Preprints of the 21st IFAC World Congress (Virtual)
Berlin, Germany, July 12-17, 2020

Surface-driven Next-Best-View planning for exploration of large-scale 3D environments

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

[1] *ONERA-DTIS, Université Paris-Saclay, Palaiseau, France*
Email: firstname.surname@onera.fr
[2] *MIS laboratory, Université de Picardie Jules Verne, Amiens, France*
Email: {fabio.morbidi, mouaddib}@u-picardie.fr

Abstract: In this paper, we propose a novel cluster-based informative path planning algorithm to simultaneously explore and inspect a *large-scale unknown* environment with an Unmanned Aerial Vehicle (UAV). Most of the existing methods address the surface inspection problem as a volume exploration problem, and consider that the surface has been scanned when the corresponding volume has been covered. Unfortunately, this approach may lead to inaccurate 3D models of the environment, and the UAV may not achieve global coverage. To overcome these critical limitations, we introduce a 3D reconstruction method based on TSDF (Truncated Signed Distance Function) mapping, which leverages the surfaces present in the environment to generate an informative exploration path for the UAV. A Probabilistic Roadmap planner, used to solve a TSP (Travelling Salesman Problem) over clusters of viewpoint configurations, ensures that the resulting 3D model is accurate and complete. Two challenging structures (a power plant and the Statue of Liberty) have been chosen to conduct realistic numerical experiments with a quadrotor UAV. Our results provide evidence that the proposed method is effective and robust.

Keywords: Next-Best-View planning, 3D reconstruction, Truncated Signed Distance Function, Probabilistic Roadmap, Unmanned Aerial Vehicle (UAV).

1. INTRODUCTION

Autonomous robots are being increasingly employed today for time-consuming and dangerous tasks usually performed by human operators. For instance, aerial robots with different on-board sensors (RGB-D and 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 (Tabib et al. (2016); Song and Jo (2018); Bircher et al. (2018)). In Next-Best-View (NBV) planning (Connolly (1985)), the robot iteratively computes the best viewpoint configurations to fully reconstruct the 3D environment, which can be (partially) known in advance or completely unknown. The robot then progressively discovers the surrounding environment during its 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.

In this paper, we study the problem of incremental exploration and surface reconstruction of an unknown 3D environment for inspection purposes, with an Unmanned Aerial Vehicle (UAV). To address this problem, a map of the environment is required for collision avoidance and to verify if the reconstruction is complete. In this work, a map representation based on the Truncated Signed Distance Function (TSDF) is considered. The surface of the obstacles is estimated by incrementally fusing the range measurements into a volumetric distance field that represents the signed distance to the closest surface. This field can then be transformed into a surface representation. In Newcombe et al. (2011); Klingensmith et al. (2015), the authors have shown that an optimized implementation of TSDF mapping, running on an embedded computer, can provide an accurate mesh reconstruction in real-time. Furthermore, TSDF data can be useful to identify missing parts of the model during the online reconstruction phase, as in Monica and Aleotti (2018). Thanks to the volumetric representation, it is possible to generate a list of candidate sensor configurations from the missing parts in the identified free space. We propose an NBV planning method which allows to visit all these configurations and which solves a Travelling Salesman Problem (TSP) based on the known map: an optimal path visiting the viewpoint configurations is computed by taking the surface coverage they provide and their location in space, into account. As the robot follows the path, it performs successive scans and completes the map. The initial path is updated by leveraging the gathered information, until the whole unknown surface is covered.

In summary, the original contributions of this work are threefold:

- We only use a TSDF representation of surfaces to measure the quality of reconstruction and to identify the unknown or incomplete areas from which new

viewpoint configurations for the UAV, can be generated.

- We propose a novel objective function for NBV planning, which incorporates the TDSF-based information gain at each robot configuration.
- We design a new algorithm, based on the computation of successive (approximate) solutions to the TSP in conjunction with a Probabilistic Roadmap planner (Lazy PRM*, Hauser (2015)), to visit the clusters of viewpoint configurations for surface reconstruction.

The remainder of this paper is organized as follows. In Section 2, we review the state of the art in autonomous inspection and exploration of unknown environments. Section 3 is devoted to the problem formulation. The proposed algorithm is described in Section 4. A detailed evaluation of the algorithm in realistic simulation environments is reported in Section 5. Finally, in Section 6, the main contributions of the paper are summarized and some possible avenues for future research are outlined.

2. RELATED WORK

The ability to plan informative paths for online exploration and modeling of 3D environments, is an essential prerequisite for truly autonomous robots. Numerous strategies for 3D modeling of an unknown environment with a sensor-equipped robotic platform exist in the literature. In this paper, we are specially concerned with NBV methods, which determine the best viewpoints to visit depending on the environment and the nature of the robot mission.

