
An incremental sampling-based approach to inspection planning: the rapidly exploring random tree of trees

Andreas Bircher, Kostas Alexis, Ulrich Schwesinger, Sammy Omari, Michael Burri and Roland Siegwart

Robotica / *FirstView* Article / April 2016, pp 1 - 14
DOI: 10.1017/S0263574716000084, Published online: 11 March 2016

Robotica: page 1 of 14. © Cambridge University Press 2016
doi:10.1017/S0263574716000084

An incremental sampling-based approach to inspection planning: the rapidly exploring random tree of trees

Andreas Bircher†, Kostas Alexis‡*, Ulrich Schwesinger†, Sammy Omari†, Michael Burri† and Roland Siegwart†

†*Autonomous Systems Lab at ETH Zurich, Leonhardstrasse 21, 8092 Zurich, Switzerland*
‡*University of Nevada, Reno, 1664 N. Virginia St., 89557, Reno, NV, US*

(Accepted January 25, 2016)

SUMMARY
A new algorithm, called rapidly exploring random tree of trees (RRTOT) is proposed, that aims to address the challenge of planning for autonomous structural inspection. Given a representation of a structure, a visibility model of an onboard sensor, an initial robot configuration and constraints, RRTOT computes inspection paths that provide full coverage. Sampling based techniques and a meta-tree structure consisting of multiple RRT* trees are employed to find admissible paths with decreasing cost. Using this approach, RRTOT does not suffer from the limitations of strategies that separate the inspection path planning problem into that of finding the minimum set of observation points and only afterwards compute the best possible path among them. Analysis is provided on the capability of RRTOT to find admissible solutions that, in the limit case, approach the optimal one. The algorithm is evaluated in both simulation and experimental studies. An unmanned rotorcraft equipped with a vision sensor was utilized as the experimental platform and validation of the achieved inspection properties was performed using **3**D reconstruction techniques.

KEYWORDS: Inspection path planning; Coverage path planning; Aerial robotics; Unmanned aerial vehicles.

1. Introduction

The vision of autonomous structural inspection operations has always inspired the research community due to its great potential in real-life applications as well as the scientific challenges it poses. Nowadays, addressing this problem efficiently gains further interest as robots have proven their capabilities in information gathering. Either in the form of unmanned ground,[1] aerial[2] or even underwater[3] vehicles and utilizing vision,[2,4,5] LiDARs[1] or other sensing devices, robots are capable of perceiving the environment and acquiring data remotely, safely and with low cost. As infrastructure ages and extends, and as the potential safety threats grow, the need for precise, frequent and time-efficient structural inspection has increased.

To address the challenges of autonomous inspection operations, a robot has to be able to compute an efficient path that results in full coverage of the structure to be inspected. This problem belongs to the class of coverage problems and despite the interest it has attracted, its inherent complications still limit the performance of the proposed solutions.

The survey in ref. [6] and the references therein provide a comprehensive overview of coverage planning methods. Some methods either simplify the inspection structure or the planning space via

* Corresponding author. E-mail: kalexis@unr.edu
E-mails: andreas.bircher@mavt.ethz.ch, ulrich.schwesinger@mavt.ethz.ch, sammy.omari@mavt.ethz.ch, michael. burri@mavt.ethz.ch, rsiegwart@ethz.ch

cell decomposition[7,8] or grid-based schemes.[9] More versatile approaches that can handle arbitrary 3D structures divide the global inspection path planning problem into two NP-hard subproblems, namely the art-gallery problem (AGP)[10] and the traveling salesman problem (TSP).[11] Given a visibility model, an AGP solver finds the minimal set of observation points that the robot has to visit to guarantee full coverage and subsequently a TSP solver computes the shortest route among them. The authors in refs.[3,12–14] present advanced methods that follow the separated approach. Despite their notable performance, such decoupled approaches suffer from specific limitations. First of all, they are unable to guarantee feasibility of the exact solution in case of non-holonomic constraints and cluttered environments since the points generated by the AGP solver might turn out to be unreachable from other observation points on the set. Moreover, even when a feasible path that ensures full coverage is computed, its optimality is in general not ensured due to the decoupled two-step optimization. This is because the AGP solver ignores the resulting cost of connecting the selected points. In order to alleviate these limitations in the inspection planning for complex 3D structures, new algorithms that do not separate the problem into the AGP and TSP but rather aim to compute the observation points and the path simultaneously have to be proposed. Among the first, the work in ref. [15] made a significant step towards this direction. However, its graph expansion based on sampling in the whole control space followed by forward simulations does support multiple vehicle configurations but comes at the inherent cost of computation time. Since many systems relevant to inspection (e.g. aerial robots with rotorcraft configurations) feature an explicit Boundary Value Solver (BVS), one can take advantage of recent results for point-to-point path planning like PRM* or RRT*[16] and design a fast algorithm that benefits from this fact. Following a rather hybrid approach, previous contribution of the authors aimed to make a step towards iteratively optimized (but not guaranteed optimal) solutions that retain a low computational load.[4,5] Ideally, one would desire to be able to utilize an inspection solver that can provide results with guaranteed (and if possible optimal) quality, at a computational load that allows its flexible utilization for problems of reasonable and realistic scale and complexity. Essentially, this calls for algorithms that encompass the capacity to approach the optimal solution, while at the same time being characterized by the property of computing first solutions extremely fast, and subsequently having the capacity to iteratively improve them.

Within this work, a new approach for full coverage, optimal inspection path planning, called RRTOT is proposed. For systems with a BVS, it guarantees to find inspection paths that give full coverage while, in the limit case of infinite iterations, it approaches the optimal solution. The RRTOT does not separate the problem into the AGP and TSP but rather employs a sampling-based motion-planning paradigm that computes and optimizes coverage solutions. Due to the nature of the inspection problem and in order to approach it in a unified way, a novel scheme is employed that grows a meta–tree of subtrees where each one of them expands through the whole configuration space. Algorithms using multiple trees are known from point-to-point path planning[17–19] to solve problems of parallelization, narrow passages or multi-goal path planning. However, to the best of our knowledge, it is the first time that such a meta-tree/subtree structure is employed to unify the optimization for good observation points and short paths for the coverage problem. While this structure gets incrementally built, admissible paths that provide full coverage are found. At the same time, the BVS enables fast tree-based optimization of existing paths which also reduces the time to high-quality solutions compared to ref. [15] for all these cases that a BVS is available. Vehicle limitations such as non-holonomic constraints are enforced by the use of the BVS and the resulting algorithm can be applied to all systems for which such a BVS exists.

The RRTOT algorithm is evaluated in both simulation and experimental studies for holonomic as well as non-holonomic robotic systems. For the experimental evaluation, the Micro Aerial Vehicle (MAV) platform shown in Fig. 1 is utilized and equipped with a vision sensor. Complying with the model of the onboard sensor, 3D inspection paths are computed and provided as reference to the MAV flight control unit. The completeness of the inspection results is evaluated using 3D reconstruction algorithms . A video of one of the recorded experiments can be found online at `https://youtu.be/e7ljyDM9h8o`. In all cases, the RRTOT algorithm presents satisfactory performance and computes fast and full coverage results that indicate its potential as a planner for structural inspection. Overall, the proposed algorithm employs a novel iterative strategy to approach the optimal full coverage solution while guaranteeing collision-free navigation and respecting vehicle constraints and sensor limitations. This approach is in contrast with the typically employed methodology of splitting the inspection path planning problem into that of minimal set of viewpoints

Fig. 1. Instant of the RRTOT inspection path execution and derived 3D model.

computation and subsequent tour computation and path optimization. Thorough evaluation studies illustrate that such an approach not only is conceptually valuable due to its capacity to converge to the optimal solution but also practically applicable.

The remainder of this paper is structured as follows. In Section 2, the inspection problem is defined followed by the description of the proposed approach in Section 3. In Section 4, the feasibility and optimality properties of the solution are analyzed. Evaluation studies are presented in Section 5, while conclusions are drawn in Section 6.

2. Problem Definition

Consider a bounded δ-dimensional Euclidean space $X \subset \mathbb{R}^\delta$ ($\delta = 2$ or 3) consisting of the obstacle regions $X_{obs} \subset X$ and the resulting free space $X_{free} = X \setminus X_{obs}$. Similarly, Ξ_{free} is the free configuration space, that can be augmented with system specific states. Further, there exists a set of manifolds $M \subset X$ to be inspected. The system dynamics of the inspecting vehicle have the form $\xi_{i+1} = f(\xi_i, \mathbf{u}_i)$, where ξ is the configuration vector, $\mathbf{u}$ is the input and the initial condition is $\xi_0 = \xi_{init}$. Paths are defined as $\sigma : \mathbb{R} \to \xi$. Additionally, a visibility model is assumed, that defines the area $\mathcal{V}(\xi) \subseteq X$ that can be inspected from configuration ξ. If a manifold lies completely inside $\mathcal{V}(\xi)$, it is called visible from ξ. The inspection path planning problem consists of the triplet $(\Xi_{free}, \xi_{init}, M)$. Subsequently, the definitions of a feasible and an optimal inspection path planning problem are given following a terminology similar to the one used in ref. [16].

Problem 1 (Feasible Inspection Path Planning). Let the set of manifolds $M \subset X$ be the area to be inspected. Further, $A_i \subseteq M$, $\forall i \in [1, N]$, with $N \in \mathbb{N}^+$ and $\bigcup_{i=[1,2,\ldots N]} A_i = M$. Given an inspection path planning problem $(\Xi_{free}, \xi_{init}, M)$, find an *admissible path* $\sigma(s)$, $s \in [0, 1]$ such that $\sigma(0) = \xi_{init}$ and $\forall i \in [1, N] \exists s_i$ such that $\sigma(s_i)$ is a configuration from which A_i is visible. If such a path exists, the problem is feasible.

Problem 2 (Optimal Inspection Path Planning). Let Σ be the set of all admissible paths. Given a feasible inspection path planning problem $(\Xi_{free}, \xi_{init}, M)$ and a cost function $c : \Sigma \to \mathbb{R}_{\geq 0}$, find an admissible path σ^* such that $c(\sigma^*) = min_{\sigma \in \Sigma} c(\sigma)$.

3. Proposed Approach

A characteristic difference of point-to-point path planning and inspection (coverage) path planning is that in the latter, once a specific subset of the structure has been inspected, paths may turn towards a new target and follow this new general direction that can even be the same way back. Reflecting this fact and in order to make use of recent results for point-to-point path planning that allow rapid

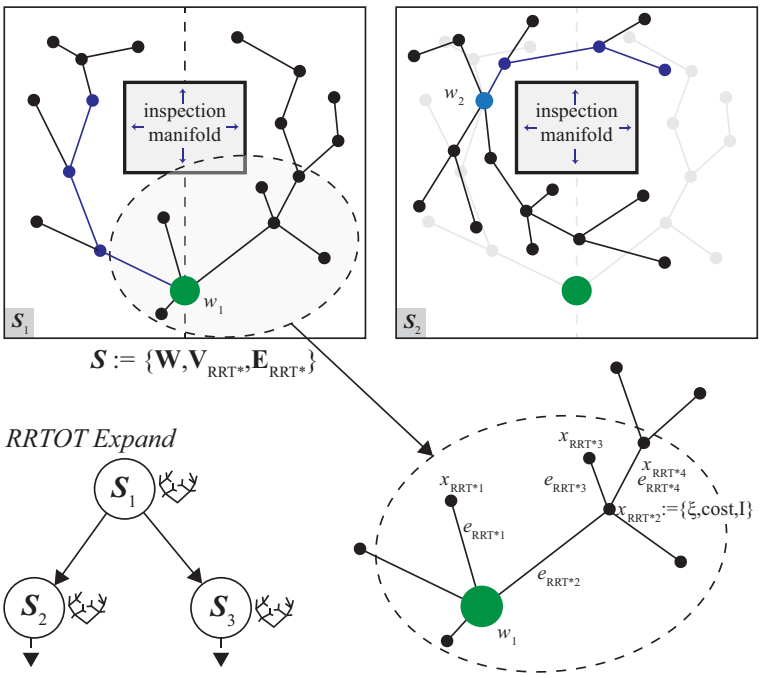


Fig. 2. Starting from ξ_{init}, the initial RRT* tree grows to cover Ξ_{free}. Of its vertices, the root configurations for new subtrees are randomly sampled. The new roots become waypoints in the new subtrees. By iteratively sampling new waypoints from existing RRT* trees, a TOT grows to find arbitrary complex paths.

exploration of the configuration space, our approach proposes a division of inspection paths into segments which follow a certain direction, that may change when switching to a new segment. From potential switching points, called *waypoints*, several path segments may branch. The resulting directed graph has a tree structure, which allows extracting solutions easily. On a lower level, each waypoint is a root of a RRT* tree, called subtree. Such a subtree spans the whole free configuration space. The end points of the next path segments are chosen from its vertices, effectively resulting in the choice of a new direction and distance to travel. This meta-tree, called tree of trees (TOT), is constructed randomly and incrementally, as outlined in Algorithm 1 and the according function descriptions. A key feature of the proposed algorithm is the optimization of existing paths (Algorithm 1, lines 9 and 12) while the TOT is constructed. Aforementioned structure allows for local cost optimization in the subtrees, while searching for full visibility paths can be mostly performed on the level of the TOT.

More formally, the TOT consists of its subtrees $S \in \mathbb{S}$, $S := \{\mathbb{W}^S, \mathbb{V}^S, \mathbb{E}^S\}$ ($\mathbb{S}$ represens the set of subtrees), consisting of waypoints $w \in \mathbb{W}^S$, vertices $x \in \mathbb{V}^S$ and edges $e \in \mathbb{E}^S$. Each vertex x is annotated with a configuration of the vehicle ξ, a cost and a set $I \subseteq M$ that is the set of the already inspected areas. It is the union of the visible areas along the path $I(\sigma(t)) = \bigcup_{s=[0,t]} \mathcal{V}(\sigma(s)) \cap M$. To construct these vertices and edges, the algorithm employs the RRT* method[16] that grows the subtrees in N_{RRT^*} iterations. Every subtree possesses its own RRT* planner that has no knowledge of the other subtrees. A new waypoint is chosen from the set of vertices of its respective parent subtree, to which it will be connected. This effectively maintains a global tree structure as shown in Fig. 2. To eventually achieve full coverage, the depth of the TOT grows with increasing iterations to find more complex paths that give higher visibility.

The annotation of vertices with the set of the already inspected part of the structure I enables the extraction of actual solutions. Every time a vertex reaches full visibility ($I = M$) and the cost is lower than the *best cost* so far, the path to this vertex is the new *best path* so far $\hat{\sigma}$. The latest best path is available as the output of the algorithm at any time.

Once a first solution is found, more effort is dedicated into improving existing solutions. As shown in Algorithm 1, at the end of every TOT-iteration, every subtree's RRT* planner is iterated for $N_{optimize}$ times. In order not to destroy existing admissible paths, the RRT* rewiring criterion has to be adapted: a vertex is only rewired, if besides decreasing the cost it does not lose any visibility,

that is $I \subseteq I'$, where I is the original and I' the new visibility. Additionally, the waypoint positions are also optimized in a rewiring step as shown in Algorithm 2. In this step, subtrees may accumulate multiple waypoints that are roots of the same subtree, that is then actually a forest (set of disjoint trees). In the optimization's rewiring step, two trees in the forest may get reconnected, effectively removing a root. Consequently, the waypoint that previously was this root is also removed.

3.1. Subroutines

The role of the employed functions within Algorithms 1 and 2 is briefly explained below:
NewSubtree(ξ, N_{RRT^*})**:** A standard RRT* tree with N_{RRT^*} iterations is grown, starting at ξ as root. If a vertex gets full visibility, $I = M$ and has lower cost from root (*CFR()*) than the best cost $c(\hat{\sigma})$, the path to the vertex is the new global best path $\hat{\sigma}$. The output is this new subtree S.
ExpandSubtree(S)**:** This function samples a vertex at random from $\mathbb{V}^S$ to use its configuration to initialize a new subtree (NewSubtree()). The same vertex can only be sampled once.
OptimizeSubtree(S, $N_{optimize}$)**:** This routine iterates the RRT* planner of the subtree S for $N_{optimize}$ iterations. The condition in the rewiring step is augmented with the requirement, that the near vertices must not lose visibility.
Near(S, x)**:** The output of this function is the set of all vertices in $\mathbb{V}^S$ that are within a distance d of x according to some distance metric using the vertices configurations. This distance d scales with $\frac{log(n)}{n}$, where n is the number of vertices in $\mathbb{V}^S$. The used distance threshold is the same as for the RRT* Near() function which is detailed in ref. [16].

Algorithm 1: This algorithm presents the body of the RRTOT inspection strategy. It iteratively builds the TOT and as soon as a first solution is found, optimizes existing paths at the end of every iteration. The functions *NewSubtree*, *OptimizeSubtree* and *RewireWaypoint* update the globally best solution $\hat{\sigma}$.

$\mathbb{S} \leftarrow S_{init} \leftarrow$ NewSubtree(ξ_{init}, N_{RRT^*})
$N_{\mathrm{TOT}} \leftarrow 0$
while execute **do**
  for all $S \in \mathbb{S}$ **do**
    $\mathbb{S} \leftarrow \mathbb{S} \cup$ ExpandSubtree(S)
  end for
  if first admissible path found **then**
    for all $S \in \mathbb{S}$ **do**
      OptimizeSubtree(S, $N_{optimize}$)
    end for
    for all $S \in \mathbb{S} \setminus S_{init}$ **do**
      RewireWaypoint(S)
    end for
  end if
  $N_{\mathrm{TOT}} \leftarrow N_{\mathrm{TOT}} + 1$
end while
return best admissible path $\hat{\sigma}$

ObstacleFree(x_i, x_j)**:** This function returns true, if the path from ξ_{x_i} of x_i to ξ_{x_j} of x_j lies entirely in Ξ_{free}. The path is subject to the encoded vehicle constraints.
CFR(x)**:** The CFR function returns the cost of moving from ξ_{init} to ξ_x of x. The path considered for the cost calculation runs along the edges of the TOT and its subtrees.
VFR(x)**:** The Visibility From Root (VFR) function returns the union of the visibility along the edges of the TOT and its subtrees from ξ_{init} to ξ_x, the configuration of x, which corresponds to I of x.
Cost(x_i, x_j)**:** The cost of moving (typically time-to-travel) from one vertex x_i to another, x_j is returned. The connecting path is subject to the vertices' configurations ξ_i and ξ_j and the system constraints.
Visibility(x_i, x_j)**:** The function returns the set of visible manifolds when moving from x_i to x_j, $\bigcup_{s=[0,1]} \mathcal{V}(\sigma(s)) \cap M$, with $\sigma(0) = \xi_i$, $\sigma(1) = \xi_j$, where ξ_i, ξ_j are the configurations corresponding

to vertices x_i, x_j respectively. Explicit solutions for the computation may exist for some systems, otherwise discrete sampling of the configurations along the path can be used as an approximation.
Parent(x)**:** Returns the parent vertex of x. Note that this may be part of a different subtree.

Algorithm 2 *RewireWaypoint*(S)**:** This algorithm presents the routine, that performs a rewiring step for the waypoints. It improves the paths in these regions, which are not affected by the optimization of the *OptimizeSubtree* function. It is highlighted that a check that confirms weather rewiring of the x_{child} with x_{near} would result in a lower cost compared to the current cost from root, while none of the VFR is lost (but could potentially be complemented by additional visibility) is performed. Only if this is the case and the edge is collision free, then the rewiring step is performed.

1: $\mathbb{V}_{children} \leftarrow$ Get child vertices of all waypoints $\mathbb{W}^S$
2: **for** *all* $x_{child} \in \mathbb{V}_{children}$ **do**
3: $S_{parent} \leftarrow$ Get parent subtree of S;
4: $\mathbb{V}_{near} \leftarrow$ Near(S_{parent}, x_{child});
5: **for** *all* $x_{near} \in \mathbb{V}_{near}$ **do**
6: **if** ObstacleFree(x_{near}, x_{child}) & CFR$(x_{child}) \geq$ CFR(x_{near}) + Cost(x_{near}, x_{child}) [4] & VFR$(x_{child}) \subseteq$ VFR$(x_{near}) \cup$ Visibility(x_{near}, x_{child}) **then**
7: $\mathbb{W}^S \leftarrow \mathbb{W}^S \cup \{x_{child}\}$;
8: $x_{parent} \leftarrow$ Parent(x_{child});
9: $\mathbb{E}^S \leftarrow \mathbb{E}^S \setminus \{(x_{parent}, x_{child})\}$;
10: $\mathbb{E}^S \leftarrow \mathbb{E}^S \cup \{(x_{near}, x_{child})\}$;
11: VFR$(x_{child}) \leftarrow$VFR$(x_{near}) \cup$ Visibility(x_{near}, x_{child});
12: **if** VFR$(x_{child}) = M$ & $c(\hat{\sigma}) >$ CFR(x_{child})
13: $\hat{\sigma} \leftarrow$ GetPath(x_{child});
14: **end if**
15: **end if**
16: **end for**
17: **end for**

GetPath(x)**:** This function reconstructs the path along the edges of the TOT and its subtrees from ξ_{init} to ξ_x, the configuration of x. This path σ is returned.

