[35] J.E. Bresenham. Algorithm for computer control of a digital plotter. *IBM Syst. J.*, 4(1):25–30, 1965.

[36] C. Godsil and G. Royle. *Algebraic Graph Theory*. Springer, 2001.

[37] H. González-Banos and J.-C. Latombe. Navigation strategies for exploring indoor environments. *Int. J. Robot. Res.*, 21(10-11):829–848, 2002.

[38] K. Helsgaun. An effective implementation of the Lin–Kernighan traveling salesman heuristic. *Eur. J. Oper. Rer.*, 126(1):106–130, 2000.

[39] S.T. Jawaid and S.L. Smith. Informative path planning as a maximum traveling salesman problem with submodular rewards. *Discrete Appl. Math.*, 186:112–127, 2015.

[40] F. Furrer, M. Burri, M. Achtelik, and R. Siegwart. RotorS – A Modular Gazebo MAV Simulator Framework. In A. Koubaa, editor, *Robot Operating System (ROS): The Complete Reference*, volume 1, pages 595–625. Springer, 2016.

[41] L. Keselman, J.I. Woodfill, A. Grunnet-Jepsen, and A. Bhowmik. Intel RealSense Stereoscopic Depth Cameras. In *Proc. IEEE Conf. Comp. Vis. Pattern Recogn. Workshops*, pages 1–10, 2017.

[42] A. Zeng, S. Song, M. Nießner, M. Fisher, J. Xiao, and T. Funkhouser. 3DMatch: Learning Local Geometric Descriptors from RGB-D Reconstructions. In *Proc. IEEE Conf. Comp. Vis. Pattern Recogn.*, pages 1802–1811, 2017.

[43] W.E. Lorensen and H.E. Cline. Marching cubes: A high resolution 3D surface construction algorithm. In *Proc. ACM SIGGRAPH Comp. Graph.*, volume 21, pages 163–169, 1987.

[44] I.A. Şucan, M. Moll, and L.E. Kavraki. The Open Motion Planning Library. *IEEE Rob. Autom. Mag.*, 19(4):72–82, 2012.

[45] D. Lague, N. Brodu, and J. Leroux. Accurate 3D comparison of complex topography with terrestrial laser scanner: Application to the Rangitikei canyon (N-Z). *ISPRS J. photogramm.*, 82:10–26, 2013.