CrossMark

# Receding horizon path planning for 3D exploration and surface inspection

Andreas Bircher[1] · Mina Kamel[1] · Kostas Alexis[2] · Helen Oleynikova[1] · Roland Siegwart[1]

**Abstract** Within this paper a new path planning algorithm for autonomous robotic exploration and inspection is presented. The proposed method plans online in a receding horizon fashion by sampling possible future configurations in a geometric random tree. The choice of the objective function enables the planning for either the exploration of unknown volume or inspection of a given surface manifold in both known and unknown volume. Application to rotorcraft Micro Aerial Vehicles is presented, although planning for other types of robotic platforms is possible, even in the absence of a boundary value solver and subject to nonholonomic constraints. Furthermore, the method allows the integration of a wide variety of sensor models. The presented analysis of computational complexity and thorough simulations-based evaluation indicate good scaling properties with respect to the scenario complexity. Feasibility and practical applicability are demonstrated in real-life experimental test cases with full on-board computation.

This is one of several papers published in *Autonomous Robots* comprising the Special Issue on Active Perception.

✉ Andreas Bircher
andreas.bircher@mavt.ethz.ch; andreas.bircher@wingtra.com

Mina Kamel
mina.kamel@mavt.ethz.ch

Kostas Alexis
kalexis@unr.edu

Helen Oleynikova
elena.oleynikova@mavt.ethz.ch

Roland Siegwart
rsiegwart@ethz.ch

[1] ETH Zurich, Leonhardstrasse 21, Zurich, Switzerland

[2] University of Nevada, Reno, 1664 N. Virginia Street, Reno, NV 89557, USA

## 1 Introduction

Ever since sensors could be moved in an automated manner such that they can map their environment, the question of what is the best way to do so has raised the interest of the research community. Among the different mobile sensing problems, those of automated robotic exploration and inspection are of particular relevance, as critical application fields such as infrastructure monitoring and maintenance, as well as advanced manufacturing have high demands for automation, system reliability, predictive fault detection in order to maximized safety and lower operation cost. The wide range of possible applications asks for versatile sensor planning approaches that can handle complex and large scale problem setups. However, one of the major challenges for such algorithms is the inherent trend for increased complexity when the size of the considered scenarios grows.

Early work in the problem of sensor planning focused on scanning tasks utilizing sensors linked to constructions around the object of interest (e.g. with articulated robots). However, the development of advanced mobile robots provoked many studies on advanced environment mapping exploiting the mobility of such vehicles. While the former is particularly focused in applications such as quality testing in automated manufacturing (Chin and Harlow 1982 and references therein), the latter concepts can be applied in industrial inspection (Burri et al. 2012; Nikolic et al. 2013; Omari et al. 2014), landscape surveillance (Khanna et al. 2015), disaster zone exploration (Colas et al. 2013; Dornhege and Kleiner 2013, mine mapping (Zlot and Bosse 2014) and other similar application tasks including one or more robots (Zlot and

Springer

**Fig. 1** An instant of the exploration of a room containing a scaffold is shown on the *left side*. In the upper part one of the employed hexacopter micro aerial vehicle (MAV), the AscTec Firefly is depicted, while the online dense reconstruction acquired during the exploration mission is shown below. The computed path is visualized in *black*, closely followed by the MAV track in *light blue*. The *right side* shows the AscTec Neo MAV during an inspection mission, with the postprocessed data as a triangular mesh depicted in the upper part (Color figure online)
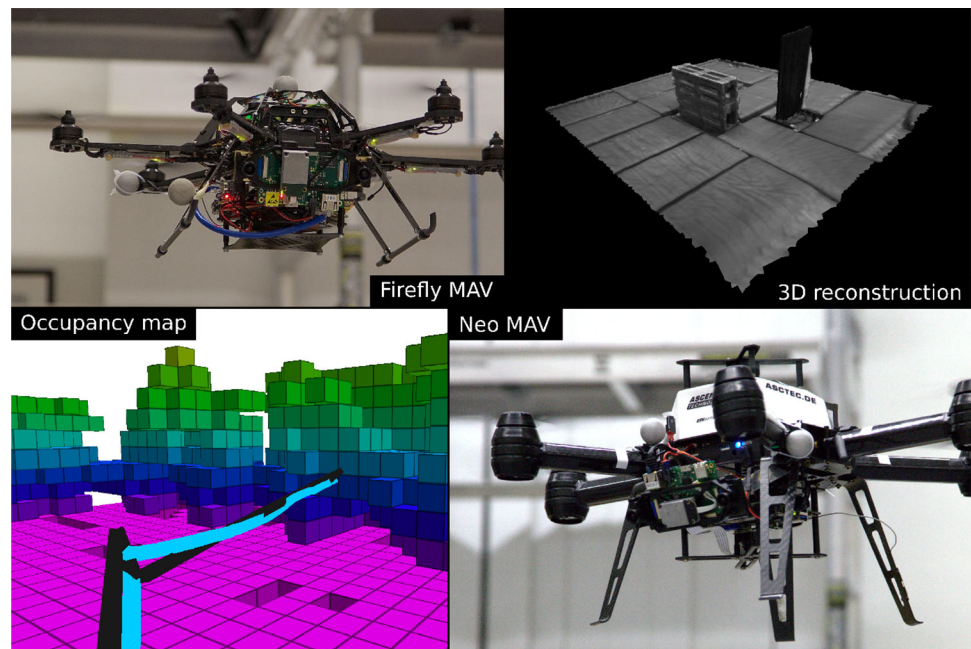
Stentz 2006). It should be noted that the specific problem setup differs depending on whether the interest is to achieve mapping and reconstruction of the surface or the volume of interest—or both of them. Another difference arises based on how much prior knowledge of the environment is available. In case of perfect knowledge, paths can be precomputed, whereas imperfect knowledge requires methods to avoid collisions or occlusions during operation (Fig. 1).

The autonomous *exploration* of a volume with given bounds, but unknown content, is a very complex task. Assuming that the sensor itself has to enter the considered volume, accurate localization, sensing and navigation is required. Moreover, since in general no a priori collision free paths are known, the planner has to run online, deciding on a next step as the exploration advances. The work in Connolly et al. (1985) proposes to solve a Next Best View Problem (NBVP). In order to map a given volume, the next best view is determined based on how much additional, unmapped volume is visible. Repetition of this process can lead to the exploration of the given volume. An advanced version of this iterative approach was presented in Vasquez-Gomez et al. (2014). A previously unknown 2D area is explored in Kuipers and Byun (1991), considering a robotic system with measurement and navigation errors. It employs hierarchical maps to describe the topological objects encountered and derives a distinctiveness map in which the controller steers "uphill". In Yamauchi (1997) the expression of frontier regions was introduced for the first time. The frontier is the border between free and unknown volumes. In their neighborhood the mapping of additional volume can be expected. The mentioned paper proposes an algorithm that finds paths to

poses at these frontiers that maximize the extension of the horizon of known volume, considering an onboard sensor. Repetition of this process leads to exploration of the volume. Advanced variants of this algorithm were presented in González-Banos and Latombe (2002), Adler et al. (2014), and Heng et al. (2015), where the last also improves the coverage of unknown volume along the path to the frontier. The concept has been extended to multi-agent exploration e.g. in Burgard et al. (2000) and Howard et al. (2006). Using a geometric representation of the detected obstacles, Surmann et al. (2003) samples possible viewpoints in the whole known free volume instead of just at the frontiers. Faigl and Kulich (2015) and Faigl et al. (2012) benchmark frontier methods in 2D scenarios for one or multiple robots respectively. Amigoni et al. (2013) presents a study on how the frequencies of decision making and measurement's integration into maps affect the speed and quality of the exploration. In Dewan et al. (2013) a integer programming formulation is employed to plan paths for the collaborative exploration by aerial and ground robots. Multi-Criteria Decision Making approaches are benchmarked in Basilico and Amigoni (2011). The work in Rosenblatt et al. (2002) uses fuzzy logic and utility fusion concepts to autonomously explore coral reefs with underwater vehicles. Finally, in Li et al. (2012) optimal paths for the exploration of an a priori known 2D area are computed offline using the $A^*$ algorithm.

In an *inspection* task, information about a surface to be reconstructed can be provided as a mesh- or voxel-based model. This prior geometric model can be obtained from CAD software, civil engineering instrumentation, Geograph-

ical Information Systems or previous mapping missions, either manually or automatically conducted. One of the early proposed approaches was the adaptation of the NBVP to the inspection of surfaces. This was achieved e.g. by Pito (1999) by forcing the view to contain portions of already mapped surface to guarantee overlap for camera-based reconstruction purposes and thereby influence the views to also detect new parts of the surface. The approach in Banta et al. (2000) deals with finding a next view on the object of interest, for which a set of criteria are proposed, depending on the stage of the reconstruction. The work in Whaite and Ferrie (1997) proposes the reduction of an uncertainty motion of the model to evaluate the next best views. Finally, Yoder and Scherer (2016) proposes an extension to the concept of frontiers, which are the boundaries of inspected parts of a surface embedded in 3D space. This allows the inspection of an object's surface without prior knowledge of its shape. While the abovementioned contributions presented concepts to inspect surfaces in an unknown or partially known environment, most inspection algorithms proposed in the literature assume a perfectly known environment. Solving the classical offline Coverage Planning Problem, the algorithms in Choset and Pignon (1998) and Acar et al. (2002) divide surfaces like floors or landscapes into cells that are covered with a sweeping pattern. Following a different approach, the method presented in Hover et al. (2012) computes inspection paths, dividing the problem into that of finding of a good set of viewpoints and subsequently connecting them with a short path. In Bircher et al. (2016) an alternation between these two steps is proposed in order to improve the resulting path and in Janoušek and Faigl (2013) the inspection problem is solved using self-organizing neural networks. Solving the problem employing a unified approach in order to probabilistically find the optimal inspection path, the works in Papadopoulos et al. (2013) and Bircher et al. (2016) rely on sampling-based concepts to explore the configuration space with a random tree, the branches of which grow to find full coverage solutions. Finally, Alexis et al. (2015) proposes the use of re-meshing strategies to uniformly cover structures represented by 3D triangular meshes. A comprehensive overview of coverage path planning algorithms can be found in Galceran and Carreras (2013) and the references therein.