3.2. Parameters

The proposed algorithm employs a small set of tuning parameters, an aspect particularly important for its straightforward utilization in real-life operations. These are briefly defined below:

N_{RRT^*}: Defines the number of iterations on the level of the RRT* planner in the subtrees. It is relevant for the initial computation of the tree. A high number generally results in a slower first solution with a better quality, while a lower number of iterations increases the chance of a fast first solution. In either way, it should be chosen high enough to cover the whole Ξ_{free}.
N_{optimize}: Defines the number of iterations on the subtree-level for the optimization step (Algorithm 1, line 9). Typically, smaller values than N_{RRT^*} are used.

4. Analysis

This section deals with the formal analysis of the full coverage feasibility and inspection path optimality characteristics of the proposed approach. Within this analysis, specific assumptions are made and firstly that the RRTOT uses RRT* planners without limitation of the edge length, thus connecting with a newly sampled vertex exactly. In addition, several assumptions about the problem setup have to be made for the analysis but also for the algorithm. Most importantly, the requirements for the RRT* algorithm[16] have to be fulfilled, namely the continuity and additivity of the cost function. In addition, the following assumptions are made:

Assumption 1 (Uniform Sampling). The sampling procedure is such that the samples are drawn from an absolutely continuous and uniform distribution on Ξ_{free}.

Assumption 2 (Initial RRT* Iterations). It is assumed, that the number of initial iterations N_{RRT^*} is not small, such that samples in regions that cannot be connected directly to root because of obstacles can be connected via previously sampled vertices and have a probability to exist after N_{RRT^*} iterations that is arbitrarily close to the obstacle free case. In addition, the largeness of N_{RRT^*} is such, that when sampling new waypoints from the vertices, the samples can be assumed independent.

Assumption 3 (Inspectability). Each piece of inspection structure $\{A_1, A_2, \ldots A_N\}$ has a nonempty open area $B(A_i) \subseteq \Xi_{free}$, called inspectability area, from which it can be inspected. That is $\mu(B(A_i)) > 0$, where $\mu(\Xi)$ denotes the Lebesgue-measure.

The aforementioned strong assumptions are necessary for the formal proofs that follow. It is however worth mentioning, that as the experimental sections will show, the algorithm behaves in a very efficient way even when aspects of these assumptions cannot easily be proven to hold in an exact sense for the experimental data, the limitations of the computational system and the constraints of the vision sensor.

4.1. Feasibility of the algorithm

Subsequently, we present the proof, that the RRTOT algorithm finds an *admissible path* with a probability that *asymptotically converges to* 1 as $N_{RRT^*} \to \infty$. In a first step, it is shown, that long enough paths are found and afterwards a lower bound for the probability to find a solution is established, that converges to 1 as the number of waypoints in a path increases.

Theorem 1 (Existence of Collision-free Long Paths). As the number of iterations $N_{TOT} \to \infty$, so does the number of waypoints r in the longest sequence of consecutive waypoints P for a feasible problem.

Proof. As the number of iterations $N_{TOT} \to \infty$, the root waypoint $w_1(\xi_{init})$ has infinite children waypoints, since in every iteration one is added. For a feasible problem, in each iteration N_{TOT} the probability of sampling a configuration $\xi \to w_2$ that, immediately or later, results in collision is $p_{N_{TOT}} < 1$ and the probability of not finding a collision-free w_2 decays to zero as $N_{TOT} \to \infty$, $\prod_{i=1}^{\infty} p_i = 0$. The same holds for any waypoints w_k and w_{k+1}, $k \in \mathbb{N}^+$. From $r > k$ follows $r \to \infty$. □

Definition 1 ((α, β)-expansiveness[20]). Let $\mathcal{R}(p)$ and $\mathcal{R}_l(p)$ be the reachable set and the locally reachable set that is reachable in one expand step, respectively. The reachability of a set $Q \subset \Xi_{free}$ is $\mathcal{R}(Q) = \bigcup_{p \in Q} \mathcal{R}(p)$ and the local reachabilty of Q is $\mathcal{R}_l(Q) = \bigcup_{p \in Q} \mathcal{R}_l(p)$. Let β be a constant in $(0, 1]$, then the β-lookout of the set Q is defined as β-LOOKOUT$(Q) = \{p \in Q \mid \mu(\mathcal{R}_l(p) \setminus Q) \geq \beta\mu(\mathcal{R}(Q) \setminus Q)\}$. Let α be a constant in $(0, 1]$. The free space Ξ_{free} is (α, β)-expansive if for every point $p \in \Xi_{free}$ and every subset $Q \subset \mathcal{R}(p)$, it holds that $\mu(\beta\text{-LOOKOUT}(Q)) \geq \alpha\mu(Q)$.

In the RRTOT algorithm, a new waypoint can be sampled anywhere in Ξ_{free} without limitations by Assumption 2. Therefore, the (local) reachability of the system is always the whole free space Ξ_{free}. Thus, for the RRTOT algorithm the free space is (α, β)-expansive, in particular $\alpha = \beta = 1$.

Theorem 2 (ref. [20]). Let Ξ_{free} be (α, β)-expansive, $g_i > 0$ be the volume of the inspectability area $B(A_i)$ and γ_i be a constant in $(0, 1]$. A sequence P of r waypoints contains a waypoint in $B(A_i)$ with probability at least $1 - \gamma_i$, if

$$r \geq (k_i/\alpha) ln(2k_i/\gamma_i) + (2/g_i) ln(2/\gamma_i) = r_{min,i}, \tag{1}$$

where $k_i = (1/\beta) ln(2/g_i)$.

The main idea for the proof of Theorem 2[20] is to calculate the probabilistic overlap of the inspectability area with the local reachability set of the sequence P. This gives the probability to sample a waypoint in the next iteration that lies inside $B(A_i)$. The expression in Eq. (1) gives the bound on that probability. This is extendable to multiple inspectability areas that are visited in a row. Doing so provides the result required.

Corollary 1 (to Theorem 2). A sequence P of r waypoints contains a waypoint in each $B(A_i)$, $\forall i \in \{1, 2, \ldots N\}$ with probability at least $\prod_{i=1}^{N}(1 - \gamma_i)$ if $r \geq \sum_{i=1}^{N} r_{min,i}$. Moreover, if the number

of iterations $N_{TOT} \to \infty$, then $r \to \infty$ as shown in Theorem 1. Suppose P is partitioned such that $r_{min,i} = r_{min,j}$, $i \neq j$ and the length of the subsequences goes to infinity as well. All γ_i decay to zero exponentially fast and an admissible path for problem 1 is found *almost surely*, that is $\lim_{r\to\infty} \prod_{i=1}^{N}(1-\gamma_i) = 1$. Therefore, RRTOT finds an *admissible path* with probability that converges to 1.

The key parts for showing feasibility of the algorithm were the relation of the lower bound of the probability to find a solution with the length of a sequence given by Theorem 2. Theorem 1 guarantees, that long enough sequences will actually be generated for the probability to converge to 1. The proof of finding full coverage solutions corresponds to the most fundamental property of the proposed algorithm.

4.2. Asymptotic optimality

Asymptotic optimality is a particularly challenging property for inspection path planning algorithms. Within this section, it is shown that a *baseline* version of the RRTOT algorithm turns out to express such a critical property. More specifically, the algorithm considered for the analysis of asymptotic optimality excludes the waypoint rewiring function (Algorithm 1, line 12) and is called the *basic-RRTOT*. Experience gathered in simulation and experiments showed that the waypoint rewiring process only benefits the algorithm evolution towards good solutions due to its intrinsic criterion (Algorithm 2, line 6). In general, solutions of the *basic-RRTOT* are observed to be therefore equal or worse than the real RRTOT's solutions that has been executed for the same amount of iterations. However, formal analysis of asymptotic optimality is only provided for the *basic*-RROT, a structurally simplified algorithm that can be used for theoretical comparisons and evaluation of other methods. In particular and under specific assumptions, it is shown that in the limit of $N_{TOT} \to \infty$, the set of admissible paths found by the *basic-RRTOT* are probabilistically equivalent to the solutions found by an idealized algorithm that iteratively executes an *ideal-Expand* function (Algorithm 3) on a tree consisting of waypoints and edges. While expanding, the algorithm keeps track of the inspected part of the structure for each waypoint. For this algorithm, the convergence to the optimal solution is proven in ref. [15]. This proof shows that the algorithm can sample a series of waypoints arbitrarily close to the optimal path in the limit of infinite iterations. As the *basic-RRTOT* finds the same solutions also one arbitrary close to the optimal will be among them. The necessary assumptions for the comparison are:

Assumption 4 (ϵ-Inspectability). There exists a constant $\epsilon \in \mathbb{R}^+$ and a suboptimal path $\sigma'(s)$, $s \in [0, 1]$, that follows the optimal path σ^* arbitrarily close, and the respective costs are arbitrarily close, such that for every A_i $\exists s_i$ with $\mathcal{B}_{\sigma'(s_i),\epsilon} \subseteq B(A_i)$, where $\mathcal{B}_{\sigma'(s_i),\epsilon}$ is the ϵ-ball centered at $\sigma'(s_i)$.

Assumption 4 rules out cases, where convergence towards the optimum is impossible, because no path close to the optimal solution is admissible.

The modification of the RRT* rewire criterion for the optimization might in some cases prevent the RRT* planners from converging to optimal paths within the subtrees. Whether this is the case or not depends on the order of sampling of the vertices but not on the set of vertices. The following assumption is therefore reasonable and necessary for the analysis of convergence to hold:

Assumption 5 (Non-optimality of Modified RRT*). Let $\xi \in \Xi_{free}$ be a configuration where an RRT* tree is initialized. When choosing a new waypoint w among its vertices, there is a non-zero probability, independent of its configuration ξ_w, that the connection of ξ to ξ_w will converge to the optimal as the RRT* iterations go to infinity.

The following analysis considers only the subset of optimal subtrees $\mathbb{S}_o \subseteq \mathbb{S}$, whose waypoints are optimally connected according to assumption 5 and are not disjoint. By Assumption 5, $\mathbb{S}_o$ grows infinitely deep and wide, as $N_{TOT} \to \infty$.

Theorem 3 (Asymptotic optimality of RRTOT). Under assumption 5, the solutions of the *basic-RRTOT* algorithm are the same as the solutions of the algorithm with the ideal-expand algorithm proven to lead in optimal solutions as shown in ref. [15].

Proof. This proof shows that the steps performed by idealexpand are equivalent to the steps performed by *basic-RRTOT*, from which follows that the solutions will be probabilistically the same.

Algorithm 3: Ideal-Expand($\mathbb{W}$)[15]

1: Sample a waypoint w' uniformly at random from $\mathcal{R}_l(\mathbb{W})$.
2: Sample a waypoint w that can reach w'.
3: Compute a control input u to move from w to w'.
4: **return** w', w, $Trajectory(w, w', u)$, u

Since $\mathcal{R}_l(w) = \Xi_{free}, \forall w \in \mathbb{W} = \mathbb{W}_{S_1} \cup \mathbb{W}_{S_2} \cup \ldots$, the order of step 1 and step 2 in Algorithm 3 is irrelevant, because every existing waypoint w can be connected to any newly sampled w'. Step 2 in the ideal-expand is equivalent to choosing the subtree to expand in the *basic-RRTOT*. Ideal-expand assumes sampling from the waypoints uniformly at random over Ξ_{free}. Again since $\mathcal{R}_l(w) = \Xi_{free}$, the waypoints are uniformly random and independent samples. In one iteration, the set of all waypoints is the set to be expanded, thus guaranteeing uniform randomness and independence. On the other hand, the vertices in an RRT* tree are also uniformly randomly distributed on Ξ_{free} and independent. Therefore, sampling new waypoints from the large set of vertices $\mathbb{V}^S$ at random is equivalent to step 1 in ideal-expand. Step 3 is performed by the RRT* in the presented algorithm, that plans the path in the configuration space and under Assumption 5 for the $OptimizeSubtree()$ function ensures the asymptotic convergence of the edges of the TOT to the optimal. From the equivalence of the two algorithms follows the equivalence of their solutions in the limit of the number of iterations going to infinity and hence the same optimality properties. □

4.3. RRTOT fast inspection solutions

The complete RRTOT algorithm (including the waypoint rewiring step) relies on specific properties and steps to provide fast inspection solutions. This section deals with an overview of the key mechanisms that enable the fast computation of full coverage inspection solutions. First of all, the (1, 1) expansiveness of the algorithm on Ξ_{free} easily makes up for the computational burden of generating RRT* trees to connect the waypoints. Especially when planning with obstacles, it is beneficial to be able to plan with connections around them, the cost of which decreases by keeping iterating the RRT* planners in the $OptimizeSubtree()$ function. Furthermore, with the rewiring of the waypoints ($RewireWaypoints()$) optimization along the whole path is achieved. The waypoints are no longer fixed points on the path that may be suboptimal and therefore would prevent the specific path from converging to the optimal. Together with the fast expansion, these two functions, that improve existing paths, result in an algorithm that finds first solutions quickly, while having the tendency to optimize them. This is particularly useful for real-life operations, where a first path is preferred to be available in short.

5. Case Studies

In order to thoroughly evaluate the capabilities of the proposed RRTOT algorithm, several simulation and experimental studies were conducted. The results verify the functionality of the algorithm and its capabilities to compute efficient inspection paths in both 2D and 3D environments for holonomic or non-holonomic vehicles. In all cases, the objective is to provide full visibility while minimizing the distance. All calculations were performed on a computer with a 3.3 GHz processor running Ubuntu 12.04 and using a single-thread C++ implementation.

5.1. 2D simulation studies

In a 2D study, the RRTOT algorithm is evaluated in indirect comparison to the performance of the algorithm[15] using an identical simulation setup as the one employed by the authors of this work. The space configuration of this simulation setup is depicted in Figs. 3 and 4. As shown, it consists of five rectangles the edges of which have to be completely inspected while the overall distance of the inspection path gets minimized. Two cases are investigated, one considering a holonomic vehicle, while a non-holonomic vehicle with a minimum turning radius of 2 units and non-negative velocity is employed for the second case. For the holonomic case, the configuration vector is $\xi^{v,H} = [x, y]^T$ with initial conditions $\xi_0^{v,H} = [45, 0]^T$, while for the non-holonomic case, the configuration vector is augmented with the vehicle's heading angle ($\xi^{v,NH} = [x, y, \psi]^T$ and $\xi_0^{v,NH} = [45, 0, \pi/2]^T$).

Table I. RRTOT Performance analysis.

Holonomic	1–st sol.	$c(\hat{\sigma}) < 185$ units	$c(\hat{\sigma}) < 169$ units
$\bar{t}_H$	0.25 s	89 s	336 s
$std(\bar{t})_H$	0.27 s	189 s	247 s
Non-holonomic	1–st sol.	$c(\hat{\sigma}) < 185$ units	$c(\hat{\sigma}) < 177$ units
$\bar{t}_{NH}$	3.3 s	559 s	28 min
$std(\bar{t})_{NH}$	2 s	795 s	29 min

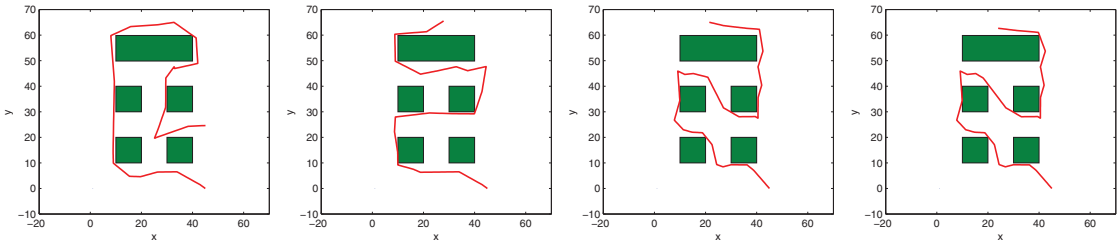


Fig. 3. A sample run of the RRTOT algorithm for the holonomic vehicle employing an omnidirectional sensor with a limited range of 15 units as only constraint ensuring inspection quality. The first result is returned quickly. The algorithm subsequently finds other paths belonging to different topologies. Figures 3c, 3d show, that a path is optimized within a certain topology. (a) Cost 196.4 in 1.1 s. (b) Cost 174.6 in 49.8 s. (c) Cost 166.7 in 140.3 s. (d) Cost 160.7 in 249.1 s.

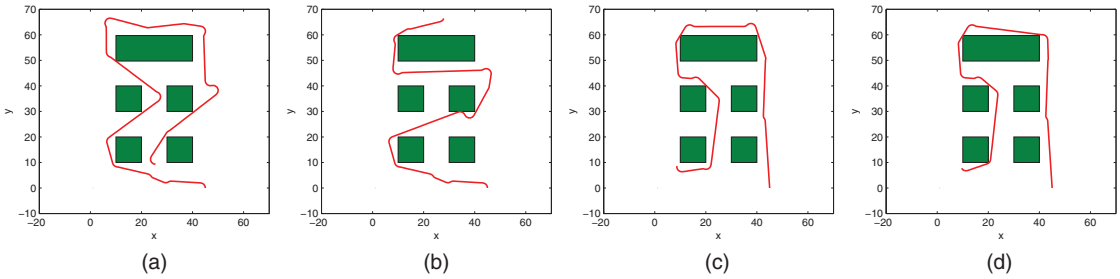


Fig. 4. A sample run of the RRTOT algorithm for the non-holonomic vehicle with a minimum turning radius of 2 units. It employs an omnidirectional sensor with a limited range of 15 units as only constraint ensuring inspection quality. The first result is returned quickly. The algorithm subsequently finds other paths belonging to different topologies. Figures 4c, 4d show, that a path is optimized within a certain topology. (a) Cost 230.6 in 2.8 s. (b) Cost 182.1 in 55.5 s. (c) Cost 176.4 in 305 s. (d) Cost 167.5 in 673 s.

Regarding the visibility model, both vehicles are considered to be equipped with an omnidirectional sensor with a range of 15 units, unless areas are occluded by parts of the structure.

Figures 3 and 4 show the computed paths in sample runs for the holonomic and non-holonomic case respectively. Within them, Figs. 3a and 4a show the first fast initial full coverage paths. In the course of computation, the algorithm comes up with better solutions, belonging to different topologies as shown in Figs. 3b, 3c and 4b, 4c. Figures 3d and 4d show that the paths are optimized within a certain topology to find solutions of decreasing cost while retaining full visibility.

To gain a statistical insight on the algorithm performance characteristics, the aforementioned process is repeated in total 20 times. The results are summarized in Table I, while the cost plots showing the convergence are depicted in Fig. 5. For further evaluation, the algorithm in ref. [15] finds, for the identical inspection problem setup, a first solution to the same non-holonomic problem in 12.87 s (standard deviation of 25.4 s) and high quality solutions with lengths of 190 and 185 units after 44 and 73 min respectively (with standard deviations of 53 and 85 min). It is highlighted once more that this comparison is indirect as the algorithm in ref. [15] can handle vehicle configurations that do not have a BVS and therefore is more versatile regarding the robot configuration. Nonetheless, it illustrates the good performance characteristics of RRTOT for all these cases of vehicles that a

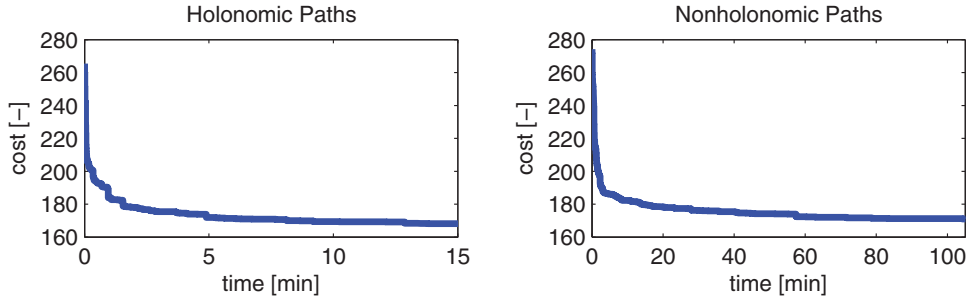


Fig. 5. Convergence curves of the 2D case study. Both curves show a fast first solution with subsequent steep convergence. For longer runtimes, the curve still exhibits decreasing cost.

BVS is available. As a very wide set of robots such as multirotor MAVs, most of the remotely operated underwater vehicles and other vastly used vehicle configurations are considered to have BVS describing their motion characteristics, RRTOT is believed to have a great potential for real-life applications.