Surface inspection methods analyze the reconstructed surface for viewpoint definition. The goal is to ensure that the reconstruction is accurate and complete. *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 guarantee continuity (Connolly (1985); Vasquez-Gomez et al. (2014); Border et al. (2018)). However, the information gain associated to each viewpoint cannot be predicted to evaluate the NBV. Some works have addressed this issue by estimating the unknown surface for viewpoint selection (Pito (1996); Kriegel et al. (2015)), but generally, only small objects are reconstructed. In addition, these works rely on strong assumptions on the location of the objects in space, and the only use of a surface representation makes it difficult to extend these approaches to robotic applications, since the free space cannot be determined. The authors in Vasquez-Gomez et al. (2014) were the first to consider a volumetric representation for frontier-based object reconstruction. This approach has been used in Yoder and Scherer (2016) for a large-scale environment, but the quality of the 3D reconstruction obtained with a UAV has not been assessed. Recently, in Monica and Aleotti (2018), new poses are generated by directly leveraging the properties of the TSDF. Finally, in Schmid et al. (2020), the authors exploited the TSDF representation for surface reconstruction and the Euclidean SDF (Oleynikova et al. (2017)) for navigation, and they proposed a path-generation strategy based on the volumetric properties of the TSDF and on the RRT* expansion Karaman and Frazzoli (2011). They also compared the reconstruction quality/completeness of their method with that of some recent volumetric approaches.

Volumetric exploration aims at exploring and at building a map of a large unknown environment, by using a 3D grid model (such as Octomap Hornung et al. (2013)) to identify known, unknown or occupied areas. The majority of recent exploration methods are *sampling-based* Bircher et al. (2016); Papachristos et al. (2019). They expand a random tree (RRT/RRT*) of sampled sensor configurations in the free space, and keep the NBV trajectory that guarantees the best volume coverage. Pose coverage is evaluated via ray tracing, and the volume is explored as the UAV moves along the assigned path. These methods, which rely on a coarse volumetric map of the environment for navigation purposes, are fast and efficient, but they do not explicitly account for the accuracy of the reconstructed surface. Some recent works have tried to combine volumetric exploration (Octomap) and surface inspection for large-scale 3D reconstruction. In Bircher et al. (2018), a preliminary exploration of the unknown volume is followed by an inspection phase. The path that ensures the maximum coverage of the surface is selected, and the inspection stops when the resolution of the facets of the reconstructed mesh is optimal. An algorithm based on the RRT* expansion is proposed in Song and Jo (2017) for the generation of an exploration path, where the nodes are sampled poses. A minimal number of viewpoint configurations is chosen, which guarantees the coverage of surface cells and the local shortest path for the aerial robot. However, the quality of the obtained 3D model is not evaluated. In Song and Jo (2018), the same authors improved their previous method by considering the incomplete areas of the 3D model, in their sampling-based pose generation algorithm. The volume coverage is evaluated via an Octomap, and the surface model is built from the TSDF volume which is used, in turn, to generate a point cloud. Such a point cloud is exploited on the fly to determine the incomplete areas that the robot should visit next.

Our literature review shows that in the existing methods for simultaneous exploration and inspection of an unknown environment, volume exploration has received by far the most attention, while surface inspection has been only addressed indirectly. In this paper, we push the envelope and explicitly solve the *surface inspection problem* by considering a frontier-based approach coupled with an NBV planning strategy for guidance in large-scale environments. Viewpoint configurations are clustered according to their location in space, to evaluate how the visit of specific areas is informative and find a suitable path. This search is formalized as a TSP which is successively solved with a heuristic algorithm until the reconstruction is completed. Moreover, instead of expanding a random tree, we use the Lazy PRM* planner (Hauser (2015)), to find the shortest path between two configurations by taking the map of the environment into account. This planner is faster than RRT, and it is multi-query, making it ideal for path checking between clusters, and thus for the resolution of the TSP. Realistic numerical experiments featuring two large-scale structures (an industrial plant and a statue) and a quadrotor UAV, show the effectiveness of our approach.

3. PROBLEM FORMULATION

In this paper, we consider a UAV with 4 degrees of freedom (the position $[x, y, z]^T \in \mathbb{R}^3$ and the yaw angle $\psi \in \mathcal{S}^1$), equipped with a depth sensor, e.g. a stereo camera or a RGB-D sensor. The aerial robot should scan an unknown but spatially-bounded environment $V \subset \mathbb{R}^3$, and it should accurately reconstruct the surface that it contains. Its 3D exploration path should also be as short as possible.