The general motivation for the presented research is to enable enhanced autonomy of mobile robots in places that are only partially or fully unknown. In such places, exploration is a first task a robot can do to enable achievement of further goals, such as those of comprehensive inspection, semantic identification, or manipulation. To this end, the algorithm presented within this work addresses the problems of autonomous robotic exploration of an unknown 3D volume, as well as the inspection of a given surface in either known, partially known or completely unknown envi-

ronments. While the result of the former is an *occupancy map*, dividing space in free and occupied volume, the latter acquires color and broadly *visual data about a known surface*. In order to solve these two different problems in a unified way, next best views are randomly sampled in the known free space. The quality of the selected views is determined by the amount of visible unmapped volume in the first case and the amount of visible uninspected surface in the second. In contrast to previous contributions, the views are sampled as nodes in a random tree, the edges of which directly provide a collision free path. In most iterations very few sampled viewpoints suffice to determine a reasonably good next step and enable efficient exploration. The robot then only executes the first edge of the tree towards the best viewpoint, after which the whole process is repeated in a receding horizon fashion. This improves the performance of the algorithm and further distinguishes the presented approach from previously known exploration and inspection planning strategies. The lightweight design of the planner allows seamless integration with the robot's control loops. As a result it enables more agile and robust robot behavior with respect to disturbances and uncertainties. In this paper, analysis on the computational complexity for both variants of the proposed algorithm is provided. Essentially, this receding horizon approach to next best view planning for autonomous exploration and inspection aims to close the loop between the planning and perception loops on-board a robotic system, enable its capacity to actively perceive its environment, and through that enhance its autonomy levels in critical mapping, monitoring and exploration tasks. An extensive experimental study has been conducted using aerial robots. Nonetheless, the proposed algorithm is applicable to a wide variety of different robot configurations.

Overall, this paper corresponds to a major extension of the author's previous preliminary work in Bircher et al. (2016). More specifically, this work is not limited to the problem of volumetric exploration but also addresses the problem of autonomous inspection and unifies the two problems of exploration of unknown volume and the inspection of surfaces in a single algorithm. Relevant new formulations of the information gain are also provided. Furthermore, profound analysis of the algorithm properties and computational cost, as well as an updated set of simulation and experimental evaluation studies is provided. Information about the open-source released software package that may be found at Bircher and Alexis (2016) and the online available dataset (Bircher et al. 2016) of the presented experimental results has been added.

The remainder of this paper is organized as follows: The considered problem is defined in Sect. 2 and the proposed solution is described in detail, as well as analyzed regarding computational cost in Sect. 3. Evaluation in simulation and real-world experiments is presented in Sects. 4 and 5. Finally, Sect. 6 gives instructions to download the implementation of

the proposed algorithm, as well as the dataset of an experiment and Sect. 7 summarizes and concludes the presented work.

## 2 Problem formulation

The first problem considered within this work consists in exploring a given bounded 3D volume $V \subset \mathbb{R}^3$. This is to start from an initial collision free configuration and find a path that leads to the mapping of the unknown volume. While $\Xi$ is the simply connected set of collision free configurations $\xi$, a path is given by $\sigma : \mathbb{R} \to \xi$, specifically from $\xi_{k-1}$ to $\xi_k$ by $\sigma_{k-1}^k(s)$, $s \in [0, 1]$ with $\sigma_{k-1}^k(0) = \xi_{k-1}$ and $\sigma_{k-1}^k(1) = \xi_k$. The path has to be such that it is collision free and can be tracked by the vehicle, considering its dynamic and kinematic constraints. The corresponding path cost is $c(\sigma_{k-1}^k)$. To accomplish the task, it has to be determined which parts of the initially unmapped volume $V_{unm} \overset{init.}{=} V$ are free $V_{free} \subseteq V$ or occupied $V_{occ} \subseteq V$. For the mapping process the volume is discretized in an occupancy map $\mathcal{M}$ consisting of cubical voxels $m \in \mathcal{M}$ with edge length $r$. The operation is subject to vehicle dynamic constraints, localization uncertainty and limitations of the employed sensor system. The assumption about the sensor is that it can identify surfaces and thus the free space between the robot and them (e.g. stereo camera, LiDAR). Limitations of the field of view and resolution affect the rate of progress and influence what can be mapped. As for most sensors the perception stops at surfaces, hollow or solid spaces, or narrow pockets can sometimes not be explored with a given setup. This residual volume is denoted by $V_{res}$.

**Definition 1** (*Residual volume*) Let $\Xi$ be the simply connected set of collision free, connected configurations and $\bar{\mathcal{V}}_m \subseteq \Xi$ the set of all configurations from which the voxel $m$ can be mapped. Then the residual volume is given as

$$V_{res} = \bigcup_{m \in \mathcal{M}} (m | \bar{\mathcal{V}}_m = \emptyset).$$

Due to the nature of the problem, a suitable path has to be computed online and in real time, as the free volume to navigate in is not known prior to its exploration. To increase the autonomy of the vehicle, the planner should run onboard with the limited resources available, while other computationally expensive tasks—such as the visual–inertial localization and mapping pipeline—also have to necessarily run onboard.

**Problem 1** (*Volumetric exploration problem*) Given a bounded volume $V$, find a collision free path $\sigma$ starting at an initial configuration $\xi_{init} \in \Xi$ that leads to identifying the free and occupied parts $V_{free}$ and $V_{occ}$ when being executed, such that there does not exist any collision free configuration

from which any piece of $V \setminus \{V_{free}, V_{occ}\}$ could be mapped. Thus, $V_{free} \cup V_{occ} = V \setminus V_{res}$.

Operating within the same constraints, the second problem refers to the inspection of given 2-dimensional manifolds $s \in S$ embedded in a bounded 3D volume $V \subset \mathbb{R}^3$ and with known location. Initially $S$ is not inspected, $S_{uni} \overset{init.}{=} S$ and a path for the vehicle has to be found, such that the whole $S$ is inspected. Again, some parts denoted by $S_{res}$ may be beyond the reach of the sensor, considering the employed system.

**Definition 2** (*Residual surface*) Let $\bar{\mathcal{V}}_s \subseteq \Xi$ be the set of all configurations from which the surface piece $s \subseteq S$ can be inspected. Then the residual surface is given as $S_{res} = \bigcup_{s \in \mathcal{S}} (s | \bar{\mathcal{V}}_s = \emptyset)$.

**Problem 2** (*Surface inspection problem*) Given a surface $S$, find a collision free path $\sigma$ starting at an initial configuration $\xi_{init} \in \Xi$, that leads to the inspection of the part $S_{insp}$ when being executed, such that there does not exist any collision free configuration from which any piece of $S \setminus S_{insp}$ could be inspected. Thus, $S_{insp} = S \setminus S_{res}$.

## 3 Proposed approach

Considering the problems of exploration of unknown volume and inspection of surfaces, the proposed unified approach employs a sampling-based receding horizon path planning paradigm, to generate paths that cover a volume $V$ in the former and a surface $S$ in the latter case. A sensing system that perceives a subset of the environment, e.g. depth camera or a laser scanner is employed to provide feedback on the explored volume. All acquired information is combined in a map representing the world. This volumetric occupancy map is used for both, collision free navigation and determination of the exploration progress. In case of surface inspection, a camera is employed to inspect the given manifold. The structure of this closed loop setup, depicted in Fig. 2, resembles receding horizon controllers like Camacho and Bordons (2003) and Alexis et al. (2012). However, in contrast to the mentioned controllers, the objective of the presented planner focuses on exploration and inspection, while penalizing distance. The continuous feedback by mapping the environment is introduced to help attenuating errors in tracking and perception, analogous to control theory, where the feedback also enhances robustness to such errors and uncertainties.

The employed representation of the environment is a volumetric occupancy map (Hornung et al. 2013) dividing the volume $V$ in cubical volumes $m \in \mathcal{M}$ that can either be marked as free, occupied or unmapped. The resulting array of voxels is saved in an octree, enabling computationally efficient access for operations like integrating new sensor data or checking for occupancy. Paths are only planned through
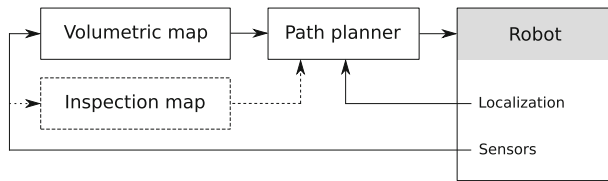
**Fig. 2** Diagram of the system setup. A volumetric occupancy map of the environment is built using the input of the onboard sensors of the robot. At any time, the mapped knowledge of the environment enables the planner to find good exploration paths, starting at the current location. These are executed by the robot and updated during operation. In case of inspection, also the required surfaces are mapped and evaluated. This feedback on the inspection progress enables the planning of good next segments of the inspection paths

known free volume $V_{free}$, thus providing collision free navigation. For a given occupancy map representing the world $\mathcal{M}$, the set of visible and unmapped voxels from configuration $\xi$ is denoted by $\mathbf{Visible_V}(\mathcal{M}, \xi)$. Every voxel $m$ in this set lies in the unmapped exploration area $V_{unm}$, the direct line of sight does not cross occupied voxels and in addition it complies with the sensor model (e.g. inside the Field of View (FoV), maximum range). Starting from the current configuration of the vehicle $\xi_0$, a geometric tree $\mathbb{T}$ is incrementally built in the configuration space. To fulfill the role of geometric tree growing, within this work the RRT algorithm (LaValle 1998) is used. The resulting tree contains $N_{\mathbb{T}}$ nodes $n$ and its edges are given by collision free paths $\sigma$, connecting the configurations of the two nodes. The quality—or information gain—of a node $\mathbf{Gain}(n)$ is the summation of the unmapped volume that can be explored at the nodes along the branch, e.g. for node $n_k$, according to

$$\mathbf{Gain}(n_k) = \mathbf{Gain}(n_{k-1}) + \mu(\mathbf{Visible_V}(\mathcal{M}, \xi_k))e^{-\lambda c(\sigma_{k-1}^k)},$$

(1)

with the tuning factor $\lambda$ penalizing high path costs (González-Banos and Latombe 2002) and $\mu()$ denoting the Lebesgue measure.

To solve the fundamentally different problem of surface inspection of a completely known surface manifold $S$, only minor adaptations to the described scheme are necessary. While initially the whole given surface is uninspected, upon the inspection of piece $s \subseteq S$ it is moved from the set of uninspected surface $S_{uni}$ to the set of inspected surface $S_{insp}$. To this end, Eq. 1 is adapted to account for the area of visible surface $\mu(\mathbf{Visible_S}(\mathcal{M}, \xi))$ instead of the unmapped volume.