5.2. 3D experimental studies

The experimental studies were conducted using an AscTec Firefly hexarotor MAV equipped with an Intel core duo computer and the visual–inertial sensor (VI-sensor) developed by the autonomous systems lab and Skybotix AG. The VIsensor integrates two HDR global shutter cameras (Aptina MT9V034) and an analog devices ADIS16448 IMU in a tightly aligned and synchronized way using an Artix^{TM}–7 FPGA, a Xilinx Zynq 7020 SoC module and an ATOM CPU running Linux. This integrated sensor system runs advanced image processing algorithms, provides complete pose estimates and builds a 3D map of the environment. An overview of the experimental platform is indicated in Fig. 6, while detailed information on the platform, the onboard controllers, the perception modules and the state estimation strategies are found in refs. [2,21–23]. For the experiments presented, a Vicon motion capture system provided exact position estimates and the Robot Operating System (ROS) was utilized as the middleware to allow solid interfacing of the RRTOT algorithm with the MAV. Precomputed paths are loaded to the MAV to be executed by the autopilot. As the recorded flight response is expected to naturally present a tracking error, for all the path computations the sensor FoV was considered slightly reduced compared to its actual value in order to provide some necessary robustness and avoid cases of not inspecting a subset of the desired structure. In general, as the proposed algorithm solves the problem of explicit inspection path planning, it naturally cannot *per se* with uncertainties regarding the model structure or inaccuracies of the sensor and vehicle model. To provide robustness, the main considerations of assuming a slightly reduced FoV and possibly more constrained non-holonomic constraints or slower maximum yaw rates are proposed.

The experimental setup refers to an area of 4 m $\times$ 4.5 m where multiple carton boxes are deployed and arranged as depicted in Fig. 7. Within this scenario, all non-overlapping surfaces of the boxes have to be inspected with minimal distance taking into account the limitations of the camera system (a single camera model is considered). Consequently, the visibility model is adapted to match the properties of the real sensor and therefore the field of view is angularly constrained on all four sides. From the nominal field of view a margin is deducted on all sides to account for errors of the inspection manifold model and the tracking of the path. A minimum and maximum distance to the object under inspection is also enquired to ensure good quality of inspection results. The orientation of the camera in the xy plane is an additional system state. Finally, although the employed vehicle can fly holonomic trajectories, the experiment is performed under both holonomic and non-holonomic considerations in order to evaluate the capabilities of the proposed inspection path planner. The encoded non-holonomic constraints are a minimum turning radius of $r_{\min} = 0.25$ m on the xy plane and first-order differentiability of the height. Such a setup is suitable for a multirotor MAV. With a bound on the height derivative and a larger radius constraint, it could also be adapted to plan paths for fixed-wing airplanes with a camera, movable around the z-axis. Figures 7a and 7b depict the offline computed paths for the holonomic and the non-holonomic case along with the experimentally

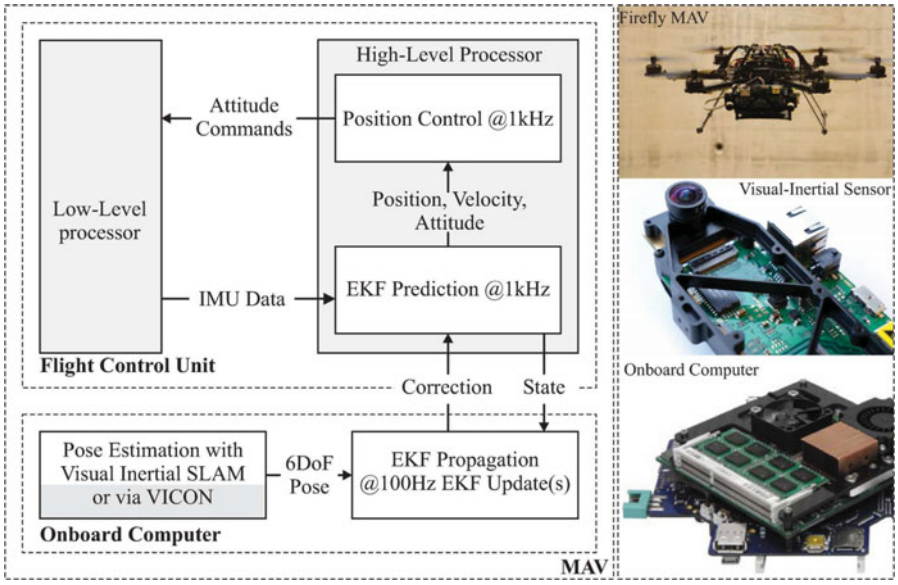


Fig. 6. Block diagram of the main algorithmical components of the flight control unit of the Firefly hexarotor, photos of the vehicle, the VI-Sensor and the high-level core duo processor. The lowlevel processor runs the attitude control of the vehicle which is commanded by the position controller running the high-level processor. The high-level unit also executes the state estimation algorithms and iterfaces the VICON pose feeds.

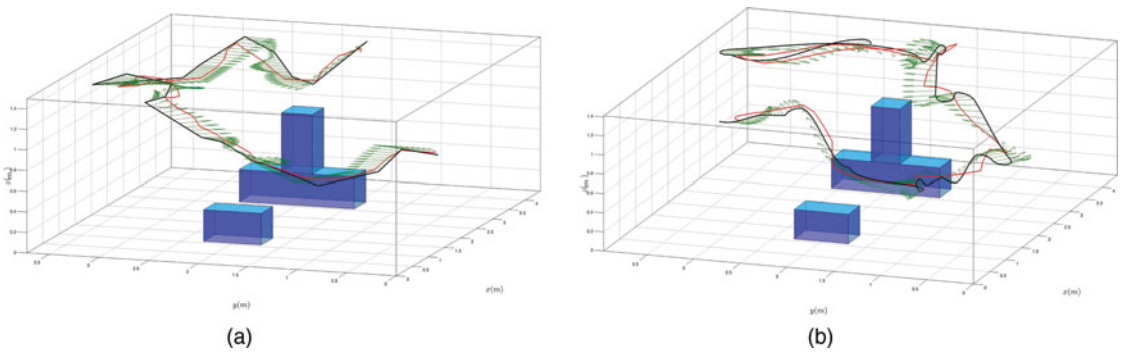


Fig. 7. Inspection paths computed by the RRTOT planner (black) and plots of the recorded vehicle response (red). The considered vehicle employs a vision sensor with a limited field of view and constraints on the sensing range for both minimal and maximal distance. The left plot corresponds to the holonomic case while the right plot presents the results when non-holonomic constraints are considered. These are a first order differentiability constraint on the height and a minimum turning radius of 0.25 m in the xy plane. The path length is 9.4 m and 13.1 m for the holonomic and non-holonomic cases respectively. (a) Holonomic case. (b) Non-holonomic case.

recorded MAV paths. Regarding the computation time, first feasible solutions are found within a few minutes, while finding a well optimized path takes in general significantly longer.

The presented inspection paths ensure full coverage of the desired 3D structure. This theoretical result was experimentally evaluated and validated by using the feeds of the onboard vision system and executing 3D reconstruction algorithms. Two reconstruction strategies were employed, namely (a) that of the free online tool 123D Catch[24] which uses images from only one of the onboard cameras to conduct the reconstruction process using computer vision algorithms, and (b) a stereo dense reconstruction pipeline that uses OctoMaps to represent the environment.[25] The results are shown in Fig. 8 and as can be seen complete reconstruction was achieved for both holonomic and non-holonomic cases. Note that a recorded response of the last experiment can be found in the video provided following this link: `https://youtu.be/e7ljyDM9h8o`.

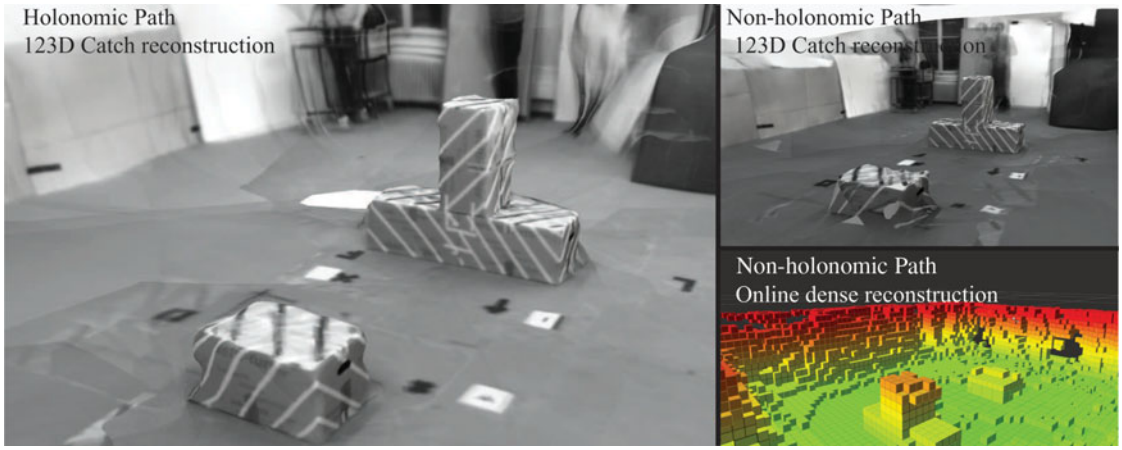


Fig. 8. 3D reconstruction results for the holonomic and non-holonomic paths using the offline 123D Catch tool as well as an online dense reconstruction pipeline that combines stereo block matching techniques fused with position information to derive a voxel-based representation. As shown, satisfactory reconstruction results are derived for both types of paths.

6. Summary & Conclusions

A new sampling-based inspection planning algorithm was presented within this work. The algorithm, the performance of which relies on a rewiring mechanism on the vertices of its tree structure, was described, followed by analysis on its capability to find admissible full coverage solutions with decreasing cost. The proposed RRTOT planner was evaluated in both simulation and experimental cases. Good inspection paths were found considering both, a holonomic and a particular case of non-holonomic vehicle with a turn radius limitation. The experimental case studies were conducted employing a hexacopter miniature aerial vehicle and an onboard vision module used for inspection. The derived experimental results and the corresponding 3D reconstructions of the inspected structures indicate the high quality of the computed solutions. Future work includes adaptations and experiments to plan for fixed-wing aircrafts as well as finding further heuristic concepts to speed up the computation while maintaining the theoretical optimality guarantees. Furthermore, one could consider employing the basic steps that are executed at each iteration of this algorithm and run them in an online fashion with the aim to iteratively explore unmapped areas (using real-time sensor data). In such a case, the goal would not be to guarantee global optimality but rather a locally optimized path to maximize the expected information gain.

Acknowledgment

This work has received funding from the European Union's Horizon 2020 Research and Innovation Programme under the Grant Agreement No.644128, AEROWORKS.

References

1. E. Stumm, A. Breitenmoser, F. Pomerleau, C. Pradalier and R. Siegwart, "Tensor-voting-based navigation for robotic inspection of 3d surfaces using lidar point clouds," **31**(12), 1465–1488 (2012).
2. M. Burri, J. Nikolic, C. Hurzeler, G. Caprari and R. Siegwart, "Aerial Service Robots for Visual Inspection of Thermal Power Plant Boiler Systems," *Applied Robotics for the Power Industry, 2012 2nd International Conference on* Zurich, Switzerland (2012) pp. 70–75.
3. B. Englot and F. S. Hover, "Three-dimensional coverage planning for an underwater inspection robot," **32**(9–10), 1048–1073 (2013).
4. A. Bircher, K. Alexis, M. Burri, P. Oettershagen, S. Omari, T. Mantel and R. Siegwart, "Structural Inspection Path Planning Via Iterative Viewpoint Resampling with Application to Aerial Robotics," *IEEE International Conference on Robotics and Automation (ICRA)*, Seattle, WA, USA (May 2015) pp. 6423–6430. [Online]. Available: `https://github.com/ethz-asl/StructuralInspectionPlanner`
5. A. Bircher, M. Kamel, K. Alexis, M. Burri, P. Oettershagen, S. Omari, T. Mantel and R. Siegwart, "Three-dimensional coverage path planning via viewpoint resampling and tour optimization for aerial robots," *Autonomous Robots*, Springer US, 1–25 (2015). DOI: 10.1007/s10514-015-9517-1, ISSN: 1573–7527.

6. E. Galceran and M. Carreras, Marc, "A survey on coverage path planning for robotics," *Robot. Auton. Syst.* **61**(12), 1258–1276 (2013).
7. J. Barraquand and J.-C. Latombe, "Robot motion planning: A distributed representation approach," *Int. J. Robot. Res.* **10**, 628–649 (1991).
8. H. Choset, "Coverage for robotics–a survey of recent results," *Ann. Math. Artif. Intell.* **31**(1–4), 113–126 (2001).
9. A. Zelinsky, R. A. Jarvis, J. Byrne and S. Yuta, "Planning Paths of Complete Coverage of an Unstructured Environment by a Mobile Robot," *Proceedings of International Conference on Advanced Robotics* vol. 13, Tsukuba (1993) pp. 533–538.
10. J. Urrutia, "Art Gallery and Illumination Problems," **In**: *Handbook of Computational Geometry*, Instituto de Mathematicas, Universidad Nacional Autonoma de Mexico: Mexico (2000) pp. 973–1027.
11. D. Shmoys, J. Lenstra, A. Kan and E. Lawler, *The Traveling Salesman Problem*. J. K. Lenstra, A. R. Kan, E. L. Lawler, D. B. Shmoys, editors. The traveling salesman problem: a guided tour of combinatorial optimization. John Wiley & Sons (1985).
12. B. Englot and F. S. Hover, "Sampling-Based Sweep Planning to Exploit Local Planarity in the Inspection of Complex 3d Structures," *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, IEEE, Vilamoura (2012) pp. 4456–4463.
13. T. Danner and L. E. Kavraki, "Randomized planning for short inspection paths," *IEEE International Conference in Robotics and Automation*, 2000, Apr 24, Vol. 2, pp. 971–976, San Francisco, CA, USA.
14. P. S. Blaer and P. K. Allen, "View planning and automated data acquisition for three-dimensional modeling of complex sites," *J. Field Robot.* **26**(11–12), 865–891.
15. G. Papadopoulos, H. Kurniawati and N. Patrikalakis, "Asymptotically Optimal Inspection Planning using Systems with Differential Constraints," *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, Karlsruhe, Germany (2013) pp. 4126–4133.
16. S. Karaman and E. Frazzoli, "Sampling-based algorithms for optimal motion planning," *Int. J. Robot. Res.* **30**(7), 846–894 (2011).
17. E. Plaku, K. E. Bekris, B. Y. Chen, A. M. Ladd and L. Kavraki, "Sampling-based roadmap of trees for parallel motion planning," *IEEE Trans. Robot.* **21**(4), 597–608 (2005).
18. W. Wang, L. Yan, X. Xu and S. X. Yang, "An Adaptive Roadmap Guided Multi-RRTs Strategy for Single Query Path Planning," *Proceedings of the 2010 IEEE International Conference on Robotics and Automation (ICRA),* IEEE, Anchorage, AK, USA (2010) pp. 2871–2876.
19. D. Devaurs, T. Simeon, J. Cortés *et al.*, "A Multi-Tree Extension of the Transition-Based RRT: Application to Ordering-and-Pathfinding Problems in Continuous Cost Spaces," *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, Chicago, IL, USA (2014) pp. 2991–2996.
20. D. Hsu, R. Kindel, J. C. Latombe and S. Rock, "Randomized kinodynamic motion planning with moving obstacles," *The International Journal of Robotics Research*, **21**(3), 233–255 (2002 March). doi: 10.1177/027836402320556421.
21. M. W. Achtelik, S. Lynen, M. Chli and R. Siegwart, "Inversion Based Direct Position Control and Trajectory Following for Micro Aerial Vehicles," *IEEE/RSJ Conference on Intelligent Robots and Systems (IROS)*, Tokyo, JP (2013) pp. 2933–2939.
22. J. Nikolic, J. Rehder, M. Burri, P. Gohl, S. Leutenegger, P. T. Furgale and R. Siegwart, "A Synchronized Visual-Inertial Sensor System with FPGA Pre-Processing for Accurate Real-Time Slam," *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)* IEEE, Hong Kong (2014) pp. 431–437.
23. Ascending Technologies GmbH, "http://www.asctec.de/".
24. Autodesk Inc., "http://www.123dapp.com/catch".
25. A. Hornung, K. M. Wurm, M. Bennewitz, C. Stachniss and W. Burgard, "OctoMap: An efficient probabilistic 3D mapping framework based on octrees," *Auton. Robots* **34**(3), 189–206 (2013).

2015 IEEE International Conference on Robotics and Automation (ICRA)
Washington State Convention Center
Seattle, Washington, May 26-30, 2015

Structural Inspection Path Planning via Iterative Viewpoint Resampling with Application to Aerial Robotics

Andreas Bircher, Kostas Alexis, Michael Burri, Philipp Oettershagen,
Sammy Omari, Thomas Mantel and Roland Siegwart[1]

***Abstract*— Within this paper, a new fast algorithm that provides efficient solutions to the problem of inspection path planning for complex 3D structures is presented. The algorithm assumes a triangular mesh representation of the structure and employs an alternating two–step optimization paradigm to find good viewpoints that together provide full coverage and a connecting path that has low cost. In every iteration, the viewpoints are chosen such that the connection cost is reduced and, subsequently, the tour is optimized. Vehicle and sensor limitations are respected within both steps. Sample implementations are provided for rotorcraft and fixed–wing unmanned aerial systems. The resulting algorithm characteristics are evaluated using simulation studies as well as multiple real–world experimental test–cases with both vehicle types.**

I. Introduction

The ongoing boom in utilizing mobile robots for real–life applications sets new demands regarding their autonomy. In the particularly interesting field of inspection operations, aerial, maritime or ground robots are already utilized for critical tasks such as infrastructure surveillance, damage assessment or victim search. In such scenarios, the structure to be inspected may be given as a 3D model (typically a mesh from CAD software, Geographical Information System data or civil engineering instrumentation) and robots are employed either to derive an updated, possibly higher–fidelity model, or scan for risks and hazards (e.g. cracks).

To facilitate autonomous inspection planning capabilities, a robot must be equipped with algorithms that allow it to quickly compute efficient paths that result in full coverage of the structure to be inspected, while respecting any sensor limitations and motion constraints that may apply. Such a problem belongs to the general class of coverage planning and –despite the interest of the community– its inherent difficulties still limit the performance, efficiency and applicability of the proposed solutions. Furthermore, so far, only few works validated such algorithms in experimental studies.

Within this work, a novel fast iterative algorithm for structural inspection is proposed. The new algorithm employs an alternating two–step optimization paradigm to find good viewpoints that together provide full coverage and lead to a connecting path that is of low cost. In every iteration, each viewpoint is chosen such as to reduce the cost–to–travel between itself and its neighbours (*first step*) and subsequently the optimally connecting tour is recomputed (*second step*), while vehicle constraints and sensor limitations are respected in all phases. Extensive evaluation studies including 3D reconstruction experiments using a rotorcraft (hexarotor) Unmanned Aerial Vehicle (UAV), as well as a 5.6m wingspan glider fixed–wing UAV (both shown in Figure 1) reveal the high–performance properties of the algorithm in challenging scenarios and subject to real vehicle and sensor constraints. An open source implementation [17] as well as the point-clouds resulting from the experiments [2] are provided for further use and development by the comunity.

Fig. 1: Indicative inspection 3D reconstruction results using the proposed structural inspection planner and a rotorcraft as well as a fixed–wing UAV equipped with camera sensors.

A short overview of the related work and how our approach contributes further is presented in Section II, followed by the problem description in Section III. Afterwards, the proposed approach is presented in Section IV, while computational analysis takes place in Section V. Finally, evaluation test–cases in simulation and experiments are shown in Section VI, followed by conclusions in section VII.

II. Related Work

In the literature, many contributions have been made towards addressing the challenges of coverage planning. System and environment allowing, the space may be represented by a simplified discrete grid and paths can be computed using wavefront algorithms [23], spanning trees [7] or neural networks [16]. The work in [4] proposed a cellular decomposition of the planning space, covering each free cell with a sweeping pattern and an advanced algorithm following this

This work has received funding from the European Union's Horizon 2020 Research and Innovation Programme under the Grant Agreement No.644128, AEROWORKS as well as by the FP7-framework Grant No. 285417, ICARUS.

[1]All authors are with the Autonomous Systems Lab at ETH Zurich, Leonhardstrasse 21, 8092, Zurich, Switzerland. email: andreas.bircher@mavt.ethz.ch

concept was presented in [1]. Using a 2D planner, the authors in [11] approximate a 3D structure using multiple 2D layers.

Aiming towards real 3D structural inspection, advanced algorithms have recently been proposed. Within the most recent contributions, those that employ a two–step optimization scheme proved to be more versatile with respect to the inspection scenario. In a first step, such algorithms compute the minimal set of viewpoints that cover the whole structure which corresponds to solving an Art Gallery Problem (AGP). As a second step, the shortest connecting tour over all these viewpoints has to be computed, which is the Traveling Salesman Problem (TSP). Fast algorithms to approximately solve these two NP–hard, but well studied problems, are known, for example in [9, 18] for the AGP, and [5, 15] for the TSP. A recent application of these concepts, that allows some redundancy in the AGP such that it is able to improve the path in a post–processing step, was presented in [12]. This algorithm can deal with 3D scenarios and is demonstrated in experiments using underwater vehicles for ship hull inspection. Addressing the problem from a different perspective, the work in [21] concentrates on deriving close–to–optimal solutions at the inherently large cost of computational efficiency. A comprehensive survey of the existing coverage path planning methods may be found in [8].