We denote by Q the set of all collision-free configurations $\mathbf{q} = [x, y, z, \psi]^T$ of the UAV, which are assumed to be exactly known. The vector-valued function $\mathbf{p}_i^j(s) : [0, 1] \to \mathbb{R}^3 \times \mathcal{S}^1$ defines the path from configuration i to configuration j, where $\mathbf{p}_i^j(0) = \mathbf{q}_i$ and $\mathbf{p}_i^j(1) = \mathbf{q}_j$. We assume that $\mathbf{p}_i^j(s)$ is collision-free and feasible for the aerial robot (i.e. the kinematic/dynamic constraints of the aerial vehicle are satisfied along the path). A localisation system provides the (exact) pose of the UAV with respect to a world reference frame.

The forward-facing depth sensor is rigidly attached to the aerial robot, and its pose calibrated with respect to the body frame of the UAV. The sensor provides a depth map and it has a limited field of view (FOV) and sensing range. The depth maps are incrementally 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 $w \geq 0$. Depending on the values of ϕ and w, one can determine if a voxel is *unknown*, *occupied* or *empty*. In fact, voxel $\mathbf{v} \in M$ is,

$$\begin{cases} \text{unknown} & \text{if } w(\mathbf{v}) = 0, \\ \text{occupied} & \text{if } w(\mathbf{v}) > 0 \wedge \phi(\mathbf{v}) \leq 0, \\ \text{empty} & \text{if } w(\mathbf{v}) > 0 \wedge \phi(\mathbf{v}) > 0, \end{cases}$$

where $w(\mathbf{v})$ and $\phi(\mathbf{v})$ are the weight and signed distance of voxel $\mathbf{v}$, respectively. Note that $\phi(\mathbf{v}) = 0$ implicitly defines a surface: hence, the TSDF volume is a volumetric representation of a surface. Every time a voxel is scanned, its associated weight is incremented by $1/z_c^2(\mathbf{v})$ to minimize the sensing error, where $z_c(\mathbf{v})$ is the distance between voxel $\mathbf{v}$ and the current sensor position. To filter out noise and reduce the number of false-positive voxels, we consider voxel $\mathbf{v}$ to be known if its weight is greater than or equal to a threshold W_{th}, i.e if $w(\mathbf{v}) \geq W_{\text{th}} > 0$. The value of W_{th} depends on the sensing range of the depth sensor, and it can be determined experimentally. In Monica and Aleotti (2018), the authors defined a *contour* as the set of empty voxels that are neighbors to both an occupied and an unknown voxel. More precisely, voxel $\mathbf{v} \in M$ is said to belong to a contour if the following three conditions are fulfilled:

$$\begin{aligned} &a)\ w(\mathbf{v}) \geq W_{\text{th}} \wedge \phi(\mathbf{v}) > 0, \\ &b)\ \exists \mathbf{r} \in N_{\mathbf{v}}^6 \text{ s.t. } w(\mathbf{r}) < W_{\text{th}}, \\ &c)\ \exists \mathbf{o} \in N_{\mathbf{v}}^{18} \text{ s.t. } w(\mathbf{o}) \geq W_{\text{th}} \wedge \phi(\mathbf{o}) \leq 0, \end{aligned} \tag{1}$$

where $N_{\mathbf{v}}^6$ and $N_{\mathbf{v}}^{18}$ denote the 6- and 18-connected voxel neighborhoods of $\mathbf{v}$, respectively. In fact, if one observes the space around the occupied voxels next to a frontier, the perception of the corresponding surface of the object improves.

The following three definitions are introduced to support our problem statement.

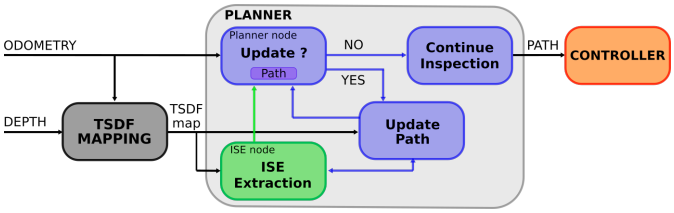

Figure 1. *General flowchart of our algorithm*: the internal architecture of the planner is shown inside the shaded box.

Definition 1 (**Incomplete surface element**). *A voxel* $\mathbf{v} \in M$ *which satisfies* (1), *represents an* Incomplete Surface Element (ISE). *We denote by* $C \subset M$ *the set of all ISEs.*

Definition 2 (**Remaining incomplete surface**). *Let* $Q_c \subseteq Q$ *be the set of all configurations* $\mathbf{q}$ *from which voxel* $\mathbf{v} \in C$ *can be completed. The* remaining incomplete surface *is then defined as* $C_{\text{rem}} = \bigcup_{\mathbf{v} \in C} \{\mathbf{v} \mid Q_c = \emptyset\}$.