For both variants of the algorithm, only the first segment of the branch to the best node $\mathbf{ExtractBestPathSegment}(n_{best})$ is executed by the robot, where $n_{best}$ is the node with the highest gain. Subsequently, the planning step is repeated, resulting in an actual receding horizon strategy. To prevent discarding of already found high quality paths, the remain-

der of the best branch is used to reinitialize the tree in the next planning iteration, re-evaluating its gains using the updated maps $\mathcal{M}$ or $S$. Tree creation is in general stopped at $N_{\mathbb{T}} = N_{\max}$, but if the best gain $g_{best}$ of $n_{best}$ remains zero, the tree construction is continued, until $g_{best} > 0$. Maintaining the best branch for the next planning steps prevents the loss of these computationally expensive paths. For practical purposes, a tolerance value $N_{TOL}$ is chosen significantly higher than $N_{\max}$ and the exploration is considered to be solved when $N_{\mathbb{T}} = N_{TOL}$ is reached, while $g_{best}$ remained zero. This also means that in practice 100 % exploration or coverage are not guaranteed. Algorithm 1 summarizes the planning procedure.

---

**Algorithm 1** Exploration Planner—Iterative Step

---

1: $\xi_0 \leftarrow$ current vehicle configuration
2: Initialize $\mathbb{T}$ with $\xi_0$ and, unless first planner call, also with previous best branch
3: $g_{best} \leftarrow 0$                                      ▷ Set best gain to zero
4: $n_{best} \leftarrow n_0(\xi_0)$                              ▷ Set best node to root
5: $N_{\mathbb{T}} \leftarrow$ Number of initial nodes in $\mathbb{T}$
6: **while** $N_{\mathbb{T}} < N_{\max}$ **or** $g_{best} = 0$ **do**
7:     Incrementally build $\mathbb{T}$ by adding $n_{new}(\xi_{new})$
8:     $N_{\mathbb{T}} \leftarrow N_{\mathbb{T}} + 1$
9:     **if** $\mathbf{Gain}(n_{new}) > g_{best}$ **then**
10:         $n_{best} \leftarrow n_{new}$
11:         $g_{best} \leftarrow \mathbf{Gain}(n_{new})$
12:     **if** $N_{\mathbb{T}} = N_{TOL}$ **then**
13:         Terminate exploration
14: $\sigma \leftarrow \mathbf{ExtractBestPathSegment}(n_{best})$
15: Delete $\mathbb{T}$
16: **return** $\sigma$

---

### 3.1 Implementation details

The described approach was designed to equip an aerial robot with exploration and inspection autonomy, but is also applicable to a wide range of robot configurations including Unmanned Ground Vehicles (UGV) and Autonomous Underwater Vehicles (AUV) or even articulated robots. For all robots with a known two-state boundary value solver (BVS), sampling can be performed in the configuration space, enabling the use of the well known random tree construction algorithms like e.g. RRT or RRT*, offering a fast coverage of the configuration space. For more complex dynamics where BVS are not explicitly known, random trees can conveniently be generated by sampling in the control space and subsequently performing forward integration. This can be used to accommodate actuator saturation or nonholonomic constraints in the presented algorithm. The actual implementation used for the experiments presented in this work is designed to plan for a rotorcraft MAV.

The considered vehicle configuration is the flat state, consisting of position and yaw, $\xi = (x, y, z, \psi)^T$. The vehi-

cle is allowed a maximum translational speed $v_{\max}$ and a constrained yaw rate $\dot{\psi}_{\max}$, both of which are assumed to be small, such that roll and pitch can be assumed to be zero. For slow maneuvering it suffices to use straight lines as tracking reference for the vehicle, specifically $\sigma_{k-1}^k = s\xi_k + (s-1)\xi_{k-1}$, with $s \in [0,1]$. The connection cost of the reference path is considered to be the Euclidean distance, $c(\sigma_{k-1}^k) = \sqrt{(x_k - x_{k-1})^2 + (y_k - y_{k-1})^2 + (z_k - z_{k-1})^2}$. The maximum speed and the yaw rate limit are enforced when sampling the path to generate a reference trajectory.

The employed camera system is assumed to be mounted with a fixed pitch, to have a limited FoV described by vertical and a horizontal opening angle $[\alpha_v, \alpha_h]$, as well as a constraint on the maximum distance that it can effectively sense. While the sensor limitation is $d_{\max}^{\text{sensor}}$, the algorithm uses a value $d_{\max}^{\text{planner}} \leq d_{\max}^{\text{sensor}}$ to determine the volumetric gain of a configuration. A lower $d_{\max}^{\text{planner}}$ ensures both robustness against suboptimal sensing conditions, as well as improved computational performance.

The depth information is incorporated in a probabilistic occupancy map, the specific implementation of which employs the octomap package (Hornung et al. 2013) and provides further functionality for the gain computation and the collision checking of subvolumes. Volumes of the shape of a box around the paths are checked to be free in the occupancy map.

For the surface gain computation the same model with a limited FoV is employed, while $d_{\max}^{\text{sensor}}$ is the range, up to which surfaces are considered to contribute. Information about the structure to inspect is loaded as a triangular surface mesh $S$, which can dynamically be refined to reach a predefined resolution $q$, being the maximum area of a facet. Refining in this case means splitting the triangle in four equal smaller triangles, if the status (state of inspection) is not the same for all. A quadtree data structure enables efficient handling of the refined surfaces, such that the time consumption of operations like gain computation or insertion of a view is minimized. A specific facet $s \in S$ is considered to belong to $\textbf{Visible}_S(\mathcal{M}, \xi)$ if the orientation is facing towards the viewpoint and the line of sight in the occupancy map is free. For the gain computation the unmapped voxels are considered free. For the insertion of a view, the line of sight has to be strictly free.

## 3.2 Computational complexity

As the planner should run onboard a mobile robot like an MAV or similar, short computation times and good scaling properties are crucial. For the exploration problem, the main scenario-dependent parameters that are relevant for these properties are the volume to be explored $V$ and the resolution of the occupancy map $r$. In addition, the number of nodes in

the tree $N_{\mathbb{T}}$ and the choice of the sensor range $d_{\max}^{\text{planner}}$ determine the duration of the path computation. For an occupancy map in an octree data structure queries like checking a voxel for its status (Hornung et al. 2013) have logarithmic complexity in the number of voxels $N_{\text{vox}}^{\text{tot}} = V/r^3$, corresponding to $\mathcal{O}(\log(V/r^3))$. The construction of an RRT tree in a fixed environment is, as shown in Karaman and Frazzoli (2011), of complexity $\mathcal{O}(N_{\mathbb{T}} \log(N_{\mathbb{T}}))$, while the query for the best node in an RRT tree only scales with $\mathcal{O}(N_{\mathbb{T}})$. The number of voxels in the fixed volume around an edge to check for collision scales with $1/r^3$ and the complexity to check the $N_{\mathbb{T}}-1$ edges in the occupancy map is therefore $\mathcal{O}(N_{\mathbb{T}}/r^3 \log(V/r^3))$. Furthermore, for every node the gain has to be computed. Considering a given FoV, the sensor volume is proportional to $V_{\text{sensor}} \propto (d_{\max}^{\text{planner}})^3$. The number of voxels to test is therefore approximately equal to $N_{\text{vox}}^{\text{sensor}} \approx \frac{V_{\text{sensor}}}{r^3}$. A ray cast to check visibility is performed for every individual voxel. Its complexity scales with the number of voxels on the ray $\mathcal{O}(d_{\max}^{\text{planner}}/r)$, resulting in $\mathcal{O}(d_{\max}^{\text{planner}}/r \log(V/r^3))$ for the visibility check and $\mathcal{O}((d_{\max}^{\text{planner}}/r)^4 \log(V/r^3))$ for one gain computation. As the construction of the RRT tree dominates its query, the total complexity of a single replanning step results in the sum of tree construction, collision checking and gain computation

$$
\begin{aligned}
&\mathcal{O}(N_{\mathbb{T}} \log(N_{\mathbb{T}}) + N_{\mathbb{T}}/r^3 \log(V/r^3) \\
&\quad + N_{\mathbb{T}}(d_{\max}^{\text{planner}}/r)^4 \log(V/r^3))
\end{aligned}
\tag{2}
$$

Notably, the complexity only depends logarithmically on the volume of the exploration problem when the map resolution and planning horizon are fixed. For very large scenarios and when most space has been mapped, it may happen that the number of necessary nodes to find any gain exceeds $N_{\max}$. These iterations generally take longer to compute but as these expensive solutions are kept for the next iterations, the frequency with which this case occurs is maintained at a minimum.

When planning for the inspection of a given surface, tree construction and collision checking remain the same, only the gain computation is altered. The necessary computation time scales with the number of uninspected facets in the considered triangular mesh $N_{uni}^{fac}$. A naive implementation checks all facets for visibility, in the implementation used for the presented experiments this is done in the occupancy map. As the distance at which facets are considered is bounded by $d_{\max}^{\text{planner}}$, a conservative estimation of the complexity results in $\mathcal{O}(N_{uni}^{fac} d_{\max}^{\text{planner}}/r \log(V/r^3))$. It is noted that a constant number of ray casts is assumed per facet, as they have approximately the same size. Scaling the gain computation complexity with the number of viewpoint configurations $N_{\mathbb{T}}$, the total complexity for the inspection path planning algorithm results therefore in
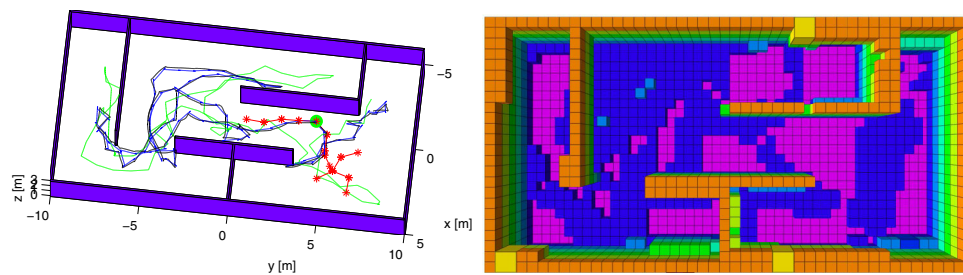
**Fig. 3** The apartment setup is shown together with an instant of the exploration mission on the *left side*. *Blue* is the reference path of the whole mission, while the vehicle response is depicted in *black*. The *green point* denotes the root of the RRT tree shown in *red*. An alterna-

$$\mathcal{O}(N_{\mathbb{T}} \log(N_{\mathbb{T}}) + N_{\mathbb{T}}/r^3 \log(V/r^3)$$
$$+ N_{\mathbb{T}} N_{uni}^{fac} d_{max}^{planner}/r \log(V/r^3)). \tag{3}$$

Also this planner's complexity depends only logarithmically on the considered volume. Moreover, as the inspection progresses, the third term for the gain computation vanishes, as the number of uninspected surfaces decreases.