The proposed fast inspection path planner retains a two–step optimization structure but contrary to trying to find a minimal set of guards in the AGP it rather tries to sample them such that the connecting path is short while ensuring full coverage. This is driven by the idea that with a continuously sensing sensor the number of viewpoints (and if this is minimal or not) is not necessarily important but mostly their configuration in space, which has to be such that short and full coverage paths are provided. As a result, this novel approach leads to full coverage paths that are of low cost and are computed quickly.

III. Problem Description

The problem of structural inspection path planning, as it is considered in this paper, consists of a 3D structure to be inspected, a system with its dynamics and constraints and an integrated sensor, the limitations of which have to be respected. The 3D structure to be inspected is represented by a triangular mesh, embedded in a bounded environment that may contain obstacle regions. The problem setup is to be such that for each triangle in the mesh, there exists an *admissible* viewpoint configuration – that is a viewpoint from which the triangle is visible for a specific sensor model. Then, for the given environment and with respect to the operational constraints, a path for the system has to be found that guarantees complete inspection of the 3D structure. Quality measures for paths are situation specific, depending on the system and mission objectives, e.g. time or distance.

As sample systems we consider a rotorcraft and a fixed–wing UAV, both equipped with a visual camera with a fixed orientation relative to the platform. Minor adaptations to the proposed path–planner enable its use for other common robot configurations such as underwater ROVs or wheeled robots.

IV. Proposed Approach

As the algorithm does not focus on minimizing the number of viewpoints, the proposed approach selects one (*admissible*) viewpoint for every triangle in the mesh of the structure to be inspected. In order to compute viewpoints that allow low–cost connections, an iterative resampling scheme is employed. Between each resampling, the best path for the current viewpoints is computed. The cost to connect to the current neighbours on the tour provides a metric for the quality of the viewpoint in the subsequent resampling. The initial selection of viewpoints for the first iteration is arbitrarily done such that full coverage is provided with non–optimized viewpoints. A fast implementation of the Lin-Kernighan-Helsgaun Heuristic (LKH) TSP solver [10] is employed to compute the best tour, while the cost of the interconnecting pieces of path is calculated by means of a boundary value solver (BVS). Algorithm 1 presents an overview of the proposed inspection planning procedure.

Algorithm 1 Inspection path planner

```
k ← 0
Sample initial viewpoint configurations
Compute cost matrix for the TSP solver (Section IV-A)
Solve the TSP problem to obtain initial tour
while running
    Resample viewpoint configurations (Section IV-B)
    Recompute the cost matrix (Section IV-A)
    Recompute best tour T_best using the LKH and update
        best tour cost c_best if applicable
    k ← k + 1
end while
return T_best, c_best
```

In the following, the formulations of the path computation and the viewpoint sampling for a rotorcraft UAV are given. Subsequently in Section IV-C, extensions and adaptations to enable planning for a fixed–wing UAV are discussed.

A. Path Computation and Cost Estimation

To find the best tour among the viewpoints, the TSP solver requires a cost matrix containing the connection cost of all pairs of viewpoints. The path generation and its cost estimation relie on a two state BVS. The BVS is either employed directly to connect the two viewpoints or as a component in a local planner, in case the direct connection is not feasible due to obstacles. In that case, our implementation makes use of the RRT*-planner [14] to find a collision–free connection. The proposed model for a rotorcraft UAV consists of position as well as yaw, $\xi = \{x, y, z, \psi\}$. Roll and pitch angles are considered to be near zero as slow maneuvering is desired to achieve increased accuracy. The path from configuration ξ_0 to ξ_1 is given by $\xi(s) = s\xi_1 + (1 - s)\xi_0$, where $s \in [0, 1]$. The single limitation considered is the speed limit. The translational limit is denoted by v_{max} while the rotational speed is limited by $\dot{\psi}_{max}$. Both values are small, such that the tracking of

paths with corners is sufficiently accurate. The resulting execution time is $t_{ex} = \max(d/v_{max}, \|\psi_1 - \psi_0\|/\dot{\psi}_{max})$, with d the Euclidean distance. The cost of a path segment corresponds to the execution time t_{ex}.

B. Viewpoint Sampling

For every triangle in the mesh, one viewpoint has to be sampled, the position and heading of which is determined sequentially in the proposed procedure while retaining visibility of the corresponding triangle. First, the position is optimized for distance to the neighbouring viewpoints using a convex problem formulation and only then, the heading is optimized. To guarantee a good result of this multistep optimization process, the position solution must be constrained such as to allow finding an orientation for which the triangle is visible.

Specifically, the constraints on the position $g = [x, y, z]$ consist of the inspection sensor limitations of minimum incidence angle, minimum and maximum range (d_{min}, d_{max}) constraints (depicted in Figure 2a). They are formulated as a set of planar constraints:

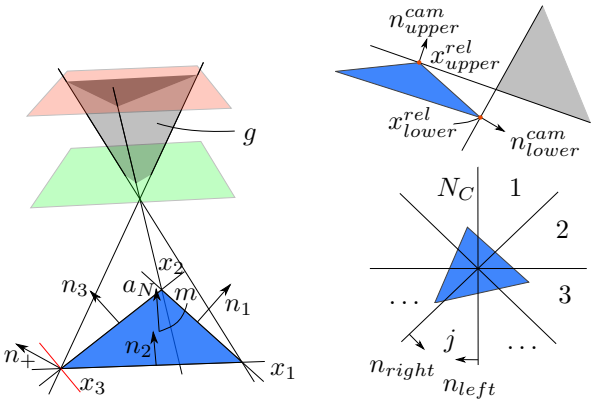
$$\begin{bmatrix} (g - x_i)^T n_i \\ (g - x_1)^T a_N \\ -(g - x_1)^T a_N \end{bmatrix} \succeq \begin{bmatrix} 0 \\ d_{min} \\ -d_{max} \end{bmatrix}, i = \{1, 2, 3\} \quad (1)$$

where x_i are the corners of the mesh triangle, a_N is the normalized triangle normal and n_i are the normals of the separating hyperplanes for the incidence angle constraints as shown in Figure 2a.

Further, the camera has a limited field of view (FoV) with a certain horizontal and vertical opening and is mounted to the system with a fixed pitch angle. The imposed constraint on the sampling space resulting from the vertical camera opening is not convex (a revoluted 2D-cone, the height of which is depending on the relevant corners of the triangle over the revolution). To approximate and convexify the problem, the space is divided in N_C equal convex pieces according to Figure 2b. The optimum is computed for every slice in order to find the globally best solution. The constraints for piece j are derived as follows: Left and right boundaries of the sampling space are the borders of the revolution segment and the cone top and bottom are represented by a single plane tangential to the centre of the slice. Angular camera constraints in horizontal direction are not encoded and instead d_{min} is chosen high enough to allow full visibility of the triangle. This leaves some space for variation in the sampling of the heading, where the horizontal constraints are enforced. Specifically, these constraints are:

$$\begin{bmatrix} (g - x^{rel}_{lower})^T n^{cam}_{lower} \\ (g - x^{rel}_{upper})^T n^{cam}_{upper} \\ (g - m)^T n_{right} \\ (g - m)^T n_{left} \end{bmatrix} \succeq \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \end{bmatrix}, \quad (2)$$

where $x^{rel}_{lower}, x^{rel}_{upper}$ are the respective relevant corners of the mesh triangle, m the middle of the triangle and n^{cam}_{lower}, n^{cam}_{upper}, n_{right} and n_{left} denote the normal of the respective separating hyperplanes.



(a) Incidence angle constraints on a triangular facet

(b) Camera constraints and convexification

Fig. 2: a) The figure depicts the three main planar angle of incidence constraints on all three sides of the triangle. For a finite number of such constraints the incidence angle is only enforced approximately. The red line (and n_+) demarks a sample orientation for a possible additional planar constraint at a corner. Minimum (green plane) and maximum (red plane) distance constraints are similar planar constraints on the sampling area. These constraints bound the sampling space, where g can be chosen, on all sides (gray area). b) The vertical camera angle constraints with the relevant corners of the triangle in red are depicted in the upper part, while beneath the partition of the space for convexification is depicted.

The optimization objective for the viewpoint sampling in iteration k, in the case of a rotorcraft UAV, is to minimize the sum of squared distances to the preceding viewpoint g_p^{k-1}, the subsequent viewpoint g_s^{k-1} and the current viewpoint in the old tour g^{k-1}. The former two parts potentially shorten the tour by moving the viewpoints closer together, while the latter limits the size of the improvement step, as g_p^{k-1} and g_s^{k-1} potentially move closer as well.

The resulting convex optimization problem is given below. Its structure as a Quadratic Program (QP) with linear constraints allows the use of an efficient solver [6].

$$\min_{g^k} \quad (g^k - g_p^{k-1})^T (g^k - g_p^{k-1}) + \quad (3)$$
$$(g^k - g_s^{k-1})^T (g^k - g_s^{k-1}) + (g^k - g^{k-1})^T (g^k - g^{k-1})$$

$$\text{s.t.} \quad \begin{bmatrix} n_1^T \\ n_2^T \\ n_3^T \\ a_N^T \\ -a_N^T \\ n^{cam\ T}_{lower} \\ n^{cam\ T}_{upper} \\ n_{right}^T \\ n_{left}^T \end{bmatrix} g^k \succeq \begin{bmatrix} n_1^T x_1 \\ n_2^T x_2 \\ n_3^T x_3 \\ a_N^T x_1 + d_{min} \\ -a_N^T x_1 - d_{max} \\ n^{cam\ T}_{lower} x^{rel}_{lower} \\ n^{cam\ T}_{upper} x^{rel}_{upper} \\ n_{right}^T m \\ n_{left}^T m \end{bmatrix} \quad (4)$$

For the computed optimal position, the heading is determined according to the criterion $\min_{\psi^k} = \left(\psi_p^{k-1} - \psi^k\right)^2 / d_p + \left(\psi_s^{k-1} - \psi^k\right)^2 / d_s$, s.t. $\mathbf{Visible}(g^k, \psi^k)$, where $\mathbf{Visible}(g^k, \psi^k)$ means that from the given configuration, g^k and ψ^k, the whole triangle is visible. d_p and d_s are the Euclidean distances from g^k to g_p^{k-1} and g_s^{k-1} respectively. For simple sensor setups establishing the boundaries on ψ^k

for $\mathbf{Visible}(g^k, \psi^k) = \text{TRUE}$ makes the solution explicit. Otherwise a grid search can be employed.

C. Extensions for Fixed-Wing UAVs

Fixed–Wing UAVs correspond to another excellent configuration for inspection operations. However, their advantages in aspects like long–endurance, come together with limitations on handling sharp turns, steep ascents or descents. Moreover, the direction of a fixed camera is related to the direction of travel. Accordingly, the implementation for the BVS and the viewpoint sampling have to be adapted.

Assuming that highly dynamic maneuvers are avoided for inspection flights, the minimum turn radius of the aircraft is constrained to be $r_{\min}$ while roll and pitch are considered to be near zero. For planning purposes, the xy–plane vehicle dynamics are captured using Dubins curves, thus minimizing the distance w.r.t. $r_{\min}$. Furthermore, in the vertical direction the path is constrained by a maximum climb and sink rate. Since these values are small, instantaneous changes are acceptable and the rate $\dot{z}$ is chosen to be constant along a path segment. If the maximum rate is exceeded, ascending/descending loitering circles are added at the end of the path segment to allow larger changes of height. In many practical cases such as flat landscape coverage, it makes sense to constrain the height of the path to a fixed value to avoid undesirable loitering circles. The fixed-wing UAV is assumed to travel with constant velocity v_{FW} and the path cost is the time $t_{ex} = l_{Path}/v_{FW}$, with l_{Path} the path length.

In contrast to the case of rotorcraft UAVs, where only the distance is minimized in the viewpoint position sampling step, the fixed-wing UAV sampler also aims to align the viewpoints on a as straight line as possible. This effectively avoids too many curly path segments and thus, together with the distance minimization tends to reduce the path length. The addition in the objective is therefore to minimize the squared distance d^2 to the straight line between the neighbouring viewpoints. Using its direction vector b, the distance is calculated as follows:

$$b = \frac{g_s^{k-1} - g_p^{k-1}}{\left\| g_s^{k-1} - g_p^{k-1} \right\|} \tag{5}$$

$$d = \left\| b \times (g^k - g_p^{k-1}) \right\| = \left\| \begin{bmatrix} 0 & -b_3 & b_2 \\ b_3 & 0 & -b_1 \\ -b_2 & b_1 & 0 \end{bmatrix} (g^k - g_p^{k-1}) \right\|$$

and with

$$\begin{bmatrix} 0 & -b_3 & b_2 \\ b_3 & 0 & -b_1 \\ -b_2 & b_1 & 0 \end{bmatrix} (g^k - g_p^{k-1}) = q \tag{6}$$

follows $d^2 = q^T q$. To avoid the insertion of unnecessary circles, the distance between the viewpoints has to be large enough according to their heading, the direction to the next viewpoint and $r_{\min}$. The bounds on that distance $l_i,\ i = \{p, s\}$ are derived geometrically and evaluated using numerical algorithms. The distance criterions are therefore $(g^k - g_i^{k-1})^T (g^k - g_i^{k-1}) \geq {l_i}^2,\ i = \{p, s\}$ which are non-convex. To convexify, the criterions are linearized around the old viewpoint. This adaptation is conservative by the exclusion of the non-convex part and attenuates the impact of the extrapolation error:

$$(g^k - g_i^{k-1})^T (g^{k-1} - g_i^{k-1}) \quad \geq \quad {l_i}^2,\ i = \{p, s\} \tag{7}$$

Wrapping all up in a single QP-formulation and adding the two slack variables ϵ_p and ϵ_s with constant C to allow occasional violation of the minimal distance criterion:

$$\min_{q, g^k} \quad \text{(objective in (4))} + q^T q + C(\epsilon_p + \epsilon_s) \tag{8}$$

$$\text{s.t.} \quad \begin{bmatrix} 0 & -b_3 & b_2 \\ b_3 & 0 & -b_1 \\ -b_2 & b_1 & 0 \end{bmatrix} (g^k - g_p^{k-1}) = q$$

$$\begin{bmatrix} \text{constraints in (4)} \\ (g^{k-1} - g_p^{k-1})^T \\ (g^{k-1} - g_s^{k-1})^T \end{bmatrix} g^k \succeq \begin{bmatrix} \text{constraints in (4)} \\ {l_p}^2 - \epsilon_p \\ {l_s}^2 - \epsilon_s \end{bmatrix} \tag{9}$$

$$\epsilon_p \geq 0$$

$$\epsilon_s \geq 0$$

The criterion of Equation (7) is inverted for the heading computation and applied as long as a feasible solution is found. The proposed approach also works efficiently in case of obstacles up to some complexity by dividing the sampling space in convex pieces that are evaluated individually. In our implementation obstacles are approximated with cuboids.

D. Additional Heuristic Concepts

Additional heuristic measures increase the quality of computed paths. These primarily concern the rotorcraft UAV path planning. In order to allow a faster and more rigorous ordering of the viewpoints, initial iterations of the algorithm consider not the nearest neighbour on the tour to minimize the distance to, but neighbours that are $N_{Neighbour}$ away on both sides. $N_{Neighbour}$ is then decremented in every iteration to finally reach 1. To further improve the viewpoint ordering, the allowable yaw rate is set lower in the initial iterations and then slowly increased to reach the maximally allowed $\dot{\psi}_{max}$.

TABLE I: Scalable Inspection Scenario

N_{facets}	[100...3600]	∠incidence	30°
FoV	$[70, 70]^\circ$	Mouting pitch	25°
Range	unconstrained	Height	200m
r_{min}	60m	v_{FW}	9m/s
$v_{\max}$	5m/s	$\dot{\psi}_{\max}$	0.5rad/s

V. Computational Analysis

In order to evaluate the capabilities of the proposed algorithm, a simple and scalable scenario is used. An array of equilateral triangles is arranged in a plane as shown in Figure 3, with a height of 1250m and a width of $\frac{\sqrt{3}}{2}$2500m. This corresponds to an area of 2.71km^2. This area is filled with a variable number of equilateral triangles, effectively corresponding to different mesh resolutions and thus numbers of viewpoints. The mesh resolution is varied to examine the impact on the quality of the resulting path, while the number of viewpoints is a meaningful parameter of the problem complexity and the time the algorithm needs for execution.

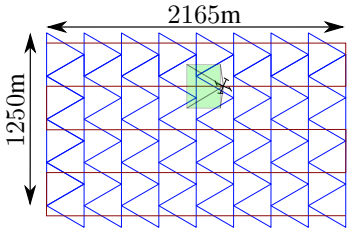
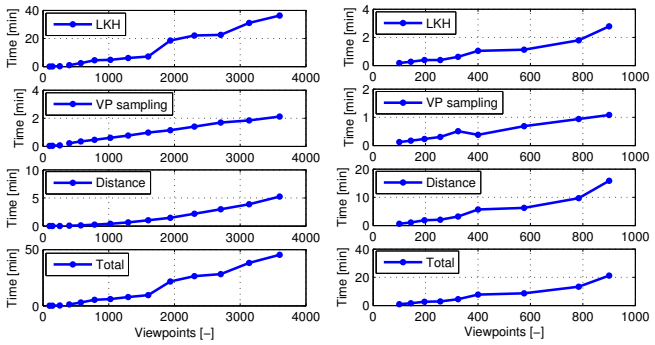
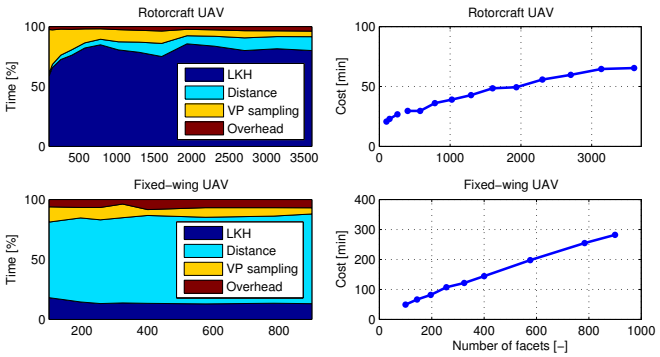


Fig. 3: Illustration of the triangular pattern that is used in different resolutions for the following analysis of the algorithm's characteristics. Overlayed in brown is a naive sweeping path with a certain base line (in this case the same as the triangle edge length).



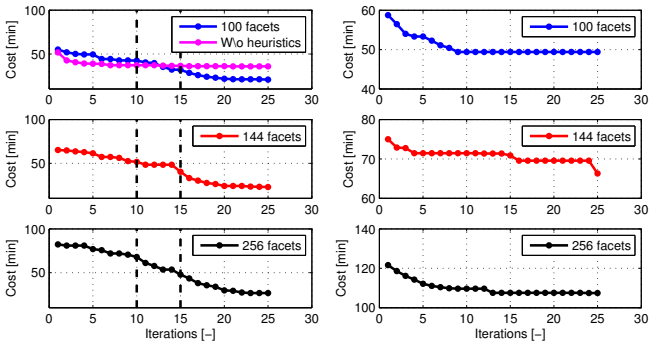
(a) Rotorcraft UAV (b) Fixed-wing UAV

Fig. 4: Correlation of the number of viewpoints for both systems with the computational time consumption. The time consumption curve is given individually for different components of the algorithm.



(a) Relative time consumption (b) Resolution dependent cost

Fig. 5: Figure 5a depicts the relative time consumption of different parts of the algorithm, while Figure 5b shows the cost of the computed paths for different amount of facets, corresponding to varying mesh resolution.



(a) Rotorcraft UAV (b) Fixed-wing UAV

Fig. 6: Improvement of the path cost over the course of 25 iterations for 100, 144 and 256 facets. The left top figure depicts a run without heuristics and one with the heuristic on the viewpoint neighbours (running for the first 10 iterations) and the heuristic on the viewpoint yaw (running for the first 15 iterations). As shown, for the fixed–wing UAV case a less smooth path improvement procedure takes place after the first iterations due to the additional complexity introduced by the nonholonomic constraints.