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

Problem 1 (**UAV exploration & inspection**). *Find a collision-free path* $\mathbf{p}_0^n$ *between the initial configuration* $\mathbf{q}_0$ *and the final configuration* $\mathbf{q}_n$, *which allows the UAV to scan the set* $C_{\text{ins}} = C \backslash C_{\text{rem}}$ *of all ISEs.*

4. PROPOSED APPROACH

During the exploration, the map is incrementally built and the UAV detects sets of incomplete surfaces or contours (cf. Section 3). The configurations which allow to complete them are generated and clustered, depending on their location in the 3D space. A directed graph is created and continuously expanded to represent the travel utility between clusters in the free space. A path ensuring collision-free navigation is extracted from this graph to maximize the utility. As the map evolves, new ISEs are discovered, and the path to complete them is updated. The 3D model is considered complete when there are no more ISEs.

As depicted in Fig. 1, the planner includes two modules. The first module is the ISE extractor which identifies incomplete areas to visit from the TSDF map, generates candidate viewpoint configurations, and clusters them. The second module is the graph-based planner, which finds the optimal path for 3D reconstruction. It relies on a single objective function, which trades off travel cost and utility of visiting a new cluster. The path is updated if obstacles are detected along the way or if the area which still needs to be scanned has been already completed. The two modules are described in more detail in the next two sections.

4.1 ISE extractor

Viewpoint generation from an incomplete element. Following (Monica and Aleotti, 2018, Sect. 3.1), we adopt a fast local approach to determine a direction $\mathbf{n_v}$ for observing the ISE $\mathbf{v}$. This approach is based on the gradient of the weight function $\nabla w(x, y, z)$, which can be computed as,

$$\mathbf{n_v} = \sum_{\mathbf{c} \in N_{\mathbf{v}}^{26}} w'(\mathbf{c}) \frac{\mathbf{c} - \mathbf{v}}{\|\mathbf{c} - \mathbf{v}\|}, \tag{2}$$

where $N_{\mathbf{v}}^{26}$ is the 26-connected neighborhood of $\mathbf{v}$ (in order to minimize noise and sampling effects). We propose the following variant of the "modified" weight function w',

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

Note that $\mathbf{n_v}$ in (2) points away from the known surface (see Fig. 2), instead of being orthogonal to it, as in a classical frontier-based approach. Such a direction guarantees a better surface coverage when the incomplete areas are located on sharp edges. A new sensor configuration is generated along the direction $\mathbf{n_v}$ at a specified distance δ_{pose} from voxel $\mathbf{v}$. The sensor points towards the incomplete element: in this way, new informative cues about the surface can be extracted, while still guaranteeing that the previously-explored regions are visible. Parameter δ_{pose} can be selected according to the technical specifications of the depth sensor (e.g., range, FOV, etc., see Chen et al. (2008)). Unfortunately, the poses $\mathbf{q}_i, \mathbf{q}_j \in Q$ generated from the ISEs $\mathbf{v}_i, \mathbf{v}_j \in C$, respectively, may occasionally be very close, i.e. $\text{dist}(\mathbf{q}_i, \mathbf{q}_j) < \epsilon$ for a small $\epsilon > 0$, or the viewing directions are almost parallel, i.e. $|\mathbf{n}_{\mathbf{v}_i} \cdot \mathbf{n}_{\mathbf{v}_j}| \simeq 1$. These configurations are then aggregated into a single viewpoint by averaging their positions and orientations.

Cluster of configurations. Large-scale environments may result in a number of possible viewpoint configurations, which is impractical for planning purposes. To overcome this problem and easily identify those areas of the surface which are more promising, 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. We say that configuration $\mathbf{q}_i$ belongs to a generic cluster u if $\exists\, \mathbf{q}_j \in u$ s.t. $d(\tau_i^j) < d_\nu$, where $d(\tau_i^j)$ denotes the length of the path τ_i^j between $\mathbf{q}_i$ and $\mathbf{q}_j$ on the directed graph we will define in Section 4.2, and d_ν is an upper bound on the distance. If no neighbors are found, d_ν is increased up to the maximum value of $d_\nu^{\max}$.

Cluster evaluation. For a given path to follow, and depending on the current TSDF map and surface shape, some viewpoint configurations will be more "useful" than others. A possible way to determine the priority of a viewpoint, is to measure its level of informativeness. For instance, in the classical *ray-tracing method* proposed

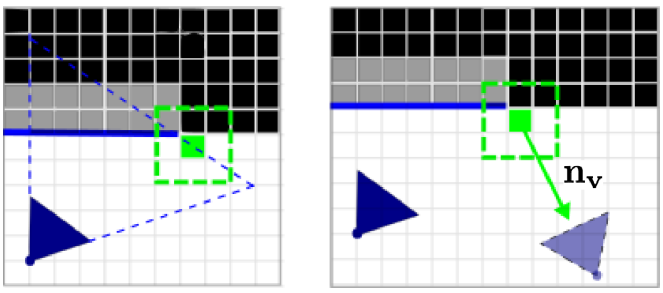

Figure 2. [left] Two-dimensional example of voxel contour $\mathbf{v}$ (filled green square). Its 2D neighborhood is represented by a dashed green square. Unknown voxels are black, occupied voxels are gray, and empty voxels are white. The reconstructed surface is depicted as a blue segment, and the sensor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