# 4 Simulation based evaluation

In order to systematically evaluate the potential of the proposed exploration and inspection planners, simulation studies have been performed using a hexacopter MAV. Two scenarios of different size are considered, while in both the volumetric exploration planner is compared to a frontier-based approach (similar to González-Banos and Latombe (2002)).

## 4.1 Simulation environment

Since the planners interact in a closed loop with the robot's perception, estimation and control loops, a detailed and realistic simulation is required. The Gazebo-based simulation environment RotorS[1] has been used along with the provided model of the AscTec Firefly hexacopter MAV.[2] It employs a stereo camera that gives real-time feedback of the simulated environment, both as an optical image stream and depth images. In order to evaluate specifically the path planning, the simulated depth images are fed to the map instead of computing them from the stereo images. Further, ground truth is used to bypass the odometry algorithms. The specifications of the perceived area are a FoV of $[60, 90]°$ in vertical and horizontal direction, respectively, while it is mounted with a downward pitch angle of $15°$. For all simulations, the size of the box model for collision detection is assumed to have a

---

[1] RotorS: An MAV gazebo simulator, https://github.com/ethz-asl/rotors_simulator.

[2] Ascending Technologies GmbH, http://www.asctec.de/.

tive exploration path computed by the frontier-based algorithm is shown with a *green line*. The floor and the roof are not visualized. On the *right side*, the occupancy map is depicted, with the voxels colored according to their height (Color figure online)

size of $0.5 \times 0.5 \times 0.3\,\text{m}^3$. Computation was performed on a computer with a i7 2.8GHz processor and 16GB RAM.

## 4.2 Apartment exploration scenario

The first scenario refers to a $20 \times 10 \times 3\text{m}^3$ apartment space, divided in different rooms by walls as depicted in Fig. 3. The vehicle starts in the centre and navigates through narrow passages in order to explore the whole volume or inspect the whole surface, respectively for the two variations of the proposed planner. The surface mesh model for the latter has been derived from the occupancy map acquired in the former and covers all walls, floor and ceiling. The resolution of the online built occupancy map is $r = 0.4\,\text{m}$ and the surface mesh is refined, such that all facets have an area of less than $0.5\,\text{m}^2$. To ensure high accuracy, the maximum sensing range for the employed depth sensor is set to $d_{max}^{sensor} = 5\,\text{m}$, while the planner only considers a maximum range of $d_{max}^{planner} = 2\,\text{m}$. Table 1 summarizes the complete set of employed parameters.

In order to assess the performance and overall quality of the exploration planner, a comparison with a frontier-based planner has been conducted. The work in González-Banos and Latombe (2002) was adapted to work with the occupancy map and the specific problem setup. For this purpose, free voxels in the neighborhood of unmapped voxels have been considered as frontier voxels. To allow collision-free placement of the vehicle, voxels up to half the collision box diagonal from the frontier cells are considered as well. The positions at the candidate voxels' center are evaluated for a set of randomly sampled orientations according to $\textbf{Gain}(n) = \mu(\textbf{Visible}_V(\mathcal{M}, \xi))e^{-\lambda l}$, where $l$ is the distance to the current vehicle location determined by an RRT. The resulting path is executed by the robot and recomputed upon reaching the frontier configuration.

For both exploration planners the scenario was executed 10 times, as the outcome is stochastic due to the use of the RRT algorithm. For the analysis, the total exploration time $t_{tot}$ is split into execution time $t_{ex}$ and computation time $t_{comp}$. The proposed planner displayed a mean total

**Table 1** Apartment exploration scenario parameters

| Parameter | Value | Parameter | Value |
|---|---|---|---|
| Area | $20 \times 10 \times 3\,\mathrm{m}^3$ | Collision box | $0.5 \times 0.5 \times 0.3\,\mathrm{m}^3$ |
| Volumetric map resolution $r$ | $0.4\,\mathrm{m}$ | Inspection mesh resolution $q$ | $0.5\,\mathrm{m}^2$ |
| $v_{max}$ | $0.2\,\mathrm{m/s}$ | $\dot{\psi}_{max}$ | $0.75\,\mathrm{rad/s}$ |
| FoV | $[60, 90]°$ | Mounting pitch | $15°$ |
| $d_{max}^{planner}$ | $2\,\mathrm{m}$ | $d_{max}^{sensor}$ | $5\,\mathrm{m}$ |
| $\lambda$ | $0.5$ | RRT max edge length $\epsilon$ | $1\,\mathrm{m}$ |
| $N_{max}$ | $15$ | $N_{TOL}$ | $200$ |



**Fig. 4** On the *left* statistics compare the Receding Horizon Next Best View Planner for exploration (*blue*) with the frontier-based approach (*red*). The depicted curves show the mean exploration progress over the execution time (computation time has been subtracted). The standard deviation is given every minute of execution time. The same is depicted on the *right side* for the inspection planner, comparing the results with no prior knowledge of the environment to the ones with a given occupancy map (like the one in Fig. 3) (Color figure online)
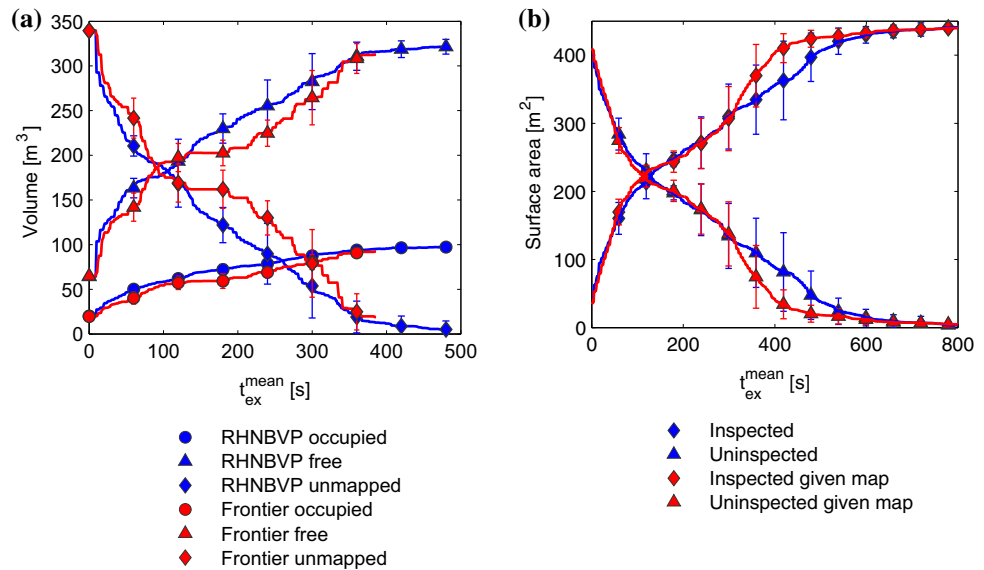
time for the exploration of $t_{tot}^{mean} = 501.9\,\mathrm{s}$ with a standard deviation of $t_{tot}^{\sigma} = 79.4\,\mathrm{s}$, where the computation time was $t_{comp}^{mean} = 15.2\,\mathrm{s}$ with standard deviation $t_{comp}^{\sigma} = 2.7\,\mathrm{s}$. A single iteration's computation lasted on average $0.153\,\mathrm{s}$, with a standard deviation of $0.139\,\mathrm{s}$. The frontier algorithm consumed on average $t_{tot}^{mean} = 469.7\mathrm{s}$, but also had more voxels that could not be inspected ($V_{res}$) at the time of termination. This is due to the limited choice of viewpoints only at the frontiers. When comparing the exploration progress over the execution time in Fig. 4, even in this simple scenario the proposed planner performs slightly better. Furthermore, the total computation time for the frontier-based case is with an average of $t_{comp}^{mean} = 83.8\,\mathrm{s}$ on a significantly higher level. Because in every iteration the whole computed path is executed, $t_{comp}^{mean}$ distributes on less iterations, resulting in an average replanning time of $5.7\,\mathrm{s}$, in which the robot waits for the next path segment to be computed. Sample results for paths of both planners are depicted in Fig. 3, together with the occupancy map acquired by a mission using the proposed planner.

This occupancy map was used to derive a surface mesh model, depicted in Fig. 5, together with a sample path of the proposed inspection planner. The resolution threshold of the surface mesh was set to $q = 0.5\,\mathrm{m}^2$. Using the
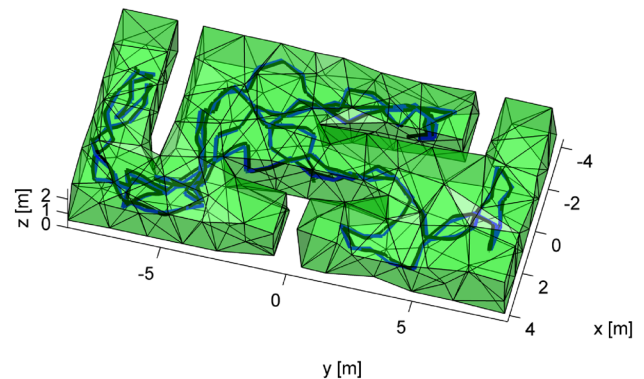


**Fig. 5** The mesh representation of the wall, ceiling and floor surface of the apartment is depicted, together with a sample inspection path (*blue*) and the recorded vehicle response (*black*) (Color figure online)

exactly same setup, but with an altered objective for uninspected surface area, the scenario was run 10 times, once without prior knowledge of the environment and a second time with a given complete occupancy map, recorded in a prior exploration run. Figure 4b depicts the inspection progress over time for both flavours of the scenario, while in the following the time consumptions are summarized (corresponding values for the inspection in known environ-