Both for the rotorcraft and the fixed–wing UAV cases, the inspection is performed from a constant height of 200m above ground, with a minimum incidence angle of 30°. The employed camera is mounted with a pitch of 25°, the FoV is 70° in both vertical and horizontal directions, while a margin for robustness is deducted in the horizontal direction, thus leading to an effectively usable opening of 60°. This reduction from the nominal FoV is done in all test cases. The rotorcraft UAV is assumed to move with a maximum translational speed of 5m/s and a maximum yaw rate of 0.5rad/s, while the minimum turn radius for the fixed-wing is 60m. Simulations for both models, using triangular patterns with variable numbers of facets, were performed on a computer with a 1.73GHz processor running Ubuntu 14.04 and using a single-thread C++ implementation. The time consumption for the computation is depicted in Figure 4 for the rotorcraft and the fixed–wing UAV cases. It shows accumulated time consumptions of the different parts of the algorithm, as well as the total, while Figure 5a depicts the relative shares. The time complexity of the algorithm for large numbers of viewpoints is dominated by the LKH for which [10] gives a time complexity of $\mathcal{O}(N^{2.2})$, where N is the number of viewpoints. Less dominant is the viewpoint sampling, with a complexity of $\mathcal{O}(N)$, since the constant complexity method of sampling has to be repeated for every viewpoint. Lastly, the complexity of the distance computations is $\mathcal{O}(N^2)$, since the distance has to be computed for all pairs of viewpoints. The predicted behaviour of the viewpoint sampling and distance computation complexities can be observed in the plots for the rotorcraft UAV case (Figure 4a), which contain scenarios for up to 3600 viewpoints. For lower numbers of viewpoints 25 iterations were performed, while for numbers of viewpoints above 900, 50 iterations were performed in order to find high quality paths. Consequently the rotorcraft graphs contain two seperate curves. The fixed–wing UAV graphs in Figure 4b were computed with 25 iterations for numbers of viewpoints up to 900. Overall, the relative time consumption plots in Figure 5a show an increasing share of computation time for the LKH for larger numbers of viewpoints, as could be expected from the theoretical complexities. Evidently, the distance computation, which timewise is insignificant for the rotorcraft UAV case, consumes a large part of the computation time for the fixed–wing case.

This is due to the fact, that the system constraints induce an asymetric TSP, which doubles the amount of viewpoints (corresponding to both directions of travel). While the TSP solver can efficiently handle this, the number of two state boundary value problems is quadrupled. As the number of facets increases, the path is more densely populated with viewpoints, thus attenuating the increase of cost through the larger amount of connections. This can be observed in Figure 6b for both the rotorcraft and the fixed–wing UAV, respectively. To further validate the proposed approach, comparison to a sweeping path as depicted in Figure 3 can be made. With an image base–line of approximately 190m 7 sweeps of 2165m length are necessary to cover the area, to which adds the translation of 6 times the base–line. With a travel speed of 5m/s the resulting path cost is 3259s, which is more than double the cost of what the proposed algorithm computes for a rotorcraft UAV when planning with e.g. 100 facets (1234.90s). This stands as an indication of the performance of the algorithm, the main strength of which is however related with complex 3D scenarios, as those presented in Section VI, where simplified approaches like the sweeping path lose their potential.

Finally, the path length over the course of 25 iterations is depiced in Figure 6 for 100, 144 and 256 facets. The effect of the additional heuristics on finding a better final solution for the rotorcraft UAV case is shown in the top plot of Figure 6a where the cost curve is compared to the one without the heuristics discussed in Section IV-D. Overall, quick progression towards lower–cost paths is achieved for all scenarios, while for the fixed–wing UAV case the curve flattens sooner due to the additional complexity introduced by the nonholonomic constraints.

VI. Evaluation Test–Cases

Within this section advanced evaluation test–cases in simulation and experimental studies are presented. Three challenging experiments are presented, one using a rotorcraft and two with a 5.6m wingspan fixed–wing UAV.

A. Complex 3D Simulation Test–Case

As a complex simulation test–case, a mesh model of the 405m high Central Radio & TV Tower in Beijing was used [3]. The employed mesh contains $N_{facets} = 1701$ triangular facets that model the real building. A rotorcraft vehicle is assumed which is subject to a maximum allowed linear velocity of $v_{\max} = 2$m/s and a maximum yaw rate $\dot{\psi}_{\max} = 0.5$rad/s while it carries a camera sensor mounted with 15° pitch and has a field of view $[120, 120]^\circ$ along the vertical and the horizontal axis respectively. Furthermore, it is enforced that the distance of the camera to the inspected structure is set between 10m and 25m to ensure safety but also close–inspection capable of revealing structural problems (e.g. cracks). The test–case parameters are summarized in Table II. The building and the derived inspection path are illustrated in Figure 7. Eventually, this complex test–study reveals the high–performance characteristics of the proposed inspection planner which are further evaluated in the experimental studies presented in subsections VI-B and VI-C.

TABLE II: Beijing Tower Inspection Scenario

N_{facets}	1701	$\angle$incidence	30°
FoV	$[120, 120]^\circ$	Mouting pitch	15°
$d_{\min}$	10m	$d_{\max}$	25m
$v_{\max}$	2m/s	$\dot{\psi}_{\max}$	0.5rad/s

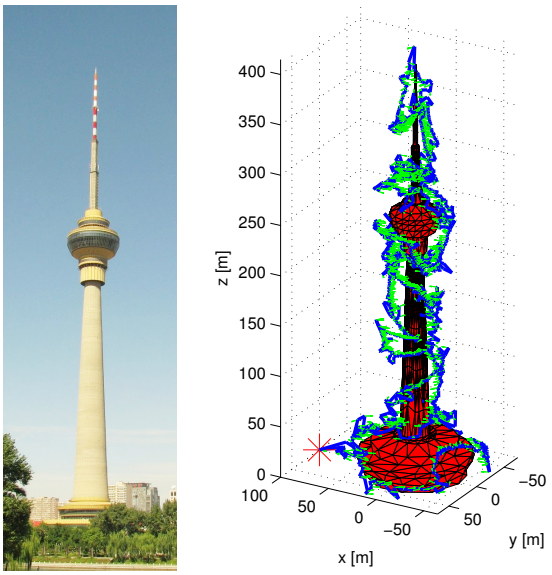


Fig. 7: Large scale structure to be inspected: The 405m high Central Radio & TV Tower in Beijing. The mesh used to compute the path contains 1701 triangular facets. After a computation time of 92s the cost for the inspection is 2997.44s with a maximal speed of 2m/s and a maximal yaw rate of 0.5rad/s. The red point denotes, start– and end–point of the inspection.

B. Rotorcraft UAV Inspection Operations

A first experimental test study was conducted using an AscTec Firefly Hexacopter MAV onboard of which the Visual–Inertial Sensor (VI–Sensor) developed by our lab and Skybotix AG was further integrated. The VI–Sensor integrates 2 HDR global shutter cameras (Aptina MT9V034) and an Analog Devices ADIS16448 IMU in a tightly aligned and synchronized way using an Artix^{TM}–7 FPGA, a Xilinx Zynq 7020 SoC module and an ATOM CPU running Linux. This integrated sensor system runs advanced image processing algorithms, provides complete pose estimates and builds a 3D map of the environment. Figure 8 shows the employed UAV equipped with the VI–Sensor.

The experimental setup refers to the inspection of the 3D structure shown in Figure 9 and consists of 106 facets capturing the structure in detail. The scenario is further complicated by a bounding box of 3×3×2.75m and a sensing minimum range $d_{\min} = 1$m. This inspection structure was offline reconstructed from a set of terrestrial images and consequently a mesh was computed to be utilized by the proposed inspection path planner. Table III summarizes the experiment parameters. Using the proposed inspection path

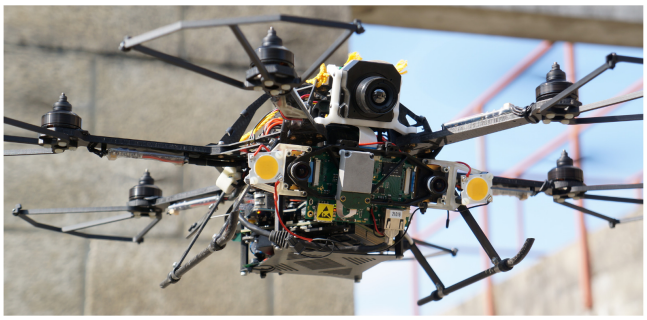


Fig. 8: The Firefly UAV equipped with the VI–Sensor.

TABLE III: Rotorcraft UAV Inspection Scenario

N_{facets}	106		
∠incidence	30°	Bounding box	3x3x2.75m
FoV	$[60, 90]°$	Mouting pitch	15°
$d_{\min}$	1m	$d_{\max}$	3m
$v_{\max}$	0.25m/s	$\dot{\psi}_{\max}$	0.5rad/s

planner, a path that guarantees complete coverage is derived and has a total length of 151.44s. Reconstruction results derived using pose–annotated (position and rotations) image sequences from one of the VI–Sensor cameras and the Pix4D software indicate excellent 3D reconstruction results, a fact that further increases confidence on the practical applicability of the proposed algorithm. The reference path and the recorded flight response along with the reconstruction results are shown in Figure 9. The arrows indicate the reference viewpoints proposed by the inspection planner.

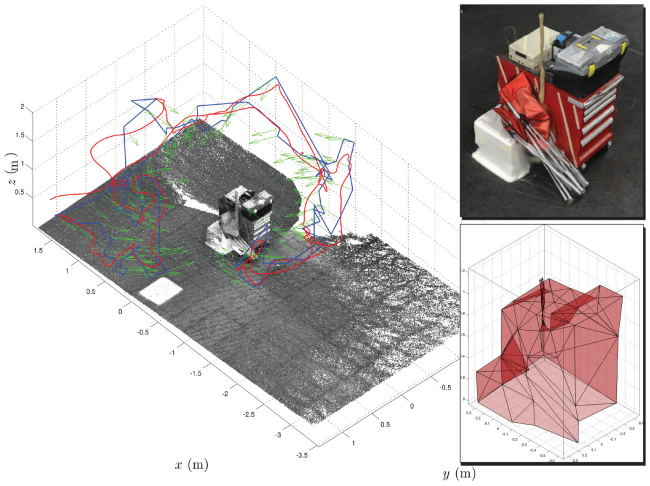


Fig. 9: Experimental study of the inspection of a trolley. The preliminary, terrestrial images–based, 3D reconstruction of the inspection structure is depicted and was used to derive a simplified mesh that was then employed by the inspection path planner to compute the inspection path shown in the Figure. The path cost is 151.44s for $v_{\max} = 0.25$m/s and $\dot{\psi}_{\max} = 0.5$rad/s .

C. Fixed-Wing UAV Inspection Operations

A second set of experiments was conducted using a long endurance fixed–wing UAV platform developed by our lab. The particular platform, AtlantikSolar [20], is a 5.6m wingspan, 7.5kg, solar–powered vehicle with robust state–estimation capabilities [22], automatic trajectory tracking control [19] and further integrates a) an advanced sensor pod with a monocular version of the aforementioned VI–Sensor with the Aptina MT9V034 camera mounted at a 50° front–down oblique view and every image is fully pose–annotated as well as b) a GPS–tagged Sony HDR-AS100VW camera. Figure 10 depicts the UAV as well as the sensor pod.

TABLE IV: Marche–en–Famenne Inspection Scenario

N_{facets}	8, 8	∠incidence	30°
v_{FW}	9m/s	$r_{\min}$	60m, 60m
FoV	$[90, 50]°$, $[120, 120]°$	Mouting pitch	50°, 90°
Range	unconstrained	Height	120m, 100m

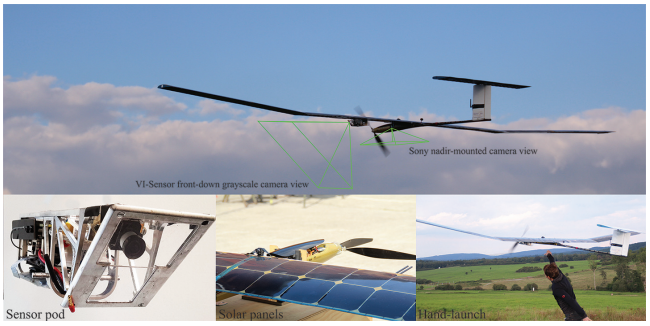


Fig. 10: The AtlantikSolar UAV with the sensor pod attached to its wings and further photos of the sensor pod, the solar cells and an instant of the hand–launching.

With this UAV corresponding to an excellent test–case for nonholonomic inspection path planning, two inspection missions were designed to be conducted within the framework of the ICARUS project field–trials in the area of Marche–en–Famenne in Belgium [13]. Geographical Information System (GIS) data were used to derive a first, rough, 8 facets ($N_{facets} = 8$) mesh of the area and subsequently two inspection paths were computed, one for the oblique view grayscale camera of the VI–Sensor with a fixed reference altitude set at an absolute value $z^r = 362$m (corresponding to 120m above the highest point to be inspected) and the other for the nadir–mounted Sony HDR-AS100VW with $z^r = 342$m, while the modelled minimum turning radius was $r_{\min} = 60$m. Table IV summarizes the parameters used for the two experiments. Figures 11 and 12 present the results for the two camera configurations. In both cases, the optimized reference inspection path, the real recorded UAV trajectory as well as an offline computed dense point–cloud of the inspection area are shown such that the completeness of coverage is visually assessed. The point clouds are derived using the pose–annotated images and the Pix4D software and are freely available online [2].

For both configurations the proposed inspection planner manages to provide short distance paths that guarantee complete coverage while accounting for the motion constraints of the fixed–wing UAV. The reconstructed point cloud is

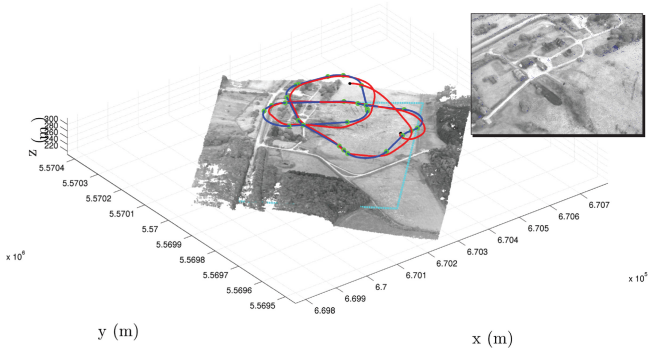


Fig. 11: Inspection path and point–cloud for 3D reconstruction purposes using the front–down mounted view grayscale camera of the VI–Sensor onboard AtlantikSolar. Blue line represents the reference path, green circles are used to indicate the actual waypoints loaded to the autopilot and red is used for the vehicle response. The planner commands the vehicle to navigate such that the camera covers the whole desired area marked with dashed cyan line. A UTM31N coordinate system is employed.

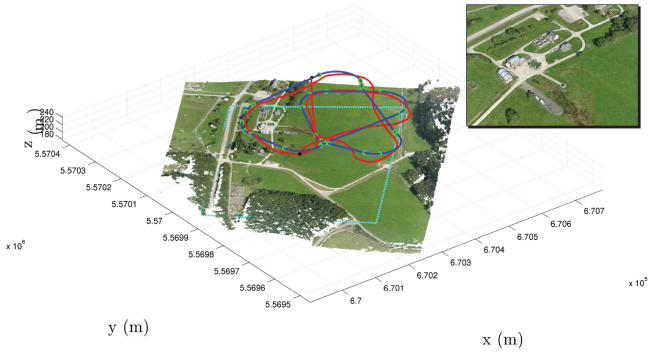


Fig. 12: Inspection path and 3D reconstruction results using the nadir mounted Sony HDR-AS100VW onboard AtlantikSolar. Blue line represents the reference path, green circles are used to indicate the actual waypoints loaded to the autopilot and red is used for the vehicle response. The planner commands the vehicle to navigate such that the camera covers the whole desired area marked with dashed cyan line. A UTM31N coordinate system is employed.

very dense which indicates that such short paths remain practically useful for high fidelity reconstruction purposes where significant overlap is typically required.

VII. Summary & Conclusions

Within this paper, a practically–oriented fast inspection path planning algorithm capable of computing efficient solutions for complex 3D structures represented by triangular meshes was presented. The method was first tested on a scalable scenario and the results were summarized both for the case of a rotorcraft as well as a fixed–wing UAV. Subsequently, the capabilities of the algorithm were demonstrated in real–world application scenarios and experimental studies using both UAV configurations. With the help of 3D–reconstruction software, the recorded inspection data were postprocessed to support the claim of finding full coverage paths and the point cloud datasets are released to enable evaluation of the inspection quality. An implementation of the presented algorithm is accessible [17] for further use and development by the community.

References

[1] E. U. Acar, H. Choset, and J. Y. Lee, "Sensor-based coverage with extended range detectors," *Robotics, IEEE Transactions on*, vol. 22, no. 1, pp. 189–198, 2006.

[2] Accompanying datasets with inspection results in Point-Cloud form. [Online]. Available: http://goo.gl/IUMWQH

[3] Beijing Central TV & Radio Tower Model, 3D Warehouse, "https://3dwarehouse.sketchup.com/."

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

[5] G. Dantzig, R. Fulkerson, and S. Johnson, "Solution of a large-scale traveling-salesman problem," *Journal of the operations research society of America*, vol. 2, no. 4, pp. 393–410, 1954.

[6] H. Ferreau, C. Kirches, A. Potschka, H. Bock, and M. Diehl, "qpOASES: A parametric active-set algorithm for quadratic programming," *Mathematical Programming Computation*, vol. 6, no. 4, pp. 327–363, 2014.

[7] Y. Gabriely and E. Rimon, "Spiral-stc: An on-line coverage algorithm of grid environments by a mobile robot," in *Robotics and Automation, 2002. Proceedings. ICRA'02. IEEE International Conference on*, vol. 1. IEEE, 2002, pp. 954–960.

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

[9] H. González-Baños, "A randomized art-gallery algorithm for sensor placement," in *Proceedings of the seventeenth annual symposium on Computational geometry*. ACM, 2001, pp. 232–240.

[10] K. Helsgaun, "An effective implementation of the lin–kernighan traveling salesman heuristic," *European Journal of Operational Research*, vol. 126, no. 1, pp. 106–130, 2000.

[11] S. Hert, S. Tiwari, and V. Lumelsky, "A terrain-covering algorithm for an auv," in *Underwater Robots*. Springer, 1996, pp. 17–45.

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

[13] ICARUS: Unmanned Search and Rescue, "http://www.fp7-icarus.eu/."

[14] S. Karaman and E. Frazzoli, "Sampling-based algorithms for optimal motion planning," vol. 30, no. 7, pp. 846–894, 2011.

[15] S. Lin and B. W. Kernighan, "An effective heuristic algorithm for the traveling-salesman problem," *Operations research*, vol. 21, no. 2, pp. 498–516, 1973.

[16] C. Luo, S. X. Yang, D. A. Stacey, and J. C. Jofriet, "A solution to vicinity problem of obstacles in complete coverage path planning," in *Robotics and Automation, 2002. Proceedings. ICRA'02. IEEE International Conference on*, vol. 1. IEEE, 2002, pp. 612–617.

[17] Open source implementation of the presented algorithm as a ROS package. [Online]. Available: https://github.com/ethz-asl/StructuralInspectionPlanner

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

[19] P. Oettershagen, A. Melzer, S. Leutenegger, K. Alexis and R. Siegwart , "Explicit model predictive control and l1-navigation strategies for fixed–wing uav path tracking," in *22nd IEEE Mediterranean Control Conference*, Palermo, Italy, 2014.

[20] P. Oettershagen, A. Melzer, T. Mantel, K. Rudin, R. Lotz, D. Siebenmann, S. Leutenegger, K. Alexis and R. Siegwart, "A solar-powered hand-launchable uav for low-altitude multi-day continuous flight," in *Robotics and Automation (ICRA), 2015 IEEE International Conference on*, May 2015, (accepted).

[21] G. Papadopoulos, H. Kurniawati, and N. M. Patrikalakis, "Asymptotically optimal inspection planning using systems with differential constraints," in *Robotics and Automation (ICRA), 2013 IEEE International Conference on*. IEEE, 2013, pp. 4126–4133.

[22] S. Leutenegger, A. Melzer, K. Alexis and R. Siegwart, "Robust state estimation for small unmanned airplanes," in *IEEE Multiconference on Systems and Control (MSC)*, Antibes, France, 2014.

[23] A. Zelinsky, R. A. Jarvis, J. Byrne, and S. Yuta, "Planning paths of complete coverage of an unstructured environment by a mobile robot," in *Proceedings of international conference on advanced robotics*, vol. 13, 1993, pp. 533–538.

Auton Robot
DOI 10.1007/s10514-015-9517-1

Three-dimensional coverage path planning via viewpoint resampling and tour optimization for aerial robots

Andreas Bircher[1] · **Mina Kamel**[1] · **Kostas Alexis**[1,2] · **Michael Burri**[1] · **Philipp Oettershagen**[1] · **Sammy Omari**[1] · **Thomas Mantel**[1] · **Roland Siegwart**[1]

Received: 2 July 2015 / Accepted: 15 October 2015

Abstract This paper presents a new algorithm for three-dimensional coverage path planning for autonomous structural inspection operations using aerial robots. The proposed approach is capable of computing short inspection paths via an alternating two-step optimization algorithm according to which at every iteration it attempts to find a new and improved set of viewpoints that together provide full coverage with decreased path cost. The algorithm supports the integration of multiple sensors with different fields of view, the limitations of which are respected. Both fixed-wing as well as rotorcraft aerial robot configurations are supported and their motion constraints are respected at all optimization steps, while the algorithm operates on both mesh- and occupancy map-based representations of the environment. To thoroughly evaluate this new path planning strategy, a set of large-scale simulation scenarios was considered, followed by multiple real-life experimental test-cases using both vehicle configurations.