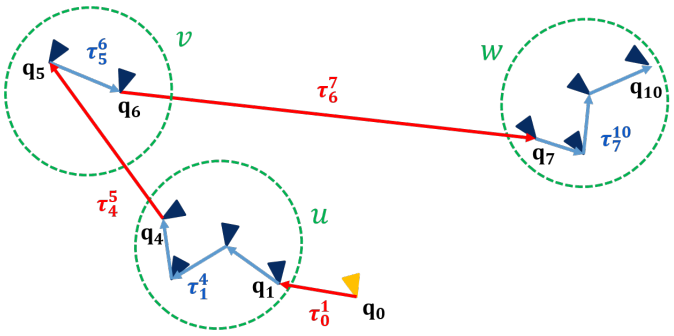

Figure 3. Example of weighted directed graph $\mathcal{G}$ with three clusters.

in Bresenham (1965), a set of rays is traced inside the voxel map located at the viewpoint sensor frustum. When a ray crosses a voxel, the attributes of this voxel are stored, e.g. known or unknown, free or occupied. This approach is often used in volume-exploration tasks, to obtain the information gain of each viewpoint. To evaluate a viewpoint configuration, we use the ray-tracing method from a frontier-based perspective, i.e. we simply count the number of ISEs that can be seen from that location. More formally, let $C_{\mathbf{q}}$ be the set of all ISEs seen from viewpoint $\mathbf{q}$ and let $C_u = \bigcup_{\mathbf{q} \in u} C_{\mathbf{q}}$. We then define the *gain* $g(u)$ of cluster u as,

$$g(u) = \frac{|C_u|}{|C|}, \tag{3}$$

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

4.2 Graph-based path planner

The planner schedules the visit of clusters of a given TSDF map. The associated TSP is solved and the path which maximizes an utility function is calculated. The utility of visiting a cluster depends on its gain and on the cost of reaching all its viewpoint configurations. The cost corresponds to the length of the feasible paths which connect configurations computed with a PRM planner.

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 a collection of weights for the edges E (see Fig. 3). 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 the UAV belongs to one of the clusters of $\mathcal{G}$, i.e. $\mathbf{q}_0 \in U$. Let $\mathbf{q}_k$ be a configuration of cluster u, and $\mathbf{q}_l$, $\mathbf{q}_m$ two configurations of cluster v. Then, the weight a_{uv} between cluster u and v is the 6-tuple defined as,

$$a_{uv} = \{\tau_k^l, \tau_l^m, g(v), d(\tau_k^l), d(\tau_l^m), f_{uv}\}, \tag{4}$$

where

- τ_k^l denotes the path from $\mathbf{q}_k \in u$ to $\mathbf{q}_l \in v$, i.e. the path between cluster u and v. In particular, we select the configuration $\mathbf{q}_l$ in v, which guarantees that τ_k^l is the shortest possible path,
- τ_l^m denotes the shortest Hamiltonian path (Godsil and Royle (2001)) including configurations in v, which starts at $\mathbf{q}_l$ and ends at $\mathbf{q}_m$,
- $g(v)$ is the gain of the cluster v, as defined in (3),
- $d(\tau_k^l)$ is the cost associated to the *inter-cluster* path τ_k^l (the length of τ_k^l),

- $d(\tau_l^m)$ is the cost associated to the *intra-cluster* path τ_l^m (the length of τ_l^m),
- f_{uv} is the *utility function* defined as,

$$f_{uv} = g(v)\exp\big(-\lambda_{tc}\, d(\tau_k^l) - \lambda_{ic}\, d(\tau_l^m)\big)\,, \quad (5)$$

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.

Note that the utility function (5), was originally proposed in González-Banos and Latombe (2002), and it has been adapted here to our new formulation. The information gain provided by a cluster is penalized by the distance to reach it and the travel cost associated to the visit of all configurations belonging to it.
The weights (4) of the directed graph $\mathcal{G}$, quantify the potential benefit of taking a path to pursue the 3D reconstruction (see Fig. 3). The goal is to maximize the total utility function. Note that since $\mathcal{G}$ is directed, we should solve an Asymmetric Travelling Salesman Problem (Punnen (2007)). After converting it to a symmetric one (i.e. to a standard TSP), the Hamiltonian path is found using the Lin-Kernighan heuristic (Helsgaun (2000)).
Successive scans along the path allow to generate new viewpoint configurations, and the information gain associated to the visit of clusters changes as the map grows. The path is locally updated if new areas are discovered as the TSDF map is completed, or if a cluster has already covered. The path is fully recomputed if the N_c clusters have been visited (cf. Fig. 1). The reconstruction process stops when no more ISEs remain.