**Fig. 6** The bridge model is displayed along with the computed exploration path (*blue*), the simulated vehicle response (*black*) and the acquired occupancy map in the lower part of the figure. These results are given for the proposed exploration planner *on the left side* and the compared frontier-based planner on the right. Note that the latter only made partial exploration due to excessive computation time for this large scale problem. The figures are available as Matlab figures on Bircher et al. (2016) for further inspection by the interested reader (Color figure online)
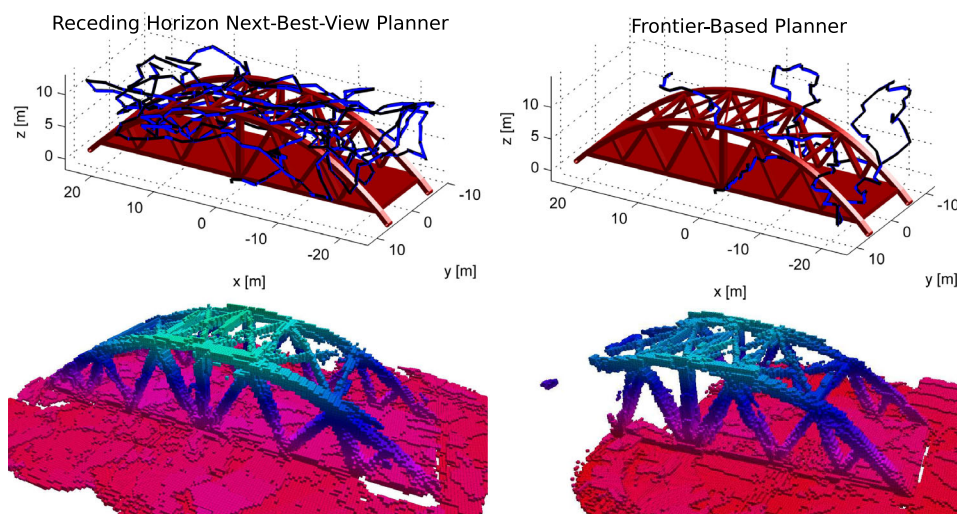
ment are given in parenthesis). The inspection mission lasted on average $t_{tot}^{mean} = 1007.2$ s (1031.5 s), with a standard deviation of $t_{tot}^{\sigma} = 229.8$ s (229.5 s). In this time, the percentage of the area that has been inspected is on average $\mu(S_{insp}^{mean})/\mu(S) = 99.6\%$ (99.3 %) with standard deviation of $\mu(S_{insp}^{\sigma})/\mu(S) = 0.27\%$ (0.27 %). The share of computation time is $t_{comp}^{mean} = 7.14$ s (7.13 s) with standard deviation $t_{comp}^{\sigma} = 0.74$ s (0.99 s). A single iteration's computation lasted on average 0.036 s (0.035 s), with a standard deviation of 0.047 s (0.049 s). Overall, the results in this scenario display a fast inspection progress in an initial phase with a slower rate as the mission proceeds, which is typical for greedy, locally planning schemes. However, initial knowledge of the environment can improve the convergence to the final inspection status, as can be seen in Fig. 4b.

This simulation scenario has shown that the performance of the proposed exploration planner is not inferior to a frontier-based approach, despite the low number of viewpoints considered per iteration. It even succeeds in finding some voxels that remain hidden to the compared algorithm. Moreover, its short computation time allows the seamless integration into the robot's control and path planning loops. At the same time, it demonstrated that changing the objective to the inspection of surfaces leads to a planning scheme, suitable to solve the problem of inspection of a given surface. While this very basic scenario highlights the ability of the presented planner to efficiently explore a given volume, a more challenging scenario will reveal its high performance and overall capacity.

### 4.3 Exploration of a volume containing a bridge

A second simulation scenario refers to a $50 \times 26 \times 14\,\mathrm{m}^3$ volume containing a bridge,[3] as depicted in Fig. 6. Starting

at the side of the bridge, in a first mission the robot has to explore the whole volume, while at the same time mapping the bridge in the occupancy map, the resolution of which is $r = 0.25$ m. Due to the larger size, the higher resolution and the geometrical complexity of the bridge, this scenario is a much more challenging problem. Table 2 summarizes the set of all employed parameters.

The total time for a single run has been $t_{tot} = 43.8$ min, where in total $t_{comp} = 9.4$ min have been spent on computation. A single replanning lasted on average 1.6 s. The median of only 1 s reveals that a large portion of the computation time has been spent for the cases, where after $N_{max}$ iterations no gain has been found. These replanning steps took up to 23 s. In contrast, the frontier-based comparison run has been aborted after $t_{tot} = 1670.1$ min. Until then, the planner spent $t_{comp} = 1660.4$ min for computation and on average 25.9 min per replanning step.

The progress over the total exploration time for both planners is displayed in Fig. 7. While the proposed planner reduces the unexplored volume significant in a short time before the curve flattens, the compared frontier-based planner only very slowly progresses, due to excessive computation time. Figure 6 depicts the computed paths on top of the simulated model, as well as the occupancy maps at the end of the executions.

In a second mission, the surface model of the bridge was assumed to be known and a slightly inflated mesh model was derived, such that—despite an inaccurate occupancy map—the surface could still be detected to be visible. The mesh resolution threshold was set to $q = 0.5\,\mathrm{m}^2$. Using exactly the same setup as in the exploration mission, but with an objective for surface inspection, a single execution was simulated. The total inspection lasted for $t_{tot} = 105.0$ min with a total computation time of $t_{comp} = 7.4$ min. On average, a single replanning lasted only 0.498 s. As opposed to the

---

[3] Bridge Model, 3D Warehouse, https://3dwarehouse.sketchup.com/.

**Table 2** Bridge exploration scenario parameters

| Parameter | Value | Parameter | Value |
|---|---|---|---|
| Area | $50 \times 26 \times 14 \, \text{m}^3$ | Collision box | $0.5 \times 0.5 \times 0.3 \, \text{m}^3$ |
| Volumetric map resolution $r$ | 0.25 m | Inspection mesh resolution $q$ | 0.5 m |
| $v_{max}$ | 0.5 m/s | $\dot{\psi}_{max}$ | 0.75 rad/s |
| FoV | $[60, 90]°$ | Mounting pitch | 15° |
| $d_{max}^{planner}$ | 2 m | $d_{max}^{sensor}$ | 10 m |
| $\lambda$ | 0.2 | RRT max edge length $\epsilon$ | 3 m |
| $N_{max}$ | 30 | $N_{TOL}$ | 5000 |

**Fig. 7** These two figures depict the exploration progress for the proposed planner on the *left* and the frontier-based comparison algorithm on the *right* for the bridge scenario. The *solid black line* denotes the unmapped volume, decaying as exploration progresses. The *red dashed line* denotes known free volume and the *blue alternately dashed* and *dotted line* denotes occupied volume (Color figure online)
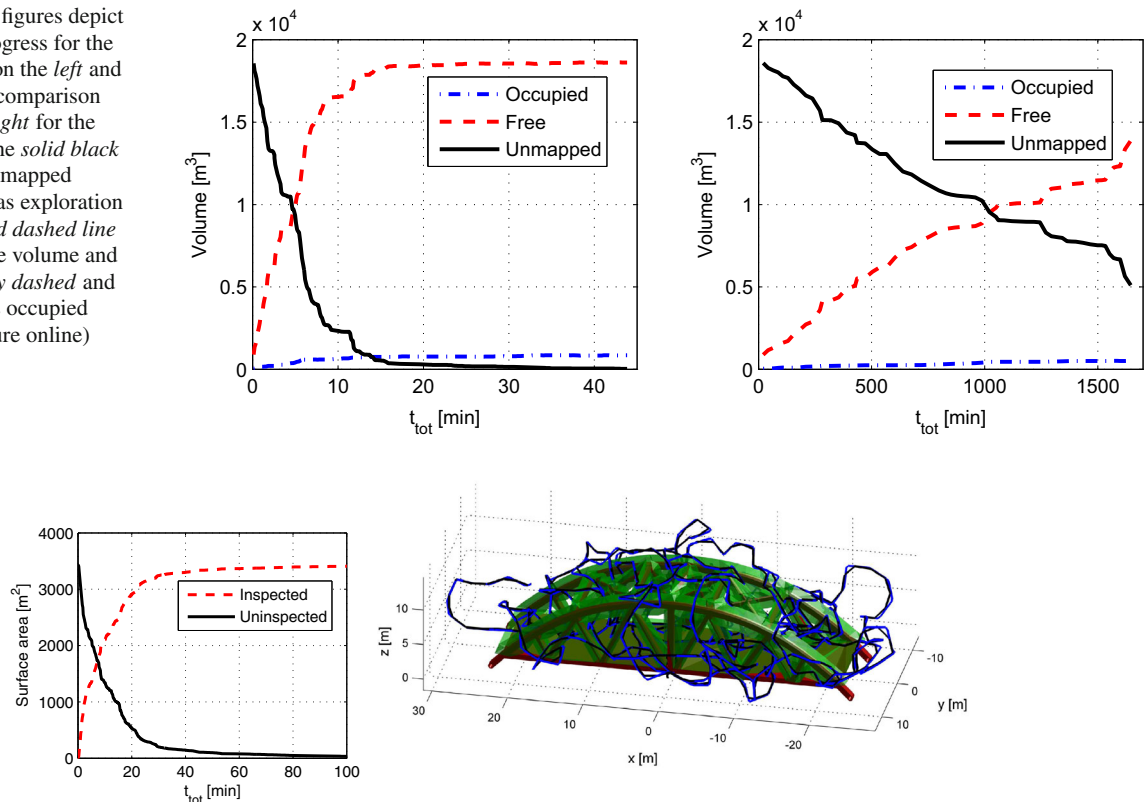




**Fig. 8** The inspection progress of the simulated bridge inspection mission is displayed on the *left side*. While most of the surface is inspected within the first 20 min, much more time is required for the remaining uninspected surfaces. *On the right* the simulated bridge model, together with the employed surface mesh (*green*), the computed inspection path (*blue*) and the vehicle response (*black*) are depicted. A version of the figure is available as Matlab figure on Bircher et al. (2016) for further inspection by the interested reader (Color figure online)

exploration missions, where the average computation time increases over time, the inspection planning becomes more efficient due to the decreasing surface that has to be considered in the gain computation. An indicative result supporting this is the average computation time of the first 100 iterations of 2.16 s and the last 100 iterations of 0.153 s. The inspection progress over time is depicted in Fig. 8 on the left side, while the right shows the bridge with the employed surface mesh, the path and the corresponding vehicle response. The final ratio of inspected area was $\mu(S_{insp})/\mu(S) = 99.1\%$.

This second, more complex scenario reveals the good scaling properties of the proposed planner, which is capable of handling large problems without excessive runtime. At the same time, the implementation of an alternative frontier-based exploration planning algorithm proves to be less suitable for more complex exploration problems, as the computation time quickly grows beyond feasibility.

## 5 Experimental evaluation

Real world experiments were conducted to further evaluate and demonstrate the performance of the proposed concept. Running with the limited onboard resources and in real-time, imposes fixed and tight limits on the amount of computations that are feasible.