Keywords Coverage planning · Aerial robots · Autonomous inspection

✉ Kostas Alexis
konstantinos.alexis@mavt.ethz.ch; kalexis@unr.edu

Andreas Bircher
andreas.bircher@mavt.ethz.ch

Mina Kamel
mina.kamel@mavt.ethz.ch

Michael Burri
michael.burri@mavt.ethz.ch

Philipp Oettershagen
philipp.oettershagen@mavt.ethz.ch

Sammy Omari
sammy.omari@mavt.ethz.ch

Thomas Mantel
thomas.mantel@mavt.ethz.ch

Roland Siegwart
rsiegwart@ethz.ch

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

[2] University of Nevada, Reno, NV, USA

1 Introduction

With the constant demand for higher automation in everyday processes, mobile robots permeate our daily life and are getting more and more utilized in civilian applications. Either in the form of aerial, maritime or ground vehicles, robots are already employed within critical tasks such as infrastructure monitoring, damage assessment or victim search. Indicative examples include those of bridge inspection (Metni and Hamel 2007), power-line monitoring (Kroll et al. 2009), boiler power-plant 3D reconstruction (Burri et al. 2012) and gas pipelines surveillance (Boon and Lovelace 2014). In most of such scenarios, the structure to be inspected may be provided in the form of a 3D model, typically a mesh or a voxels-based representation derived either using CAD software, Geographical Information Systems data or civil engineering instrumentation, or previous sensor-based reconstruction missions. Robots are then (re-)employed to derive an updated, possibly higher-fidelity model or scan for changes, risks and hazards (e.g. cracks). Among all possible different robot configurations, Unmanned Aerial Vehicles (UAVs) have attracted significant attention for a variety of structural inspection operations, for their ability to move in unstructured environments.

To facilitate the means for autonomous inspection, a robot must be equipped with the necessary accurate control units, the appropriate sensor systems and the relevant global path

Published online: 02 November 2015

planning intelligence. Such path planning algorithms should be able to lead to the quick computation of efficient paths that result in full coverage of the structure to be inspected, while respecting the on-board sensor limitations as well as the vehicle motion constraints that may apply. Due to the nature of the coverage problem, and despite the efforts of the community, its inherent difficulties still pose hard limitations on the performance, efficiency and practical applicability of the proposed solutions especially when 3D structures are considered. Furthermore, to the authors' best knowledge, only very few works validated their algorithmical contributions with real-life experiments, assessed their ability to be implemented on-board or evaluated the quality of the reconstruction results using the robot sensors.

Within this work, a novel, fast, iterative algorithm for three-dimensional structural coverage planning for aerial robotic inspection applications is proposed. The new path planning algorithm employs an alternating two-step optimization paradigm to compute good viewpoints that together provide full coverage while leading to a connecting path that has a low cost (e.g. distance, time-to-travel). Within every iteration, the set of updated viewpoint configurations is selected such that a) combining all viewpoints full coverage is achieved, and b) the cost-to-travel between the corresponding vehicle configuration and the neighboring viewpoint configurations gets reduced (*first step*). Subsequently, the optimally connecting and collision-free tour is recomputed (*second step*). The robots under consideration can integrate an arbitrary sensor system that is not limited to a single field of view, e.g. an aerial robot with two cameras facing in different directions. Not only the sensor's respective limitations like limited field of view, maximum and minimum practical observation distances are respected, but also the motion constraints of the vehicle. To evaluate the capabilities of the new structural inspection path planning algorithm, a variety of challenging and large-scale simulation test-cases were initially employed. Finally, the algorithm was thoroughly tested in practice based on flight experiments using both a small rotorcraft (hexarotor) UAV as well as a 5.6 m wingspan solar-powered fixed-wing UAV (both shown in Fig. 1). Analysis of the recorded flight results indicates the high-performance properties of the algorithm in challenging scenarios with complex inspection structures and subject to real vehicle and sensor constraints. Overall, this paper corresponds to a major extension of the authors' previous preliminary work in Bircher et al. (2015b) in terms of advanced sensor models that are not limited to a single field of view, different world representations of mesh and occupancy maps, improved collision checking mechanisms, implementation of algorithm updates and heuristic extensions. Additionally, extensive simulation studies using large-scale 3D scenarios and a whole new set of experimental studies of increased complexity are provided, as well as discussion of important implementation details and scientific documentation of our open-source released toolbox that may be found in Bircher and Alexis (2015). Furthermore, a rich and continuously updated dataset is released online (Bircher et al. 2015a). Finally, this paper also provides further details on the aerial robots employed, their autopilot, perception and computation properties and components.

This paper is structured as follows. A short overview of the related work and how our approach contributes further is presented in Sect. 2, followed by the problem description in Sect. 3. The proposed approach is presented in Sect. 4. Evaluation test-cases in simulation and experiments are shown in Sects. 5 and 6 while brief notes on the code and dataset released are provided in Sect. 7. Finally, conclusions are drawn in Sect. 8.

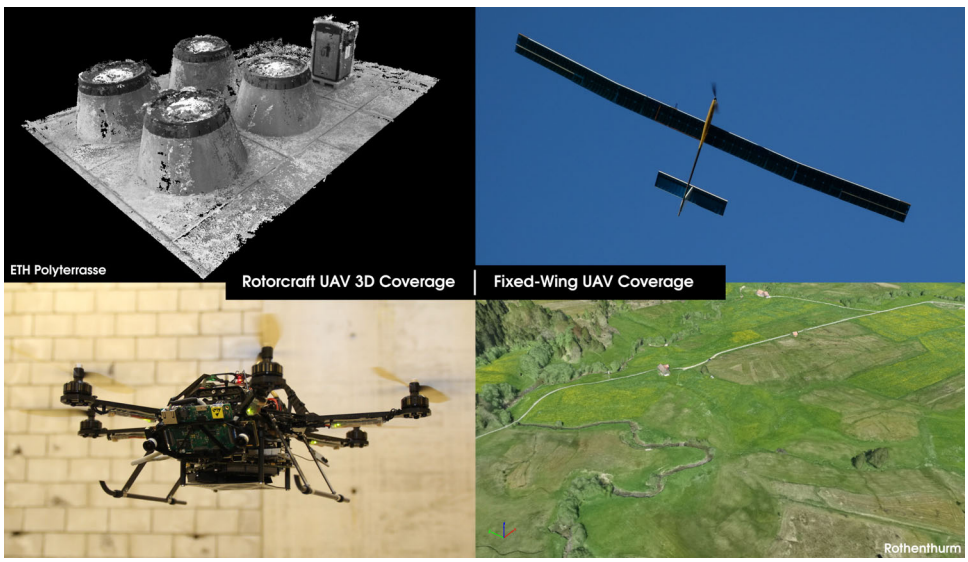


Fig. 1 Indicative inspection 3D reconstruction results using the proposed structural inspection planner and a rotorcraft as well as a fixed-wing UAV equipped with camera sensors

2 Related work

Due to the difficulty of coverage planning problems, early contributions relied on simplifications of the problem setup. Depending on the employed system and the environment, the space may be represented by a simplified discrete grid and paths can be computed using wavefront algorithms (Zelinsky et al. 1993), spanning trees (Gabriely and Rimon 2002) or neural networks (Luo et al. 2002). The work in Choset and Pignon (1998) proposed a cellular decomposition of the planning space, covering each free cell with a sweeping pattern. An advanced algorithm following this concept was presented in Acar et al. (2006). Also covering surface cells with a sweeping pattern, the work in Atkar et al. (2004) proposes a method for uniform coverage in spray painting applications. Employing a 2D planner, the authors in Hert et al. (1996) approximate a 3D structure using multiple 2D layers, treated individually. In order to enable versatile inspection path planning for real 3D structures, advanced algorithms have recently been proposed. Within the most recent contributions, those that employ a two-step optimization scheme proved to be more versatile with respect to the inspection scenario. The first step of such algorithms is to compute the minimal set of viewpoints that covers the whole structure. This corresponds to solving an art gallery problem (AGP), a NP-hard but well studied problem. Fast algorithms to solve this problem are presented for example in O'rourke (1987); González-Baños (2001). As a second step in these coverage planning approaches, the shortest connecting tour over all these viewpoints has to be computed, which is the traveling salesman problem (TSP). Also this problem is NP-hard, but fast algorithms have long been proposed, e.g. (Dantzig et al. 1954; Lin and Kernighan 1973). A recent application of these concepts, that allows some redundancy in the AGP such that it is able to improve the path in a post-processing step, was presented in Hover et al. (2012). This algorithm can deal with 3D scenarios and is demonstrated in experiments using underwater vehicles for ship hull inspection. Addressing the problem from a different perspective, the work in Papadopoulos et al. (2013) concentrates on deriving close-to-optimal solutions at the inherently large cost of computational efficiency. A comprehensive survey of the existing coverage path planning methods may be found in Galceran and Carreras (2013).

The proposed fast inspection path planner retains a two-step optimization structure, but contrary to trying to find a minimal set of viewpoints in the AGP it rather tries to sample them such that the connecting path is short while ensuring full coverage. This is driven by the idea that with a continuously sensing sensor the number of viewpoints (and if this is minimal or not) is not necessarily important, but mostly their configuration in space, which has to be such that short paths are provided. As a result, this novel approach leads to full coverage paths that are of low cost and are computed in short time.

3 Problem description

The problem of structural inspection path planning is, within the scope of this work, defined as that of the challenge to find a high quality path that guarantees the complete coverage of a given 3D structure subject to dynamic constraints of the vehicle and limitations of the on-board sensor system. The 3D structure to be inspected may be represented in a computationally efficient way such as a triangular mesh or a voxel-based octomap and is embedded into a bounded environment that may contain obstacle regions. The problem setup is to be such that for each triangle in the mesh, there exists an *admissible* viewpoint configuration—that is a viewpoint from which the triangle is visible for a specific sensor model. Then, for the given environment and with respect to the operational constraints, a path connecting all viewpoints has to be found which guarantees complete inspection of the 3D structure. Quality measures for paths are situation specific, depending on the system and mission objectives, e.g. short time or distance.

As sample systems we consider a rotorcraft and a fixed-wing UAV, both equipped with one (or more) visual cameras with a fixed orientation relative to the platform. Minor adaptations to the proposed path-planner enable its use for other common robot configurations such as underwater ROVs or wheeled robots. The proposed approach also works with multi-camera setups or even omnidirectional sensors.

4 Proposed approach

As the algorithm does not focus on minimizing the number of viewpoints, the proposed approach selects one admissible viewpoint for every triangle in the mesh of the structure to be inspected. In order to compute viewpoints that allow low-cost connections, an iterative resampling scheme is employed. Between each resampling, the best path connecting the current viewpoints is recomputed. The quality of the viewpoints is assessed by the cost to connect to their respective neighbors on the latest tour. This cost is minimized in the subsequent resampling, resulting in locally optimized paths. Initialization of the viewpoints is arbitrarily done such that full coverage is provided with, at this stage, non-optimized viewpoints. A fast implementation of the Lin–Kernighan–Helsgaun Heuristic (LKH) TSP solver (Helsgaun 2000) is employed to compute the best tour, while the cost of the interconnecting pieces of path is calculated by means of a boundary value solver (BVS). The iterative search for better

paths is terminated after a maximum of $N_{\max}$ iterations. An overview of the proposed inspection planning procedure is presented in Algorithm 1.

Algorithm 1 Structural Inspection Planner

```
k ← 0
Sample initial viewpoint configurations
Compute cost matrix for the TSP solver (Sect. 4.1)
Solve the TSP problem to obtain initial tour
while k < N_max do
    Resample viewpoint configurations (Sect. 4.2)
    Recompute the cost matrix (Sect. 4.1)
    Recompute best tour T_best using the LKH and update best tour
    cost c_best if applicable
    k ← k + 1
end while
return T_best, c_best
```

In the following, the path generation, as well as the optimization problem formulation to sample the viewpoints for a rotorcraft UAV are described. Adaptations to plan for a fixed-wing UAV are discussed in Sect. 4.3. Subsequently, some extensions for collision avoidance and heuristic speed-up are described.

4.1 Path computation and cost estimation

In order to find the best tour among the viewpoints, the TSP solver requires a cost matrix containing the connection cost of all pairs of viewpoints. A two state BVS is employed to generate paths and estimate the respective costs. The BVS is either employed directly to connect the two viewpoints or as a component in a local planner. The latter applies in case the direct connection is not feasible due to obstacles. In that case, our implementation makes use of the RRT*-planner (Karaman and Frazzoli 2011) to find a collision-free connection. The state of the proposed model for a rotorcraft UAV consists of position as well as yaw, $\xi = \{x, y, z, \psi\}$. As slow maneuvering is desired to achieve increased inspection accuracy (in terms of localization as well as control loop robustness and performance), the roll and pitch angles are considered to be near zero during the path planning computation process. The path from configuration ξ_0 to ξ_1 is given by $\xi(s) = s\xi_1 + (1-s)\xi_0$, where $s \in [0, 1]$. The single limitation considered is the speed limit. The translational limit is denoted by v_{max} while the rotational speed is limited by $\dot{\psi}_{max}$. To allow sufficiently accurate tracking of paths with corners, both values are small. The resulting execution time is $t_{ex} = \max(d/v_{max}, \|\psi_1 - \psi_0\|/\dot{\psi}_{max})$, with d the Euclidean distance. The considered cost metric of a path segment corresponds to the execution time t_{ex}.

4.2 Viewpoint sampling

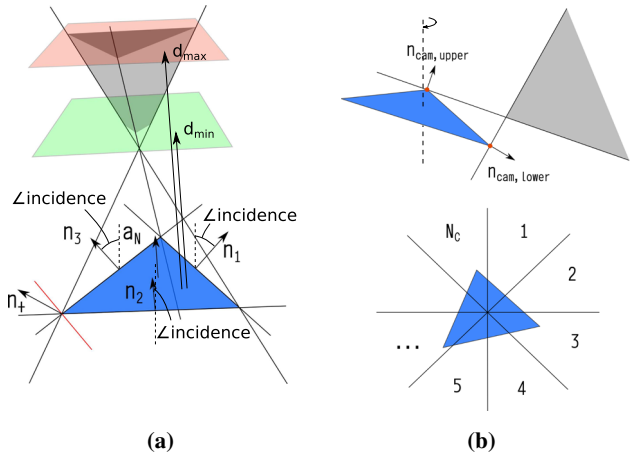
Considering a mesh-based representation of the environment, a viewpoint has to be sampled for each of its triangular facets. In the proposed procedure, its position and heading is determined sequentially while retaining visibility of the corresponding triangle. First, the position is optimized for distance w.r.t. the neighboring viewpoints using a convex problem formulation and only then, the heading is optimized. To guarantee a good result of this multistep optimization process, the position solution must be constrained such as to allow finding an orientation for which the triangle is visible.

More specifically, the constraints on the position $g = [x, y, z]$ consist of the inspection sensor limitations of minimum incidence angle, minimum and maximum range (d_{min}, d_{max}) constraints (depicted in Fig. 2a). They are formulated as a set of planar constraints:

$$\begin{bmatrix} (g - x_i)^T n_i \\ (g - x_1)^T a_N \\ -(g - x_1)^T a_N \end{bmatrix} \succeq \begin{bmatrix} 0 \\ d_{min} \\ -d_{max} \end{bmatrix}, i = \{1, 2, 3\} \tag{1}$$

where x_i are the corners of the mesh triangle, a_N is the normalized triangle normal and n_i are the normals of the separating hyperplanes for the incidence angle constraints as shown in Fig. 2a.

Further, the sensor has a limited field of view (FoV) with a certain vertical and horizontal opening and is mounted to the system with a fixed pitch angle and relative heading. As illustrated in Fig. 2b, in 2D the angle constraining the lower bound of the FoV constrains the upper bound of the sampling space, together with the most restrictive corner of the facet. With the equivalent constraint from the upper bound of the FoV, sampling is constrained to a triangular section. The total sampling space resulting from the vertical camera opening constraint is the union of all these triangular sections over all horizontal directions, which is not convex. To approximate and convexify the problem, the space is divided in N_C equal convex pieces according to Fig. 2b. The optimum is computed for every slice in order to find the globally best solution. Multiple sensors with different vertical FoVs are handled equally, resulting in a multiple of N_C convex pieces that possibly overlap. The constraints for piece j are derived as follows: Left and right boundaries of the sampling space are the borders of the revolution segment and the cone top and bottom are represented by a single plane tangential to the centre of the slice. Angular camera constraints in horizontal direction are not encoded and instead d_{min} is chosen high enough to allow full visibility of the triangle. This leaves some space for variation in the sampling of the heading, where the horizontal constraints are enforced. Specifically, the abovementioned constraints are:



(a) **(b)**

Fig. 2 **a** Incidence angle constraints on a triangular facet. The figure depicts the three main planar angle of incidence constraints on all three sides of the triangle. For a finite number of such constraints the incidence angle is only enforced approximately. The *red line* (and n_+) demarks a sample orientation for a possible additional planar constraint at a corner. Minimum (*green plane*) and maximum (*red plane*) distance constraints are similar planar constraints on the sampling area. These constraints bound the sampling space, where g can be chosen, on all sides (*gray area*). **b** Camera constraints and convexification. The vertical camera angle constraints with the relevant corners of the *triangle in red* are depicted in the *upper part*. In 2D they constrain the sampling space to a triangular region, the union over all horizontal direction of which is not convex. *Beneath* the partition of the space for convexification is depicted (Color figure online)

$$\begin{bmatrix} (g - x^{rel}_{lower})^T n^{cam}_{lower} \\ (g - x^{rel}_{upper})^T n^{cam}_{upper} \\ (g - m)^T n_{right} \\ (g - m)^T n_{left} \end{bmatrix} \succeq \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \end{bmatrix}, \tag{2}$$

where $x^{rel}_{lower}, x^{rel}_{upper}$ are the respective relevant corners of the mesh triangle, m the middle of the triangle and n^{cam}_{lower}, n^{cam}_{upper}, n_{right} and n_{left} denote the normal of the respective separating hyperplanes.

The optimization objective for the viewpoint sampling in iteration k, in the case of a rotorcraft UAV, is to minimize the sum of squared distances to the preceding viewpoint g^{k-1}_p, the subsequent viewpoint g^{k-1}_s and the current viewpoint g^{k-1} in the old tour. The former two parts potentially shorten the tour by moving the viewpoints closer together, while the latter limits the size of the improvement step, as g^{k-1}_p and g^{k-1}_s potentially move closer as well. The weighting matrix B for the neighbor distance is given by $\mathbf{diag}(b_{const}, b_{const}, a_{const} + b_{const})$, where b_{const} is the general weight for distance to neighbors, while a_{const} additionally punishes changes in height. The distance to the current viewpoint in the old tour is weighted by the matrix $D = \mathbf{diag}(d_{const}, d_{const}, d_{const})$.

The resulting convex optimization problem is given below. Its structure as a quadratic program (QP) with linear constraints allows the use of an efficient solver (Ferreau et al. 2014).

$$\min_{g^k} \; (g^k - g^{k-1}_p)^T B (g^k - g^{k-1}_p) + (g^k - g^{k-1}_s)^T B (g^k - g^{k-1}_s) + (g^k - g^{k-1})^T D (g^k - g^{k-1}) \tag{3}$$

$$\text{s.t.} \quad \begin{bmatrix} n_1^T \\ n_2^T \\ n_3^T \\ a_N^T \\ -a_N^T \\ n^{cam\,T}_{lower} \\ n^{cam\,T}_{upper} \\ n^T_{right} \\ n^T_{left} \end{bmatrix} g^k \succeq \begin{bmatrix} n_1^T x_1 \\ n_2^T x_2 \\ n_3^T x_3 \\ a_N^T x_1 + d_{min} \\ -a_N^T x_1 - d_{max} \\ n^{cam\,T}_{lower} x^{rel}_{lower} \\ n^{cam\,T}_{upper} x^{rel}_{upper} \\ n^T_{right} m \\ n^T_{left} m \end{bmatrix} \tag{4}$$

For the computed optimal position, the heading is determined according to the criterion $\min_{\psi^k} = \left(\psi^{k-1}_p - \psi^k\right)^2 / d_p + \left(\psi^{k-1}_s - \psi^k\right)^2 / d_s$, s.t. $\mathbf{Visible}(g^k, \psi^k)$, where $\mathbf{Visible}(g^k, \psi^k)$ means that from the given configuration, g^k and ψ^k, the whole triangle is visible for at least one of the employed sensors. d_p and d_s are the Euclidean distances from g^k to g^{k-1}_p and g^{k-1}_s respectively. For simple sensor setups establishing the boundaries on ψ^k for $\mathbf{Visible}(g^k, \psi^k) = \text{TRUE}$ makes the solution explicit. Otherwise a grid search can be employed.

4.3 Extensions for fixed-wing UAVs

Fixed-wing UAVs correspond to another excellent configuration for inspection operations. However, their advantages in aspects like long-endurance, come together with limitations on handling sharp turns, steep ascents or descents. Moreover, the direction of a fixed camera is necessarily related to the direction of travel. Accordingly, the implementations for the BVS and the viewpoint sampling have to be adapted.