5. NUMERICAL EXPERIMENTS

The proposed surface-driven method has been validated via realistic numerical experiments. We chose a benchmark industrial plant, which is widely used in the volumetric exploration literature (Scenario 1). In order to study the impact of the penalty terms in (5), on the reconstruction accuracy/completeness, we also considered a monumental statue (Scenario 2). The simulation parameters used in the two scenarios are reported in Table 1. To simulate the aerial vehicle, we used the ROS-Gazebo [1] environment. For the quadrotor UAV with an on-board stereo camera, we leveraged the model provided by the RotorS simulator (Furrer et al. (2016)). The TSDF volume was generated with the algorithm proposed in Zeng et al. (2017), where the reconstruction is performed with MarchingCubes (Lorensen and Cline (1987)) and the weight increment has been modified as detailed in Section 3.

Table 1: Parameters used in the numerical experiments.

Parameter	**Scenario 1**	**Scenario 2**
Voxel resolution $\rho_{\mathbf{v}}$ [m]	0.3	0.15
Threshold W_{th}	0.3	0.3
Camera range [m]	[1.6, 8]	[1, 5]
Camera FOV [deg] (H, V)	90 × 60	90 × 60
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

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

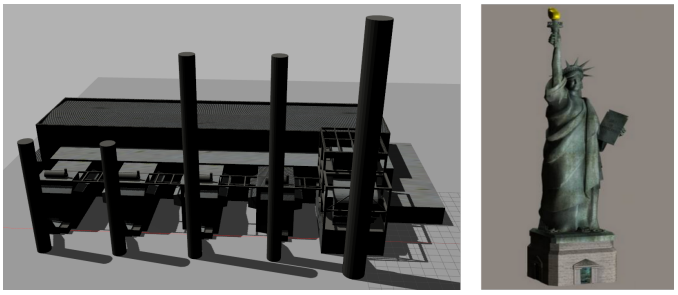

Figure 4. *Gazebo simulation environments*: [left] **Scenario 1**, Powerplant; [right] **Scenario 2**, the Statue of Liberty.

The path of the UAV is computed with the Lazy PRM* planner from the Open Motion Planning Library (Şucan et al. (2012)), which finds the shortest path between two configurations by taking the structure of the TSDF map into account (the collision radius is 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 with respect to RRT (the average runtime is below 1 s). Model Predictive Control (Kamel et al. (2017)) has been used to track the generated paths, the reference translational velocity being fixed at 0.5 m/s.
The virtual environments considered in Scenario 1 and 2 are shown in Fig. 4. In Scenario 1, the Powerplant model [2] has been scaled to fit in a box of size $65 \times 42 \times 15\,\text{m}^3$ (as a result, the five flues have the same height). 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] ($20\times20\times60\,\text{m}^3$), which contains multiple sharp edges and fine details (diadem and gown). The results of the numerical experiments are reported in Table 2. To obtain statistically-significant values, 10 trials per scenario have been carried out. As in the exploration methods (cf. Section 2), we evaluated the total length of the path of the UAV, and the completion time, i.e. the time necessary to cover the 3D environments. We ran our algorithm on a Dell Precision 7520 with 2.90 GHz Intel Core i7 processor, 16 GB RAM and Quadro M2200 graphics card. The reconstructed 3D surface has been evaluated with CloudCompare [4] using the M3C2 (Multiscale Model to Model Cloud Comparison) algorithm (Lague et al. (2013)). To quantify how well the surface has been recovered, we sampled dense point clouds on the reconstructed and ground truth (GT) meshes, and we measured the deviation by performing a cloud-to-cloud comparison (see Fig. 5). For a fair evaluation, we pruned all the invisible surfaces of the GT mesh beforehand (e.g. the interior floor and walls), and we restricted our analysis to the exterior surface mesh only. We consider that a point belonging to the GT point cloud has been covered by a corresponding one in the reconstructed cloud, if their absolute distance is less than the length of the half diagonal of a voxel, i.e. less than $e_{\max} = \rho_{\mathbf{v}}\sqrt{3}/2$, where $\rho_{\mathbf{v}}$ is the voxel resolution. Note that $2\,e_{\max}$ is the maximum reconstruction error provided by the MarchingCubes algorithm by default (see, Lorensen and Cline (1987)). The quality of

[2] http://models.gazebosim.org/
[3] https://free3D.com/
[4] https://www.danielgm.net/cc/