### 5.1 The firefly aerial robot platform

The first experiment was performed using the AscTec Firefly hexacopter MAV (see footnote 2) onboard of which the Visual–Inertial Sensor (VI-Sensor) developed by the Autonomous Systems Lab at ETH Zurich and Skybotix AG is integrated. The sensor provides stereo images hardware time-synchronized with the data from a high quality Inertial Measurement Unit (IMU). They are used by a visual–inertial odometry algorithm for pose estimation (Bloesch et al. 2015; Lynen et al. 2013). Further, also a point cloud of the environment is constructed from the stereo images and is used for the environment mapping. Using the full state estimate feedback, a trajectory tracking model predictive controller is implemented on the aerial robot that considers the vehicle dynamics and provides attitude and thrust reference for the low-level controller running on the autopilot provided by the MAV manufacturer. Mapping, estimation, high-level control and planning are running on an on-board computer with a i7 3.1 GHz processor and 8 GB RAM.

An external motion capture system (VICON) was used only to monitor the vehicle ground-truth state and provide the option to intervene in case of failure using a programmed "safety pilot".

### 5.2 Exploration experiment scenario

The experimental scenario refers to a closed room with a size of $9 \times 7 \times 2 \, \text{m}^3$. It contains scaffold elements at the walls, which set high demands in terms of perception accuracy and robustness, as well as collision free navigation capability, as the structure consists of thin poles. The employed aerial robot and the scaffold structure are depicted in Fig. 9. The attached stereo camera module's FoV is $[60, 90]°$ and it is mounted with a downward pitch of $15°$. For planning purposes, a sensing range of $d_{\max}^{\text{planner}} = 1 \, \text{m}$ is considered, while the sensor's range is set to $d_{\max}^{\text{sensor}} = 5 \, \text{m}$. The $\lambda$-parameter for the gain computation is set to 0.5, which corresponds to a strong penalty on distance. Collision checking is performed in the $r = 0.2 \, \text{m}$ resolution occupancy map. All the employed parameters are summarized in Table 3, while Fig. 10 shows the progress at distinct instants of the mission execution. The colored voxels depict the occupied space, while encoding their height by the color. The exploration starts at $t_{tot} = 0$, after an initialization motion to allow the computation of a first collision free path segment. The MAV subsequently explores the volume, first mostly nearby regions by yawing, which is favored by the high $\lambda$-parameter and then also by moving to different regions of the volume to explore. The vehicle accurately follows (light blue) the computed path (black), which favors the precise and swift exploration of remaining unmapped areas. The high
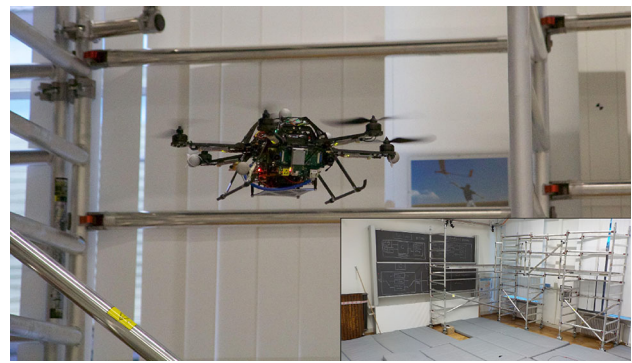


**Fig. 9** The AscTec Firefly hexacopter MAV is depicted, together with the scaffold structure used for the exploration experiment

**Table 3** Scaffold exploration experiment parameters

| Parameter | Value | Parameter | Value |
|---|---|---|---|
| Area | $9 \times 7 \times 2 \, \text{m}^3$ | Volumetric map resolution $r$ | $0.2 \, \text{m}$ |
| $v_{\max}$ | $0.25 \, \text{m/s}$ | $\dot{\psi}_{\max}$ | $0.3 \, \text{rad/s}$ |
| FoV | $[60, 90]°$ | Mounting pitch | $15°$ |
| $d_{\max}^{\text{planner}}$ | $1 \, \text{m}$ | $d_{\max}^{\text{sensor}}$ | $5 \, \text{m}$ |
| $\lambda$ | $0.5$ | RRT max edge length $\epsilon$ | $1.5 \, \text{m}$ |
| $N_{\max}$ | $20$ | Collision box | $1.2 \times 1.2 \times 0.5 \, \text{m}^3$ |

resolution of the employed occupancy map requires views from different directions to explore voxels that may be hidden behind parts of the scaffold. The exploration finishes at $t_{tot} = 253.37 \, \text{s}$ and after 58 planning iterations. Of the total time, $t_{comp} = 11.5 \, \text{s}$ have been used for computation, which corresponds to an average value of $0.199 \, \text{s}$ per computation step (standard deviation $0.125 \, \text{s}$).

This experiment demonstrates the applicability of the proposed algorithm to exploration using real small aerial robots. Furthermore, the very short computation times highlight the high performance in computing the next path segment. A second experiment will focus on the inspection of a given structure.

### 5.3 The neo aerial robot platform

For the second experiment an AscTec Neo hexacopter MAV was employed (see Fig. 11). In addition to the VI–Sensor, mounted equally as on the Firefly MAV, an upward facing camera offers additional convenience in exploration and inspection missions. Point clouds are generated using a monocular version of the depth reconstruction algorithm, considering consecutive images with the travelled distance as baseline, which is again estimated using a visual–inertial odometry algorithm (Bloesch et al. 2015; Lynen et al. 2013). As with the Firefly MAV, the computed trajectories are

**Fig. 10** The exploration experiment in a closed room is depicted. The *colored* voxels represent occupied parts of the occupancy map (*colored* according to their height) while the computed path is given in *black* and the vehicle response in *light blue*. The initial phase of the exploration mission is dominated by yawing motions to maximize exploration without traveling large distances. Subsequently, the MAV explores regions further away, to eventually accomplish its mission (Color figure online)
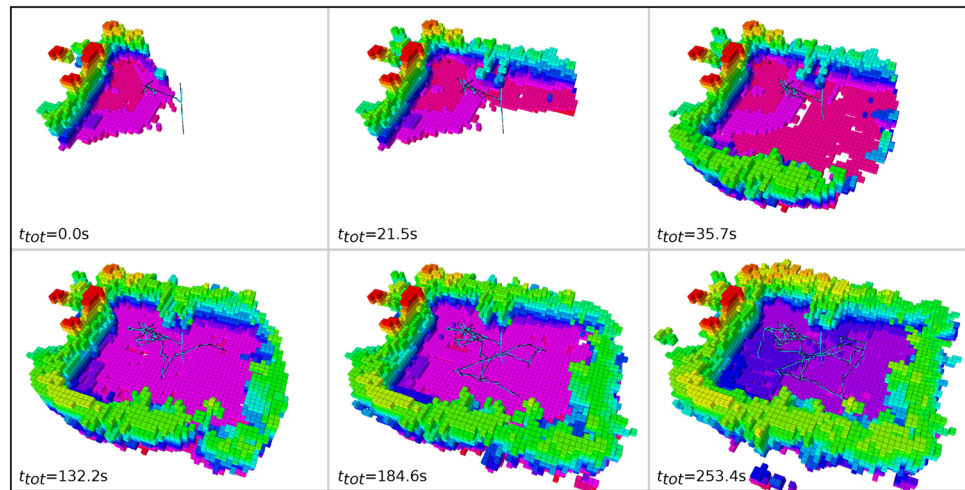


**Fig. 11** The AscTec Neo hexacopter MAV is depicted, with the fields of view of the stereo camera pair and the upward facing camera visualized in *green* (Color figure online)

tracked using a model predictive controller. The Neo features the same on-board computer as the Firefly.

### 5.4 Inspection experiment scenario

In the second experiment, the objects of interest are two wooden pallets and a rusty boiler, arranged in a setup within a volume of $6 \times 6 \times 3\,m^3$ as depicted in Fig. 12. The objective is to perform a first exploration flight, as the objects and their environment have to be identified. In a second flight, using a mesh model of the object's surface, an inspection mission is executed. The AscTec Neo hexacopter MAV, as depicted in Fig. 11, features a stereo camera pair and an upward facing camera. Due to the sampling based nature of the algorithm, a second camera can conveniently be accommodated in the gain checking and insertion of the point clouds in the map. Both camera's measurement ranges are limited by $d_{max}^{sensor} = 5\,m^3$, while the planner only considers $d_{max}^{planner} = 2.5\,m$. This camera setup offers increased inspection and exploration performance, as the field of view is increased to the single camera

case. It enables overhead volume exploration and oblique surfaces can be inspected with higher accuracy. The robot's motions are constrained by a maximum translational velocity of $v_{max} = 0.2\,m/s$ and a yaw rate limit of $\dot{\psi}_{max} = 0.5\,rad/s$. The parameter $\lambda = 0.5$ imposes a high cost on distance, such that close-by gain is collected first, rather than going back and forth. A box model of the robot of $1.2 \times 1.2 \times 0.5\,m^3$ is considered for the collision checking in the occupancy map, which has a resolution of $r = 0.3\,m$. The employed parameters are summarized in Table 4.

In the first part of the mission, the occupancy map is created by exploring the bounded volume. The resulting map along with the performed path is depicted in Fig. 12. The average computation time during the $t_{tot} = 284.4\,s$ exploration was $t_{comp}^{mean} = 0.730\,s$ with a standard deviation of $t_{comp}^{\sigma} = 0.104\,s$.

Starting with the constructed occupancy map, the second part of the mission was to inspect the surface of the objects of interest. For this purpose, a mesh model with a resolution threshold of $q = 0.01\,m^2$ has been employed as depicted on the left side of Fig. 13. For robustness with respect to the detection method of the inspected surfaces, a slightly inflated box mesh model of the considered objects has been used. The resulting inspection path is depicted on the right side of Fig. 13, together with an offline reconstructed point cloud of the perceived structure. The total inspection lasted for $t_{tot} = 322.0\,s$, whereof $t_{comp} = 30.9\,s$ have been used for the computation, on average $t_{comp}^{mean} = 0.469s$ with a standard deviation of $t_{comp}^{\sigma} = 0.091\,s$. The point cloud in Fig. 13 has been computed using the Pix4d 3D reconstruction software,[4] using the monocular image streams. Its density indicates the quality of the inspection results.

This second experiment does not only show the planner's capability to inspect surfaces, but also proposes the use of a

---

[4] Pix4D, http://pix4d.com/.