Assuming that highly dynamic maneuvers are avoided for inspection flights, the minimum turn radius of the aircraft is constrained to be $r_{\min}$ while pitch and roll (as well as bank) are considered to be near zero. For planning purposes, the xy-plane vehicle dynamics are captured using Dubins curves, thus minimizing the distance w.r.t. $r_{\min}$. Furthermore, in the vertical direction the path is constrained by a maximum climb and sink rate. Since these values are small, instantaneous changes are acceptable and the rate $\dot{z}$ is chosen to be constant along a path segment. If the maximum rate is exceeded, ascending/descending loitering circles are added at the end of the path segment to allow larger changes of height. In many practical cases such as flat landscape coverage, it makes sense to constrain the height of the path to a fixed value to avoid undesirable loitering circles. The fixed-wing UAV is assumed to travel with constant velocity v_{FW}, and the path cost is the time $t_{ex} = l_{Path}/v_{FW}$, with l_{Path} the path length.

In contrast to the case of rotorcraft UAVs, where only the distance is minimized in the viewpoint position sampling step, the fixed-wing UAV sampler also aims to align the viewpoints on a as straight line as possible. This effectively avoids too many curly path segments and thus, together with the distance minimization tends to reduce the path length. The addition in the objective is therefore to minimize the squared distance d^2 to the straight line between the neighboring viewpoints. Using its direction vector b, the distance is calculated as follows:

$$b = \frac{g_s^{k-1} - g_p^{k-1}}{\left\| g_s^{k-1} - g_p^{k-1} \right\|}$$

$$d = \left\| b \times (g^k - g_p^{k-1}) \right\| = \left\| \begin{bmatrix} 0 & -b_3 & b_2 \\ b_3 & 0 & -b_1 \\ -b_2 & b_1 & 0 \end{bmatrix} (g^k - g_p^{k-1}) \right\| \tag{5}$$

and with

$$\begin{bmatrix} 0 & -b_3 & b_2 \\ b_3 & 0 & -b_1 \\ -b_2 & b_1 & 0 \end{bmatrix} (g^k - g_p^{k-1}) = q \tag{6}$$

follows $d^2 = q^T q$. To avoid the insertion of unnecessary circles, the distance between the viewpoints has to be large enough according to their heading, the direction to the next viewpoint and $r_{\min}$. The bounds on that distance $l_i,\ i = \{p, s\}$ are derived geometrically and evaluated using numerical algorithms. The distance criteria are therefore $(g^k - g_i^{k-1})^T (g^k - g_i^{k-1}) \geq {l_i}^2,\ i = \{p, s\}$ which are non-convex. To convexify, the criteria are linearized around the old viewpoint. This adaptation is conservative by the exclusion of the non-convex part and attenuates the impact of the extrapolation error:

$$\left(g^k - g_i^{k-1}\right)^T \left(g^{k-1} - g_i^{k-1}\right) \geq {l_i}^2,\ i = \{p, s\} \tag{7}$$

Adding the two slack variables ϵ_p and ϵ_s with constant weight C to allow occasional violation of the minimal distance criterion results in the following QP-formulation:

$$\min_{q, g^k} \quad \text{(objective in (3))} + q^T q + C(\epsilon_p + \epsilon_s) \tag{8}$$

$$\text{s.t.} \quad \begin{bmatrix} 0 & -b_3 & b_2 \\ b_3 & 0 & -b_1 \\ -b_2 & b_1 & 0 \end{bmatrix} (g^k - g_p^{k-1}) = q$$

$$\begin{bmatrix} \text{constraints in (4)} \\ (g^{k-1} - g_p^{k-1})^T \\ (g^{k-1} - g_s^{k-1})^T \end{bmatrix} g^k \geq \begin{bmatrix} \text{constraints in (4)} \\ {l_p}^2 + (g^{k-1} - g_p^{k-1})^T g_p^{k-1} - \epsilon_p \\ {l_s}^2 + (g^{k-1} - g_s^{k-1})^T g_s^{k-1} - \epsilon_s \end{bmatrix}$$

$$\epsilon_p \geq 0$$

$$\epsilon_s \geq 0 \tag{9}$$

The criterion of Eq. 7 is inverted for the heading computation and applied as long as a feasible solution is found.

Recall the weight of the distance to neighbor viewpoints $B = \mathbf{diag}(b_{const}, b_{const}, a_{const} + b_{const})$. While in the rotorcraft case a_{const} has a minor role, for the fixed-wing planning it provides an effective means to enforce minimization of height difference and thus has a major impact on the path quality. A high a_{const} reflects the cost of loitering circles to gain or lose height.

4.4 Obstacle avoidance

The proposed approach also works efficiently in cluttered environments. Obstacle avoidance is easily achieved by integrating a collision check in the BVS, that prevents connections crossing facets of the mesh. In order to shape a scenario according to requirements, additional cuboidal obstacle regions can be defined, representing obstacles that do not belong to the structure to inspect. To make sure that the facets are visible from their viewpoints, visibility is checked for occlusion by parts of the mesh. If interference is detected, the QP is reformulated to include an additional planar constraint defined by a corner x^I of the interfering facet triangle, the centre of which is closest to the facet, as well as its normal a_N^I:

$${a_N^I}^T g^k \geq {a_N^I}^T x^I \tag{10}$$

For meshes that do not self-intersect, this is a conservative exclusion of cluttered subareas. Depending on whether a cuboidal obstacle region is some structure or just a no-fly zone, it is also included in the visibility check. When, during the viewpoint sampling, such an obstacle structure is encountered, the optimization problem is executed individually over eight subregions. Each is bounded away from the cuboid on one of its sides, similar to Eq. 10.

4.5 Additional heuristic concepts

Additional heuristic measures increase the quality of computed paths. Primarily for the rotorcraft UAV path planning additional heuristics can speed up the improvement. To allow a faster and more rigorous ordering of the viewpoints, initial iterations of the algorithm consider not only the nearest neighbor on the tour to minimize the distance to, but the neighbors that are $N_{Neighbor}$ away on both sides. In every iteration $N_{Neighbor}$ is then decremented to finally reach 1.

5 Simulation studies

Three simulation test-cases demonstrate the applicability of the proposed algorithm to typical inspection problems. The

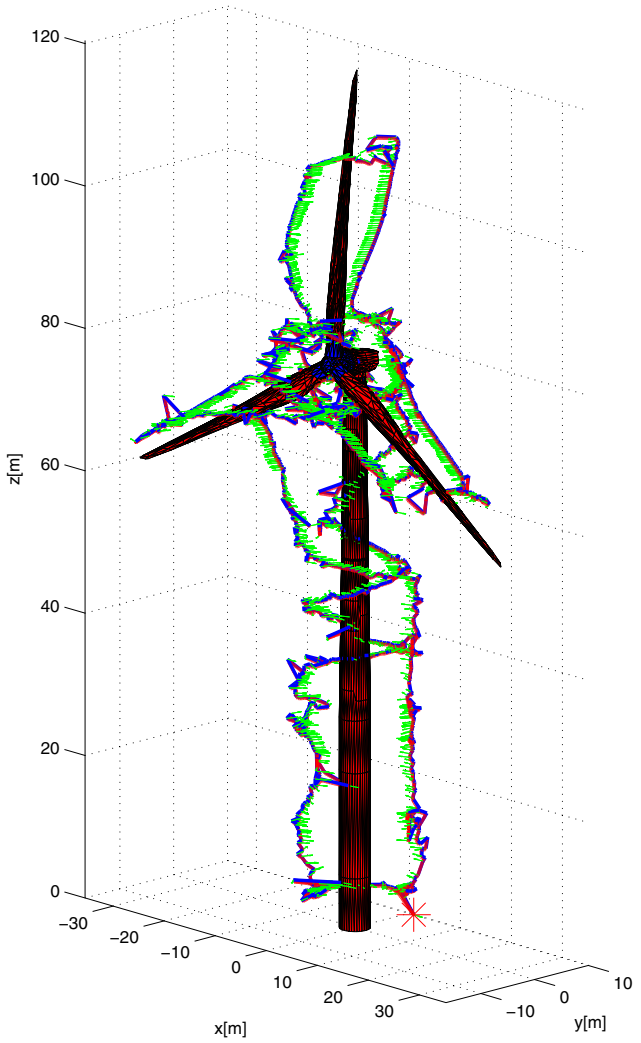


Fig. 3 Wind turbine path (FoV: [120, 90]°, $d_{\min}$: 4 m, ∠incidence: 35°) simulated in the Gazebo-based simulator RotorS (RotorS). The mesh of the structure is depicted, of which the *red part* has to be inspected. The computed path is plotted in *blue*, with the *green arrows* indicating the optimized viewpoints. The closely following *red line* corresponds to the simulated vehicle response (Color figure online)

utilization in large scenarios as well as use-cases with different system setups are presented.

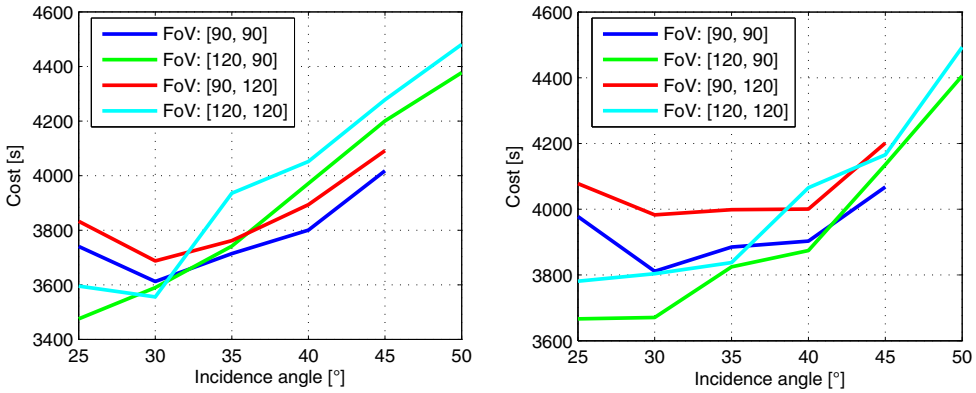
5.1 Wind turbine simulation scenario

As a realistic large scale simulation test-case for rotorcraft inspection, a mesh model of a 120 m high wind turbine (see Fig. 3) was used (Grabcad). Indeed a large scale scenario, such as the inspection of a wind farm, facilitates the need for optimized paths such that the mission can be conducted with the endurance-limited small but safe aerial robots without the need to land and change batteries for multiple times. The employed mesh is split in two parts, one comprising the autonomously inspectable part (red), while the other contains surfaces close to the turbine's mechanisms (blue), unsuitable for visual inspection from distances of multiple meters. For the first subset of the mesh with 2220 triangular facets inspection paths are computed. In order to examine the influence of different parameters on the path quality, multiple executions are performed. Various fields of view are employed, ranging from 90 to 120° in both vertical and horizontal direction. The enforced minimal incidence angle is varied from 25° to 50° with increments of 5°. Finally, paths are computed for minimally enforced inspection distances to the structure of 2.5 and 4 m, while the maximal distance is set at 10 m for all parameter sets. Using randomized initial viewpoints, 10 paths were computed for all parameter sets in order to obtain indicative statistics. For the inspection mission, a rotorcraft vehicle is assumed which is subject to a maximum allowed linear velocity of $v_{\max} = 0.5$ m/s and a maximum yaw rate $\dot{\psi}_{\max} = 0.75$ rad/s while it carries the camera sensor mounted with 0° pitch. The test-case parameters are summarized in Table 1. The turbine and the derived inspection path for a sample parameter set are illustrated in Fig. 3. The computation time in this specific case was 260 s. Also depicted is the vehicle response of a sample simulation run in the RotorS simulator (RotorS). RotorS is an open-source Gazebo-based aerial robots simulator that not only simulates the vehicle dynamics but also a variety of sensors such as the IMU, a generic odometry sensor as well as a complete and highly integrated system called visual–inertial sensor (VI-Sensor) (Skybotix) (for more information refer to Sect. 6.1.1). In addition to the vehicle response, such a simulation also reveals the collision free quality of the path, as physical interaction models between objects are included. Due to the sparse optical features in the simulation, the quality of the reconstruction from the recorded data is limited and therefore not included.

The employed path is computed with the specific parameters of a FoV = [120, 90]°, minimum inspection distance $d_{\min} = 4$ m and angle of incidence ∠incidence = 35°. The complete statistics for all parameter sets are depicted in Fig. 4. The left figure for $d_{\min} = 2.5$ m shows clear trends for varying incidence angle constraints. With a large FoV in vertical direction, an approximately linear correlation between the chosen incidence angle and path cost is found. When the vertical FoV is smaller, it constrains the path quality for lower angles of incidence, where the curves flatten. This is due to the fact, that most facets are almost vertical and large incidence angle and limited vertical FoV result in similar planar constraints on the sampling space. At the same time larger angles of incidence in combination with a limited vertical FoV render the problem infeasible because horizontal facets are not visible with the required accuracy. No significant influence of the horizontal FoV is found in the investigated range. This is because it can be compensated by yawing which is inexpensive considering the rather large dis-

Table 1 Wind turbine scenario

N_{facets}	2220	∠incidence	25−50°
FoV	[90, 90]°, [120, 90]°, [90, 120]°, [120, 120]°	Mouting pitch	0°
$d_{\min}$	2.5, 4 m	$d_{\max}$	10 m
$v_{\max}$	0.5 m/s	$\dot{\psi}_{\max}$	0.75 rad/s



(a) $d_{\min} = 2.5$m **(b)** $d_{\min} = 4$m

Fig. 4 The influence on path quality of different parameters in the wind turbine scenario is investigated. The *left plot* depicts the mean path lengths over 10 runs, depending on incidence angle for different fields of view. The minimal inspection distance corresponds to 2.5 m. The *right plot* contains the same information, but for a minimal inspection distance of 4 m

tances between the viewpoints. Similar results are found for a minimal inspection distance of 4 m, although path costs are generally larger and the influence of the constraints smaller. For structures that are mostly locally convex, this result can be expected.

5.2 Staircase inspection scenario

A second simulation experiment investigates the use of the proposed planner for multi-camera setups. As an inspection scenario, a staircase is considered. A first model of the environment is derived by exploring the structure using the handheld VI-sensor and computing an occupancy map of the obtained synchronized stereo vision data and pose estimates. A part of the staircase, as well as its occupancy map counterpart are depicted in Fig. 5a. From the occupancy map a mesh is derived, representing the surface of the occupied voxels. As high fidelity of the resulting mesh relies on a high resolution of the occupancy map, the number of facets may very fast grow beyond feasibility. Therefore the mesh is subsampled and only then used to compute the inspection paths. This processing pipeline allows the convenient derivation of a first model to plan an inspection path.

This case-study assumes the same vehicle as in the previous one, however, three different camera setups are considered for comparison. The first uses a single camera with a FoV of [120, 120]° and zero pitch, while the others employ a multi-camera setup. One has an additional camera with the same specifications facing in the opposite angular direction, while the last employs a 360° setup, featuring four such cameras arranged in 90° spacing. A minimum angle of incidence of 20° is enforced along with an inspection distance between 0.3 and 1.5 m. The allowed traveling speeds are $v_{\max} = 0.5$ m/s and $\dot{\psi}_{\max} = 0.5$ rad/s. The mesh was used for occlusion and collision detection in all steps of the algorithm. A summary of the employed parameters can be found in Table 2. Figure 5b depicts the cost evolution over the course of 20 planner iterations for all three setups. While in all cases high quality paths could be found, larger FoVs obviously result in lower path costs. The inspection paths after 20 iterations, together with the derived mesh, are depicted in Fig. 6 and have a final cost of 217.3, 185.6 and 150.7 s for the three considered systems respectively. Due to the complex structure, which makes it difficult to find collision-free connections between the viewpoints, the computation for the three paths lasted 288, 357 and 266 s.

.

5.3 Fixed-wing UAV large-scale landscape coverage mission

A third test case, this time relevant with the fixed-wing UAV, consists of a 33 km^2 landscape around 4158 m Jungfrau and 4107 m Mönch, two mountains in the Swiss Alps. The inspiration for this scenario is a rapid-response search mission of lost mountaineers, thus requiring detailed and gapless mapping. The chosen area reaches from the mountains down into the valleys with height differences of more than 2400 m. Its steep climbs, in some places almost vertical, render simplified approaches like lawn-mower patterns useless.

From the high-resolution 3D map provided by Swisstopo, a downsampled mesh was derived, containing 657 facets.

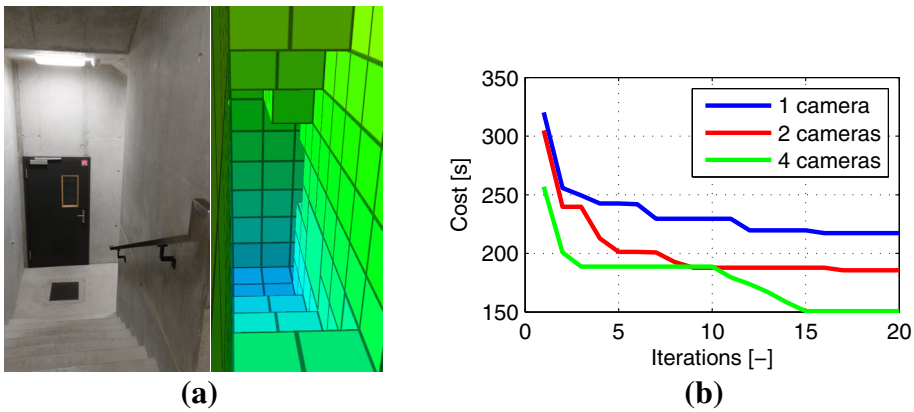


Fig. 5 The *left* figure depicts a part of the considered staircase, both as photo and occupancy map derived from the recorded preliminary data of the VI-sensor. The plots on the *right side* show the correlation between the cost of the obtained paths and the number of employed sensors. In all cases, more sensors improve the resulting path cost. **a** Staircase and derived occupancy map. **b** Cost plots

Table 2 Staircase inspection scenario

N_{facets}	180	∠incidence	20°
FoV	[120, 120]°	Mouting pitch	0°
$d_{\min}$	0.3 m	$d_{\max}$	1.5 m
$v_{\max}$	0.5 m/s	$\dot{\psi}_{\max}$	0.5 rad/s

This mesh was used to compute a short path that gives full coverage. Furthermore, it was also used for occlusion and collision checking in all steps of the computation. The system under consideration has a minimum turning radius constraint of 60 m, a flight speed of 10 m/s and a maximum climb- and sink-rate of 1 m/s. The employed sensor has a FoV of [120, 120]° and is mounted with a relative pitch of 25°. In order to ensure good quality of the coverage, the incidence angle is constrained to 30° and the enforced distance lies between 300 and 400 m. This ensures high enough quality of the recorded sensor data to enable reconstruction of the inspected landscape in adequate resolution to be used for victim search or similar tasks. The mission parameters are summarized in Table 3. The computation lasted for 1916 s, resulted in an inspection path that has a total cost of 516 min and is depicted in Fig. 7. Obviously, the limit on the climb- and sink-rate makes it strenuous to overcome the vast altitude differences, as they translate to minimally required horizontal path lengths. However, the computed path takes advantage of this by covering the mountainside along the way up or down as highlighted in Fig. 7. This effectively reduces the cost of the computed path.

Overall, these three very different inspection scenarios emphasize the versatility of the presented planner with respect to different mission and system setups. By varying the mission parameters, the resulting inspection path can be shaped to suit the mission requirements. This, and the complexity of the presented scenarios, reveal the wide range of use-cases, where the planner can be employed. Its applica-

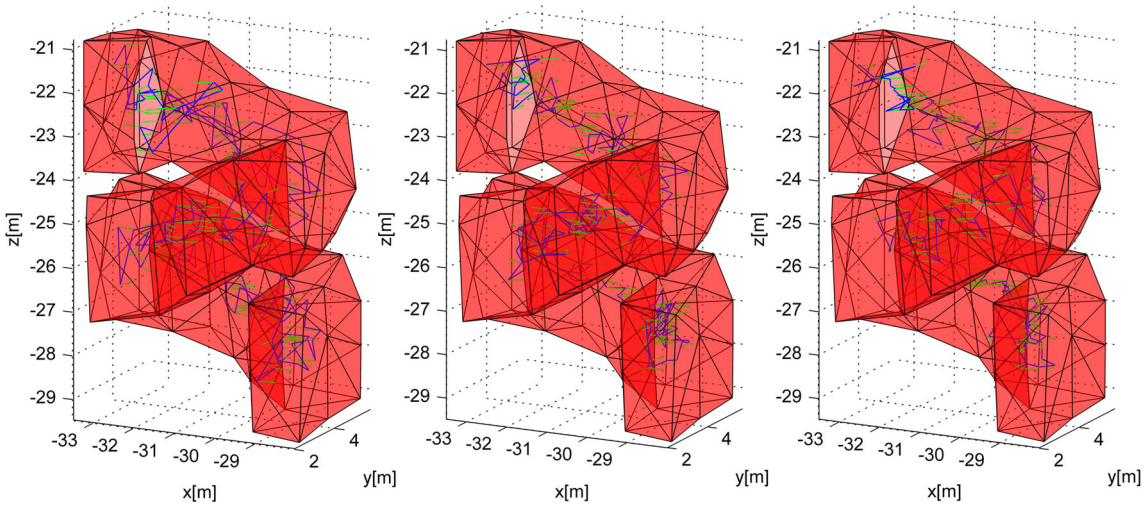


Fig. 6 Paths for the staircase scenario have been computed using three different system setups. The first (*left*, cost 217.3 s) employs one vision sensor looking to the front. A second system (*middle*, cost 185.6 s) additionally features a sensor looking to the back and the third (*right*, cost 150.7 s) also makes use of one sensor to each side, resulting in a 360° field of view in *horizontal* direction. The paths (*blue*) are plotted together with the mesh used for planning (*red*) and the viewpoints (*green*) each of which is looking at a certain facet (Color figure online)

Table 3 Jungfrau & Mönch inspection scenario

N_{facets}	657	∠incidence	30°
v_{FW}	10 m/s	$r_{\min}$	60 m
FoV	[120, 120]°	Mouting pitch	25°
$d_{\min}$	300 m	$d_{\max}$	400 m

bility to real world inspection missions is further evaluated in experimental studies presented in Sects. 6.1 and 6.2.