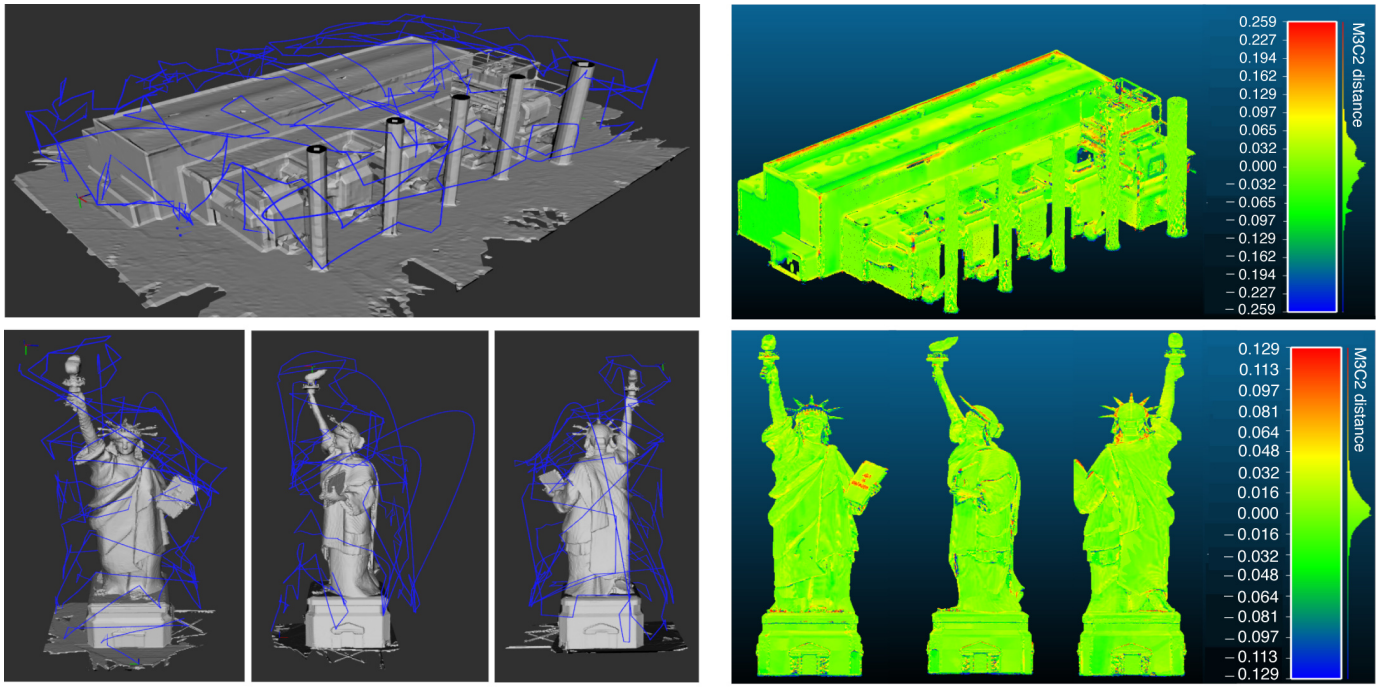

Figure 5. [top] **Scenario 1** and [bottom] **Scenario 2**: [left] Reconstructed mesh and 3D exploration path $\mathbf{p}_0^n$ (blue) of the quadrotor UAV; [right] Signed distance error. The colour coding shows the error in meters with respect to the ground truth, computed with CloudCompare's M3C2 algorithm.

the recovered surface is evaluated in Table 2, 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).

The choice of the penalty terms λ_{tc} and λ_{ic} appearing in the utility function (5), depends on the nature of the 3D environment where the UAV evolves. Scenarios 1 and 2 are, in this respect, quite representative. In wide box-like environments as Scenario 1, the ISEs tend to appear in the proximity of occluded regions and sharp edges, and large extents 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. $\lambda_{tc} \gg \lambda_{ic}$. On the other hand, the pedestal of the statue excluded, Scenario 2 predominantly consists of round surfaces and the average distance between two clusters is much smaller than in Scenario 1. As a consequence, similar penalty terms should be selected this time (i.e. $\lambda_{tc} \simeq \lambda_{ic}$, see Table 1). The upper bound $d_\nu^{\max}$ on the distance between cluster configurations d_ν, changes during the reconstruction, and its default value has been determined empirically by considering the spatial distribution of viewpoint configurations. The algorithm by Song and Jo (2018) exhibits similar completion times (around 35 min.) to ours, for Scenario 1. The gap is more important in Scenario 2: in fact, our algorithm took 36 min., while that of Song and Jo (2017), around 53 min. However, in our case, the trajectory of the quadrotor UAV is longer (780 m vs. 324 m, in Scenario 1), and more jagged. This is not surprising, since the viewpoint configurations have been generated for accurate 3D reconstruction and not for navigation purposes as in Song and Jo (2018). Moreover, no trajectory refinement (e.g. smoothing) is performed (to that effect, we plan to use a receding-horizon formulation in future works). Nevertheless, our method guarantees that all the regions that are accessible to the UAV are covered. In addition, in keeping with the recent analysis in Schmid et al. (2020), it turns out to be competitive with the state-of-the-art approaches in terms of overall 3D reconstruction quality (Schmid et al. (2020) report an RMSE of 6.4 ± 0.8 cm). However, further work is needed to perform a comparative study under identical simulation conditions. It is finally worth pointing out here, that the quality of 3D reconstruction is resolution dependent: in fact, it is inversely proportional to the size of 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 the quadrotor UAV explores a large-scale environment using only on-board sensing and processing, a trade-off between reconstruction quality and computational efficiency should be found.