**Fig. 12** On the *left side*, the two objects of interest in the second experiment are depicted. The first consists of two wooden pallets, while the second is a rusty piece of boiler. On the *right side*, the occupancy map acquired during the exploration is depicted, showing the two objects in the middle of a $6 \times 6 \, \text{m}^2$ volume. The computed exploration path is depicted in *black*, while the vehicle response is overlaid in *light blue* (Color figure online)
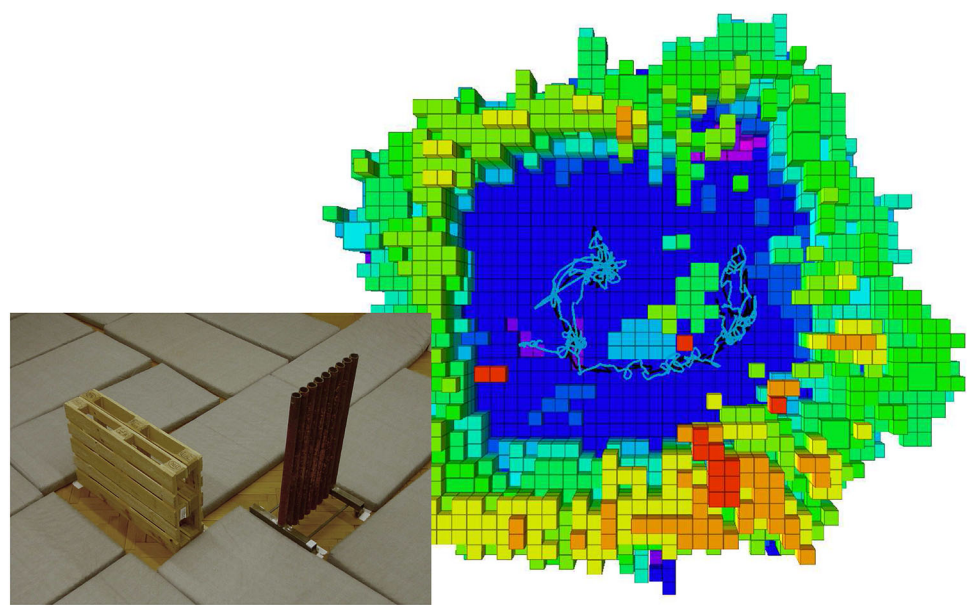


**Table 4** Exploration and inspection experiment parameters

| Parameter | Value | Parameter | Value |
|---|---|---|---|
| Area | $6 \times 6 \times 3 \, \text{m}^3$ | Collision box | $1.2 \times 1.2 \times 0.5 \, \text{m}^3$ |
| Volumetric map resolution $r$ | $0.3 \, \text{m}$ | Inspection mesh resolution $q$ | $0.01 \, \text{m}^2$ |
| $v_{\max}$ | $0.2 \, \text{m/s}$ | $\dot{\psi}_{\max}$ | $0.5 \, \text{rad/s}$ |
| FoV | $[60, 90]°, [60, 90]°$ | Mounting pitch | $15°, -90°$ |
| $d_{\max}^{\text{planner}}$ | $2.5 \, \text{m}$ | $d_{\max}^{\text{sensor}}$ | $5 \, \text{m}$ |
| $\lambda$ | $0.5$ | RRT max edge length $\epsilon$ | $0.5 \, \text{m}$ |
| $N_{\max}$ | $20$ | $N_{TOL}$ | $200$ |

**Fig. 13** In the second part of the second experiment two objects are inspected. Their occupancy map signature is depicted on the top of the *left side*, while beneath the surface mesh model at an instant of the mission is overlaid. The *red* parts have already been inspected, while the *blue* part still needs to be covered. On the *right side* the computed inspection path is plotted in *blue* and the corresponding vehicle response in *black*. The depicted environment is a point cloud reconstructed with the Pix4d software (see footnote 4) using monocular image streams (Color figure online)
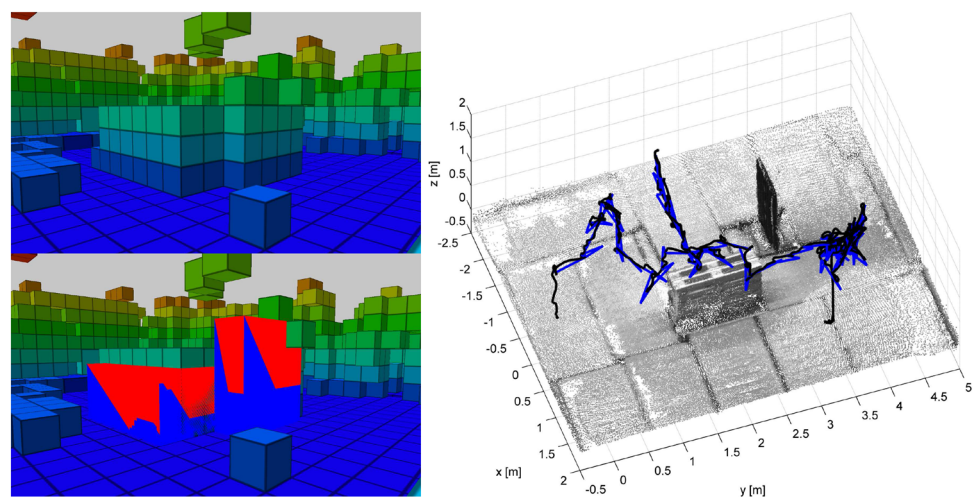


combination of the two presented planners in order to first explore a given volume to subsequently inspect structures that have been identified during the first part. Such a procedure is relevant, e.g. in industrial inspection, where frequent missions take place in a changing environment. Prior explo-

ration to acquire a high fidelity model of the structures and obstacles enable the subsequent inspection of the updated zones of interest, while at the same time ensuring collision free operation. A video of the recorded result is available at https://youtube.com/D6uVejyMea4.

## 6 Code and dataset release

As a part of the contribution of this paper, an implementation of the proposed algorithms is available online (Bircher and Alexis 2016). The package comprises the planner, as well as an interface for the integration with the Gazebo-based simulator RotorS (see footnote 1) along with demo scenarios. While various types of robots can be integrated with the existing framework, an implementation to plan for multicopters using RRT is provided. Additionally, a dataset with postprocessed data recorded during the presented experiment is released (Bircher et al. 2016) to enable detailed inspection of the achieved quality of the results.

## 7 Summary and conclusion

A new path planning scheme has been introduced that enables the online planning of good paths to explore a given bounded volume in a receding horizon manner. It has been shown that with minor adaptation of the objective function, the proposed planning method finds good paths for the inspection of an a priori known surface in either known or unknown environment. These different scenarios have been evaluated in challenging simulations and further validated in real world experiments using rotorcraft MAVs. Analysis on the computational complexity has been provided and the good scaling properties with respect to the scenario size have further been highlighted in the presented simulation scenarios of different scale. The implementation of the proposed path planner is released at Bircher and Alexis (2016) for use and further development by the community along with sample results, which will continuously be updated. A sample dataset of postprocessed experimental data is available at Bircher et al. (2016). Further work could be dedicated to plan for various robotic platforms like fixed wing UAVs or robot arms, or large scale experimental scenarios. Towards the direction of higher dimensional states, a 6 degree of freedom rotorcraft MAV model in combination with a random tree sampled in control space could enable the planning of exploration paths for dynamic flight.

## References

Acar, E. U., Choset, H., Rizzi, A. A., Atkar, P. N., & Hull, D. (2002). Morse decompositions for coverage tasks. *The International Journal of Robotics Research*, *21*(4), 331–344.

Adler, B., Xiao, J., & Zhang, J. (2014). Autonomous exploration of urban environments using unmanned aerial vehicles. *Journal of Field Robotics*, *31*(6), 912–939.

Alexis, K., Papachristos, C., Siegwart, R., & Tzes, A. (2015). Uniform coverage structural inspection path-planning for micro aerial vehicles. In IEEE international symposium on intelligent control (ISIC) (pp. 59–64). IEEE.

Alexis, K., Nikolakopoulos, G., & Tzes, A. (2012). Model predictive quadrotor control: attitude, altitude and position experimental studies. *Control Theory & Applications, IET*, *6*(12), 1812–1827.

Amigoni, F., Li, A. Q., & Holz, D. (2013) Evaluating the impact of perception and decision timing on autonomous robotic exploration. In *2013 european conference on mobile robots (ECMR)* (pp. 68–73). IEEE.

Banta, J. E., Wong, L. M., Dumont, C., Abidi, M., et al. (2000). A next-best-view system for autonomous 3-d object reconstruction. *IEEE Transactions on Systems, Man and Cybernetics, Part A: Systems and Humans*, *30*(5), 589–598.

Basilico, N., & Amigoni, F. (2011). Exploration strategies based on multi-criteria decision making for searching environments in rescue operations. *Autonomous Robots*, *31*(4), 401–417.

Bircher, A., Alexis, K. (2016). Receding Horizon Next Best View Planner Code Release. https://github.com/ethz-asl/nbvplanner

Bircher, A., Alexis, K., Schwesinger, U., Omari, S., Burri, M., & Siegwart, R. (2016). An incremental sampling-based approach to inspection planning: the rapidly exploring random tree of trees. *Robotica* (pp. 1–14).

Bircher, A., Kamel, M., Alexis, K., & Siegwart, R. (2016). Next Best View Planning Dataset Release. https://github.com/ethz-asl/nbvplanner/wiki/Example-Results.

Bircher, A., Kamel, M., Alexis, K., Oleynikova, H., & Siegwart, R. (2016). Receding horizon "next-best-view" planner for 3D exploration. In *2016 IEEE international conference on robotics and automation (ICRA)* (pp. 1462–1468). IEEE.

Bircher, A., Kamel, M., Alexis, K., Burri, M., Oettershagen, P., Omari, S., et al. (2016). Three-dimensional coverage path planning via viewpoint resampling and tour optimization for aerial robots. *Autonomous Robots*, *40*(6), 1059–1078.

Bloesch, M., Omari, S., Hutter, M., & Siegwart, R. (2015). Robust visual inertial odometry using a direct ekf-based approach. In *2015 IEEE/RSJ international conference on intelligent robots and systems (IROS)* (pp. 298–304). IEEE.

Burgard, W., Moors, M., Fox, D., Simmons, R., & Thrun, S. (2000) Collaborative multi-robot exploration. In *Proceedings of the IEEE international conference on robotics and automation (ICRA'00)* (Vol. 1, pp. 476–481). IEEE.

Burri, M., Nikolic, J., Hurzeler, C., Caprari, G., & Siegwart, R. (2012). Aerial service robots for visual inspection of thermal power plant boiler systems. In *2012 2nd international conference on applied robotics for the power industry (CARPI)* (pp. 70–75).

Camacho, E. F., & Bordons, C. (2003). *Model Predictive Control*. Berlin: Springer.

Chin, R. T., & Harlow, C. A. (1982). Automated visual inspection: A survey. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *6*, 557–573.

Choset, H., Pignon, P. (1998). Coverage path planning: The boustrophedon cellular decomposition. In *Field and service robotics* (pp. 203–209). Springer

Colas, F., Mahesh, S., Pomerleau, F., Liu, M., & Siegwart, R.(2013). 3d path planning and execution for search and rescue ground robots. In *2013 IEEE/RSJ international conference on intelligent robots and systems (IROS)* (pp. 722–727). IEEE.