Table 2: Numerical results (averages over 10 trials).

Criterion	**Scenario 1**	**Scenario 2**
Path length [m]	780	547
Completion time [min.]	32	36
$e_{\max}$ [cm]	25.98	12.99
Surface coverage [%]	91.5	92.3
M3C2 Avg. error [cm]	0.14	0.29
M3C2 Std. dev. error [cm]	5.85	3.41
RMSE [cm]	5.86	3.43

6. CONCLUSIONS AND FUTURE WORK

In this paper, we have presented a new surface-driven Next-Best-View planning algorithm for the exploration and inspection of large-scale environments with a UAV. In particular, a novel cluster-based 3D reconstruction gain and cost-utility formulation has been proposed. Realistic

numerical experiments with ROS and Gazebo have successfully validated the proposed method on two challenging outdoor environments.

There are several promising directions for further research we would like to explore in the future. Before implementing our method in real-time on a hardware platform (a quadrotor UAV), extensive numerical experiments will be carried out in the presence of noisy measurements and localization uncertainty. The full-attitude control of the aerial robot along a computed path, is another subject of ongoing research. Finally, we are planning to adapt our approach to a multi-robot cooperative setting (cf. Corah and Michael (2019)), in order to speed up simultaneous exploration and 3D reconstruction.

ACKNOWLEDGMENT

This work has been supported by ONERA DTIS and by the Hauts-de-France region, through the research project ScanBot, "Scanners Robotisés pour la Numérisation Automatique du Patrimoine" (2018-2021).

REFERENCES

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

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

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

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

Chen, S., Li, Y.F., Wang, W., and Zhang, J. (2008). *Active Sensor Planning for Multiview Vision Tasks*, volume 1. Springer.

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

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

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

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

Godsil, C. and Royle, G. (2001). *Algebraic Graph Theory*. Springer.

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

Hauser, K. (2015). Lazy Collision Checking in Asymptotically-Optimal Motion Planning. In *Proc. IEEE Int. Conf. Robot. Automat.*, 2951–2957.

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

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

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

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

Klingensmith, M., Dryanovski, I., Srinivasa, S., and Xiao, J. (2015). Chisel: Real Time Large Scale 3D Reconstruction Onboard a Mobile Device using Spatially Hashed Signed Distance Fields. In *Proc. Robotics: Science and Systems*, volume 4.

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

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

Lorensen, W.E. and Cline, H.E. (1987). Marching cubes: A high resolution 3D surface construction algorithm. In *Proc. ACM SIGGRAPH Computer Graphics*, volume 21, 163–169.

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

Newcombe, R.A., Izadi, S., Hilliges, O., Molyneaux, D., Kim, D., Davison, A.J., Kohli, P., Shotton, J., Hodges, S., and Fitzgibbon, A.W. (2011). KinectFusion: Real-time dense surface mapping and tracking. In *Proc. 10th IEEE Int. Symp. Mixed Augmen. Reality*, volume 11, 127–136.

Oleynikova, H., Taylor, Z., Fehr, M., Siegwart, R., and Nieto, J. (2017). Voxblox: Incremental 3D Euclidean Signed Distance Fields for on-board MAV planning. In *Proc. IEEE/RSJ Int. Conf. Intel. Robots Syst.*, 1366–1373.

Papachristos, C., Kamel, M., Popović, M., Khattak, S., Bircher, A., Oleynikova, H., Dang, T., Mascarich, F., Alexis, K., and Siegwart, R. (2019). Autonomous Exploration and Inspection Path Planning for Aerial Robots Using the Robot Operating System. In A. Koubaa (ed.), *Robot Operating System (ROS)*, volume 3, 67–111. Springer.

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

Punnen, A.P. (2007). The Traveling Salesman Problem: Applications, Formulations and Variations. In G. Gutin and A.P. Punnen (eds.), *The Traveling Salesman Problem and Its Variations*, volume 12, 1–28. Springer.

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

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

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

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

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

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

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