Connolly, C., et al. (1985). The determination of next best views. In *Proceedings of the 1985 IEEE international conference on robotics and automation*, (Vol. 2, pp. 432–435). IEEE.

Dewan, A., Mahendran, A., Soni, N., & Krishna, M.(2013). Heterogeneous UGV-MAV exploration using integer programming. In *2013 IEEE/RSJ international conference on intelligent robots and systems (IROS)* (pp. 5742–5749). IEEE.

Dornhege, C., & Kleiner, A. (2013). A frontier-void-based approach for autonomous exploration in 3d. *Advanced Robotics*, *27*(6), 459–468.

Faigl, J., & Kulich, M.(2015) On benchmarking of frontier-based multi-robot exploration strategies. In *2015 european conference on mobile robots (ECMR)* (pp. 1–8). IEEE.

Faigl, J., Kulic, M., & Přeučil, L. (2012). Goal assignment using distance cost in multi-robot exploration. In *2012 IEEE/RSJ international conference on intelligent robots and systems (IROS)* (pp. 3741–3746). IEEE.

Galceran, E., & Carreras, M. (2013). A survey on coverage path planning for robotics. *Robotics and Autonomous Systems*, *61*(12), 1258–1276.

González-Banos, H. H., & Latombe, J.-C. (2002). Navigation strategies for exploring indoor environments. *The International Journal of Robotics Research*, *21*(10–11), 829–848.

Heng, L., Gotovos, A., Krause, A., & Pollefeys, M. (2015). Efficient visual exploration and coverage with a micro aerial vehicle in unknown environments. In *IEEE international conference on robotics and automation (ICRA)*.

Hornung, A., Wurm, K. M., Bennewitz, M., Stachniss, C., & Burgard, W. (2013). Octomap: An efficient probabilistic 3d mapping framework based on octrees. *Autonomous Robots*, *34*(3), 189–206.

Hover, F. S., Eustice, R. M., Kim, A., Englot, B., Johannsson, H., Kaess, M., & Leonard, J. J.(2012). Advanced perception, navigation and planning for autonomous in-water ship hull inspection. *The International Journal of Robotics Research,31*(12), 1445–1464.

Howard, A., Parker, L. E., & Sukhatme, G. S. (2006). Experiments with a large heterogeneous mobile robot team: Exploration, mapping, deployment and detection. *The International Journal of Robotics Research*, *25*(5–6), 431–447.

Janoušek, P., & Faigl, J. (2013). Speeding up coverage queries in 3D multi-goal path planning. In *2013 IEEE international conference on robotics and automation (ICRA)* (Vol. 1. IEEE, pp. 5082–5087).

Karaman, S., & Frazzoli, E. (2011). Sampling-based algorithms for optimal motion planning. *The International Journal of Robotics Research*, *30*(7), 846–894.

Khanna, R., Moller, M., Pfeifer, J., Liebisch, F., Walter, A., & Siegwart, R. (2015). Beyond point clouds-3d mapping and field parameter measurements using uavs. In *2015 IEEE 20th conference on emerging technologies & factory automation (ETFA)* (pp. 1–4). IEEE.

Kuipers, B., & Byun, Y.-T. (1991). A robot exploration and mapping strategy based on a semantic hierarchy of spatial representations. *Robotics and Autonomous Systems*, *8*(1), 47–63.

LaValle, S. M.(1998) Rapidly-exploring random trees a new tool for path planning.

Li, A. Q., Amigoni, F., & Basilico, N. (2012). Searching for optimal off-line exploration paths in grid environments for a robot with limited visibility. *AAAI*.

Lynen, S., Achtelik, M. W., Weiss, S., Chli, M., & Siegwart, R. (2013). A robust and modular multi-sensor fusion approach applied to mav navigation. In *2013 IEEE/RSJ international conference on intelligent robots and systems (IROS)* (pp. 3923–3929). IEEE.

Nikolic, J., Burri, M., Rehder, J., Leutenegger, S., Huerzeler, C., & Siegwart, R. (2013). A UAV system for inspection of industrial facilities. In *IEEE aerospace conference*.

Omari, S., Gohl, P., Burri, M., Achtelik, M., & Siegwart, R. (2014). Visual industrial inspection using aerial robots. In *2014 3rd international conference on applied robotics for the power industry (CARPI)* (pp. 1–5). IEEE.

Papadopoulos, G., Kurniawati, H., Patrikalakis, N. M.(2013). Asymptotically optimal inspection planning using systems with differential constraints. In *2013 IEEE international conference on robotics and automation (ICRA)* (pp. 4126–4133). IEEE.

Pito, R. (1999). A solution to the next best view problem for automated surface acquisition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *21*(10), 1016–1030.

Rosenblatt, J., Williams, S., & Durrant-Whyte, H. (2002). A behavior-based architecture for autonomous underwater exploration. *Information Sciences*, *145*(1), 69–87.

Surmann, H., Nüchter, A., & Hertzberg, J. (2003). An autonomous mobile robot with a 3d laser range finder for 3d exploration and digitalization of indoor environments. *Robotics and Autonomous Systems*, *45*(3), 181–198.

Vasquez-Gomez, J. I., Sucar, L. E., Murrieta-Cid, R., & Lopez-Damian, E. (2014). Volumetric next best view planning for 3d object reconstruction with positioning error. *International Journal of Advanced Robotic Systems*, *11*, 159.

Whaite, P., & Ferrie, F. P. (1997). Autonomous exploration: Driven by uncertainty. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *19*(3), 193–205.

Yamauchi, B.(1997). A frontier-based approach for autonomous exploration. In *Proceedings of the 1997 IEEE international symposium on computational intelligence in robotics and automation (CIRA'97)* (pp. 146–151). IEEE.

Yoder, L., & Scherer, S. (2016). Autonomous exploration for infrastructure modeling with a micro aerial vehicle. In *Field and service robotics* (pp. 427–440). Springer.

Zlot, R., & Bosse, M. (2014). Efficient large-scale three-dimensional mobile mapping for underground mines. *Journal of Field Robotics*, *31*(5), 758–779.

Zlot, R., & Stentz, A. (2006). Market-based multirobot coordination for complex tasks. *The International Journal of Robotics Research*, *25*(1), 73–101.
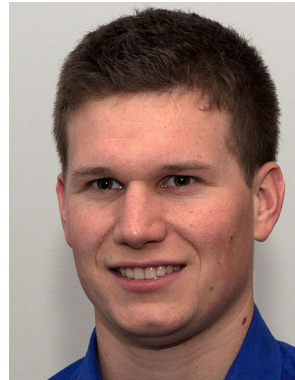
**Andreas Bircher** is software developer at Wingtra aerial robotics and a research assistant at the Autonomous Systems Lab at ETH Zurich, headed by Prof. Dr Roland Siegwart. He studied at ETH Zurich, where he received his Masters diploma in Robotics, Systems & Control in 2014. Since, his research focuses on the area of path planning for autonomous robots. Recent results cover inspection and exploration path planning for aerial robots.
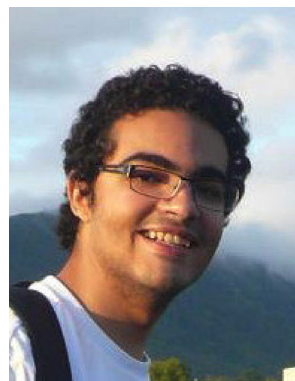


**Mina Kamel** received his bachelor degree with honor in automation engineering from the Politecnico di Milano in 2012 with a scholarship from the Italian government. Afterwards he received a masters degree with distinction in Robotics, Systems and Control from ETH Zurich in 2014. During his master thesis, he contributed to the aerial construction project in the Flying Machine Arena, developing a tensile structures simulator and estimator. Since then, Mina has been working as a Ph.D. Candidate at the Autonomous Systems Lab at ETH Zurich. His research focuses on achieving advanced levels of navigational and operational autonomy for aerial robots equipped with aerial manipulation capabilities.

**Kostas Alexis** is currently an Assist. Professor at the University of Nevada, Reno developing his lab activities towards advanced robotics autonomy and broad integration and utilization of aerial robotics. Formerly, he had been working as a Senior Researcher at the Autonomous Systems Lab of ETH Zurich where, among others, he participated in the research activities of the lab for advanced control and superior autonomy for Micro Aerial Vehicles, as well as in the efforts to design and develop the AtlantikSolar Solar-powered Unmanned Aerial Vehicle. Dr. Alexis' research interests encompass the fields of control, motion planning and autonomous navigation systems. He is the author or co-author of more than 60 publications -many of them awarded. Furthermore, he has organized or participated in several large-scale multi-million research projects with broad international involvement and collaboration. Dr. Alexis received his Ph.D. in the field of aerial robotics control and collaboration from the University of Patras, Greece in 2011.

**Helen Oleynikova** received a Bachelor in Robotics Engineering from Olin College of Engineering in 2011, and a Masters in Robotics, Systems and Control from ETH Zurich in 2015. In between, she also worked as a Software Engineer at Google on StreetView on detection and recognition algorithms. She is currently pursuing a Ph.D. in Mechanical Engineering at the Autonomous Systems Lab at ETH Zurich, where she focuses on mapping and obstacle avoidance.

**Roland Siegwart** is full professor for autonomous systems at ETH Zurich since July 2006. He has a Diploma in Mechanical Engineering (1983) and Ph.D. in Mechatronics (1989) from ETH Zurich. In 1989/90 he spent one year as postdoctoral fellow at Stanford University. After that he worked part time as R&D director at MECOS Traxler AG and as lecturer and deputy head at the Institute of Robotics, ETH Zürich. In 1996 he was appointed as associate and later full professor for autonomous microsystems and robots at the Ecole Polytechnique Fédérale de Lausanne (EPFL). During his period at EPFL he was Deputy Head of the National Competence Center for Research (NCCR) on Multimodal Information Management (IM2), co-initiator and founding Chairman of Space Center EPFL and Vice Dean of the School of Engineering. In 2005 he hold a visiting position at NASA Ames and Stanford University. Roland Siegwart is member of the Swiss Academy of Engineering Sciences and board member of the European Network of Robotics (EURON). He served as Vice President for Technical Activities (2004/05) and is currently Distinguished Lecturer (2006/07) and AdCom Member (2007–2009) of the IEEE Robotics and Automation Society. He is member of the "Bewilligungsausschuss Exzellenzinitiative" of the "Deutsche Forschungsgemeinschaft (DFG)". He is coordinator of two European projects and co-founder of several spin-off companies.