6 Experimental evaluation

To evaluate the real-life efficiency of the algorithm, a set of experiments was conducted using both rotorcraft and fixed-wing UAVs. For each of them, a smaller and a larger-scale scenario was considered while in all cases the robots were relying solely on on-board sensor data operating with and without GPS-support for the fixed-wing and rotorcraft cases correspondingly. For all conducted experiments, the post-processed results are available as a public dataset to allow direct assessment of the quality of the derived results.

6.1 Rotorcraft UAV inspection operations

Extensive experimental studies were conducted using a highly autonomous MAV system developed around the AscTec Firefly Hexacopter and the VI-Sensor developed by our lab and Skybotix AG. The overall system, its hardware specifications, control properties, perception capabilities and autonomy levels are briefly overview in Sect. 6.1.1. Subsequently, the inspection path planning results are thoroughly discussed in Sects. 6.1.2 and 6.1.3.

6.1.1 MAV platform overview

As mentioned earlier in this paper, the first employed aerial robotic platform is built around an AscTec Firefly Hexacopter MAV on-board of which the VI-Sensor developed by our lab and Skybotix AG is further integrated. The VI-Sensor integrates a stereo camera pair consisting of two HDR global shutter cameras (Aptina MT9V034) and an Analog Devices ADIS16448 IMU that are tightly aligned and synchronized using an Artix^{TM}-7 FPGA, a Xilinx Zynq 7020 SoC module and an ATOM CPU running Linux. This integrated sensor system runs advanced image processing algorithms and in combination with the employed state estimation algorithms, it provides accurate and complete pose estimates while it simultaneously builds a 3D dense reconstruction of the environment. Provided the full state estimate feedback, the aerial robot implements a cascaded trajectory controller consisting of a linear model predictive control (MPC) on the position dynamics that provides thrust commands and attitude references to the low-level attitude control of the vehicle which is implemented on-board the AscTec electronics. Figure 8 provides an overview of these loops alongside with a photo of the vehicle and its computational components.

The AscTec Firefly Hexacopter UAV comes with a reliable factory tuned attitude controller, on top of which a trajectory tracking controller based on MPC (Alexis et al. 2012; Camacho and Bordons 2003) approach was designed. The dynamics of the closed loop attitude dynamics (roll and pitch) have been identified using system identification techniques

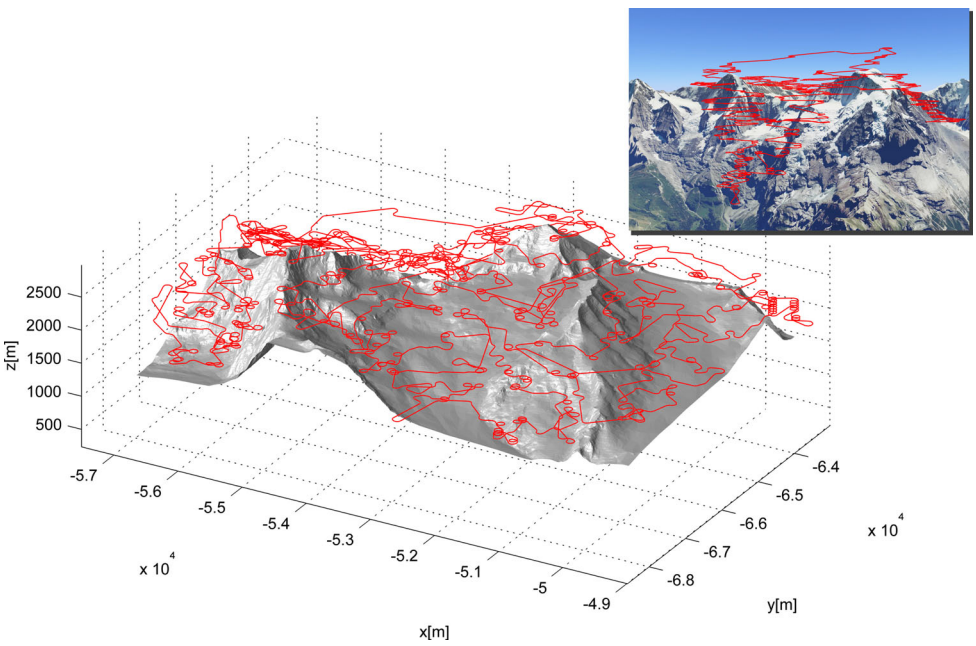


Fig. 7 A 33 km^2 area in the Swiss Alps is considered as the inspection structure. The path for the employed fixed-wing system has a total cost of 516 min. The *left* figure depicts the total considered landscape area with the computed path on transformed ENU local coordinates, while the *right* side reveals the monotonously rising and sinking path. Due to the large altitude changes of up to 2400 m, the solution to the inspection path planning problem is dominated by zig–zagging and circling path segments. (Source of height model: Swiss Federal Office of Topography)

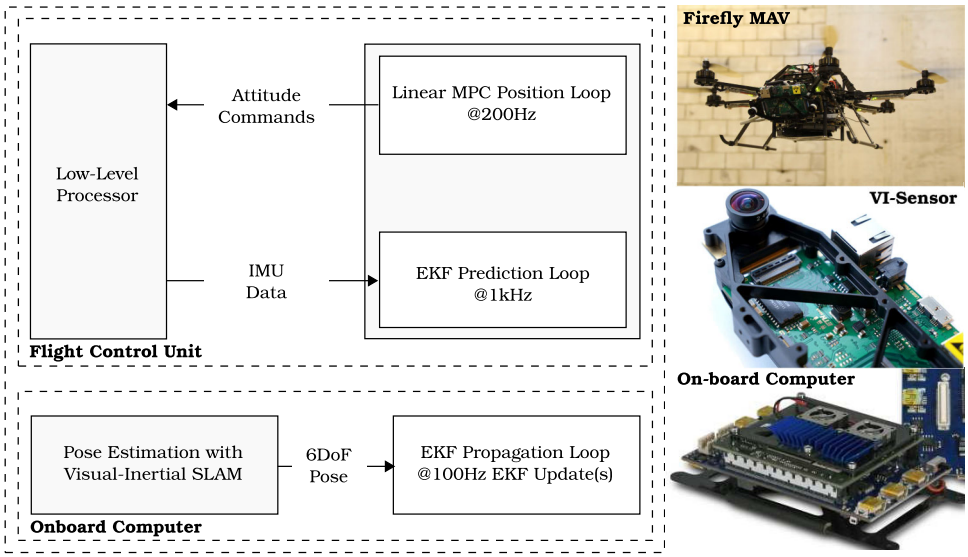


Fig. 8 The firefly UAV main software and hardware components

and are approximated by a first order system. To achieve an offset-free reference tracking, a disturbance observer based on a Kalman Filter is designed and tuned to provide external disturbance estimation $\hat{\mathbf{d}}$ with the assumption that external disturbances are constant over the prediction horizon. To formulate the receding horizon optimization problem, let the state vector be $\mathbf{x} = \begin{bmatrix} x & y & z & \dot{x} & \dot{y} & \dot{z} & \phi & \theta \end{bmatrix}^T$, the control input vector be $\mathbf{u} = \begin{bmatrix} \phi_{cmd} & \theta_{cmd} & T_{cmd} \end{bmatrix}^T$ and the disturbance vector be $\mathbf{d} = \begin{bmatrix} d_x & d_y & d_z \end{bmatrix}^T$, where x, y, z indicates the position of the hexacopter, ϕ, θ are roll and pitch angles respectively, while ϕ_{cmd}, θ_{cmd} and T_{cmd} are the roll command, pitch command and thrust command respectively. Finally $\mathbf{d}$ represents the external disturbance acting on the hexacopter. Note that the heading of the vehicle is not considered in the state vector because an external control loop is used to track the vehicle's heading. The system is linearized around hover-flight, and at every time step, the following optimization problem is solved:

$$\begin{aligned} \min_{\mathbf{u}_i} \sum_{i=0}^{N-1} & \left(\mathbf{x}_i - \mathbf{x}_{ss,i}\right)^T \mathbf{Q} \left(\mathbf{x}_i - \mathbf{x}_{ss,i}\right) + \left(\mathbf{u}_i - \mathbf{u}_{ss,i}\right)^T \\ & \times \mathbf{R}_u \left(\mathbf{u}_i - \mathbf{u}_{ss,i}\right) + \left(\mathbf{u}_i - \mathbf{u}_{i-1}\right)^T \mathbf{R}_\omega \left(\mathbf{u}_i - \mathbf{u}_{i-1}\right) \\ & + \left(\mathbf{x}_N - \mathbf{x}_{ss,N}\right)^T \mathbf{P} \left(\mathbf{x}_N - \mathbf{x}_{ss,N}\right) \\ \text{s.t.} \quad & \mathbf{x}_{i+1} = \mathbf{A}\mathbf{x}_i + \mathbf{B}\mathbf{u}_i + \mathbf{B}_d\mathbf{d}_i \\ & \mathbf{d}_{i+1} = \mathbf{d}_i \\ & \mathbf{u}_{min} \leq \mathbf{u}_i \leq \mathbf{u}_{max} \end{aligned} \tag{11}$$

where $\mathbf{A}$ is the dynamic matrix of the linearized system, $\mathbf{B}$ is the input transfer matrix and $\mathbf{B}_d$ is the disturbance transfer matrix, $\mathbf{Q}, \mathbf{R}_u, \mathbf{R}_\omega$ and $\mathbf{P}$ are the state penalty matrix, input penalty matrix, input change rate penalty matrix and terminal cost matrix respectively, $\mathbf{u}_{min}$ and $\mathbf{u}_{max}$ are the minimum and maximum input vector respectively, $\mathbf{x}_{ss,i}$ and $\mathbf{u}_{ss,i}$ are the controller state and input respectively in steady state at the time step i. Note that the terminal state penalty matrix is computed such that $\mathbf{P}$ is the solution of the discrete time Riccati equation to guarantee recursive feasibility of the receding horizon problem. Finally, it is highlighted that the aforementioned optimization problem (11) is solved using the FORCES toolbox interior point solver (Domahidi 2012) at a rate of 100Hz, by feeding the measured system state $\mathbf{x}_0$, the estimated disturbances $\mathbf{d}_0 = \hat{\mathbf{d}}$ and the future controller state and input $\mathbf{x}_{ss}, \mathbf{u}_{ss}$ steady state vectors.

6.1.2 Inspection path planning of trolley structure

The first experimental test case refers to the inspection of the trolley 3D structure depicted in Fig. 9. The considered mesh model consists of 106 facets capturing the structure in sufficient detail. To raise the challenge, the overall robot workspace is considered to be bounded within a box $3 \times 3 \times 2.75$ m, while a minimum sensing range of $d_{\min} = 1$ m is imposed. The initial mesh that was used by the path planner was extracted out of handheld terrestrial images. Table 4 summarizes the experiment parameters. Using the proposed inspection path planner, a path that guarantees complete coverage was derived and has a total length of 151.44 s. Reconstruction results were computed from the fully pose annotated images of the VI-Sensor utilizing the Pix4D software. The overall reconstruction is of high-quality which increases confidence on the practical usability of such distance-optimized paths. The post-processed files can be found online at (Bircher et al. 2015a). The reference path and the recorded flight response along with the reconstruction results are shown in Fig. 9. The arrows indicate the

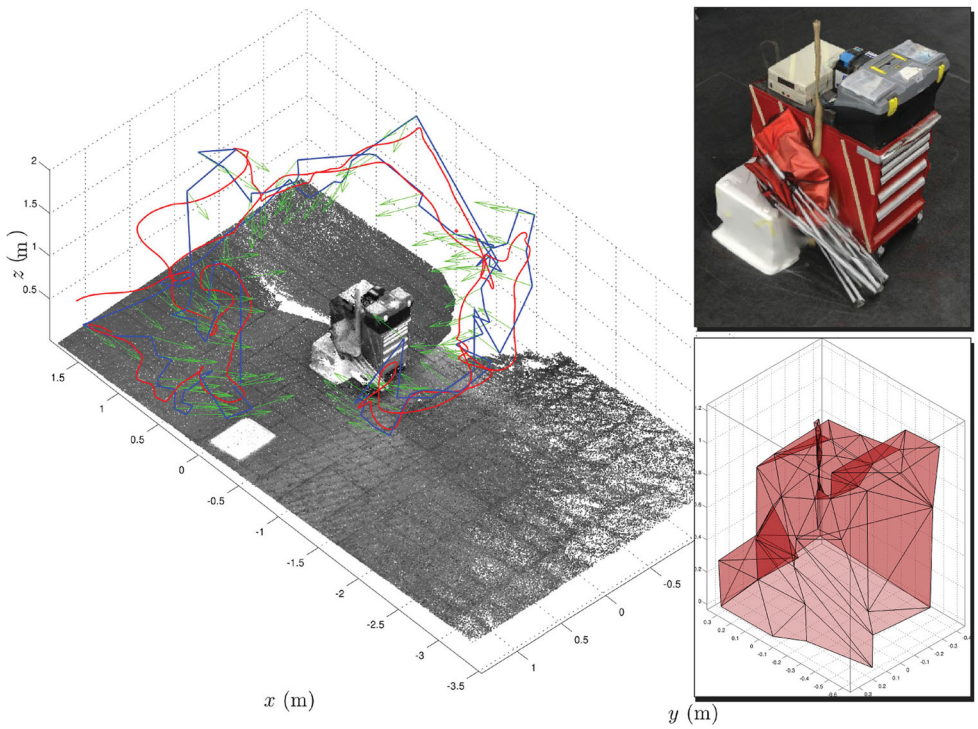


Fig. 9 Experimental study of the inspection of a trolley. The preliminary, terrestrial images-based, 3D reconstruction of the inspection structure is depicted and was used to derive a simplified mesh that was then employed by the inspection path planner to compute the inspection path shown in the Figure. The path cost is 151.44 s for $v_{\max} = 0.25$ m/s and $\dot{\psi}_{\max} = 0.5$ rad/s

Table 4 Rotorcraft UAV inspection scenario trolley

N_{facets}	106		
∠incidence	30°	Bounding box	$3 \times 3 \times 2.75$ m
FoV	[60, 90]°	Mouting pitch	15°
$d_{\min}$	1 m	$d_{\max}$	3 m
$v_{\max}$	0.25 m/s	$\dot{\psi}_{\max}$	0.5 rad/s

reference viewpoints proposed by the inspection planner. A video of the recorded result is available at https://youtu.be/qbwoKFJUm4k.

6.1.3 Inspection path planning of the ETH polyterrasse

The second experimental test-case was that of inspecting a subset of the ETH Polyterrasse, specifically its extruding truncated conic structures. This corresponds to a larger-scale scenario which also poses significantly more challenges regarding the autonomous operation of the aerial robot and its capability to self-localize and map its environment. As far as the inspection path planning problem is concerned, it was conducted based on a rough CAD model of four of these truncated conic structures and an obstacle structure that was around. This option to derive the mesh of the structure to be used for path planning was selected due to the fact that the scene in this case is constructed out of the combination of rather simple geometric components. To ensure safety, a minimum distance of 0.5 m was imposed to the inspection structure (Table 5) and 1 m to the obstacle. The reference path, derived in 6.1 s of computation, along with the experimentally recorded flight path of the vehicle on top of the extracted point cloud are depicted in Fig. 10. As shown—and can be further validated by loading the files provided as publicly released point clouds and triangular mesh in Bircher et al. (2015a)—high quality, very dense reconstruction was achieved. A video of the recorded result is available at https://youtu.be/5kI5ppGTcIQ.

6.2 Fixed-wing UAV inspection operations

A second set of experiments was conducted using a long endurance fixed-wing UAV platform developed by our lab. The particular platform, AtlantikSolar (Oettershagen et al. 2015), is a 5.6 m wingspan, 7.5 kg, solar-powered vehicle. The overall platform is overviewed in Sect. 6.2.1 and the relevant inspection path planning results are presented in Sect. 6.2.3.

6.2.1 Fixed-wing UAV platform overview

The AtlantikSolar UAV (Fig. 11; Table 6) is a hand-launchable low-altitude long-endurance (LALE) solar-powered UAV optimized for long-endurance flight. A detailed overview over the conceptual design of AtlantikSolar is given in Oettershagen et al. (2015). The design methodology is based on the works in Noth (2008); Leutenegger (2014); Oettershagen et al. (2015). The platform owes

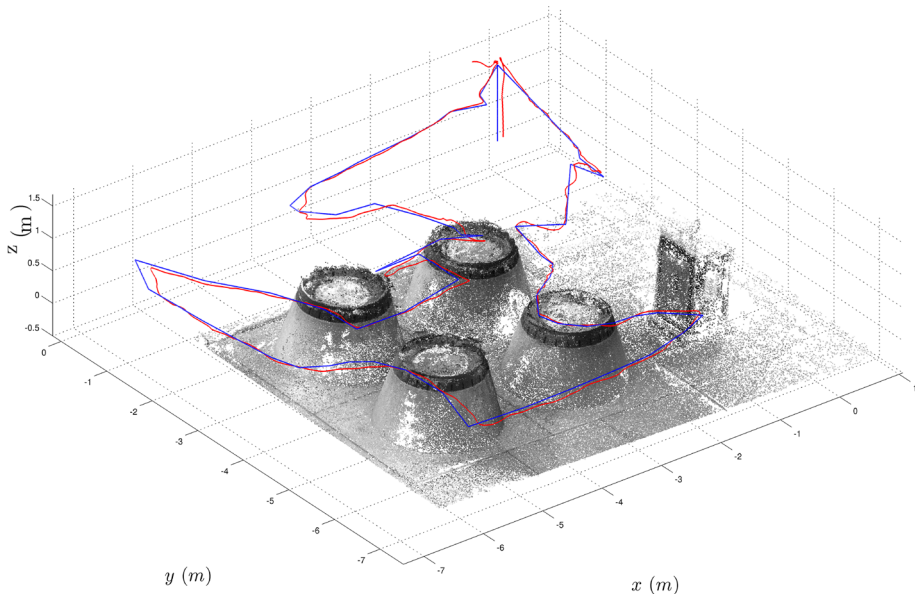


Fig. 10 Experimental study of the inspection of a subset of the ETH Polyterrasse truncated cones. The inspection path was computed based on a rough CAD model and the polyhedric obstacle was also included. The path cost is 167.3 s for $v_{\max} = 0.25$ m/s and $\dot{\psi}_{\max} = 0.5$ rad/s

Table 5 Rotorcraft UAV inspection scenario polyterasse

N_{facets}	178	∠incidence	30°
FoV	[60, 90]°	Mouting pitch	20°
$d_{\min}$	0.5 m	$d_{\max}$	4 m
$v_{\max}$	0.25 m/s	$\dot{\psi}_{\max}$	0.5 rad/s

much of its configuration to the optimization of power consumption. Lightweight composite materials are used in its fabrication. The AtlantikSolar prototype UAV features 88 SunPower *E*60 cells 2.9 kg of cylindrical high energy density Li-Ion batteries (Panasonic NCR18650b, 243 W h kg^{-1}, 700 W h total). The two ailerons, the elevator and the rudder are driven by brushless actuators. The propulsion system consists of a carbon-fiber propeller, a 5:1 reduction gearbox and a 450 W brushless DC motor.

A Pixhawk PX4 (Autopilot 2015) that was subject to major hardware modifications (e.g. integration of the ADIS16448 IMU and the Sensirion SDP600 differential pressure sensor as well as re-writing of the estimation and control algorithms have been performed) is the centerpiece of the avionics system. To provide reliable and drift-free long-term autonomous operation, a light-weight EKF-based state estimator, as presented in Leutenegger et al. (2014), is implemented on the autopilot. Robustness against temporal GPS losses is enhanced through the inclusion of airspeed measurements from a differential barometer. To increase flight safety, the

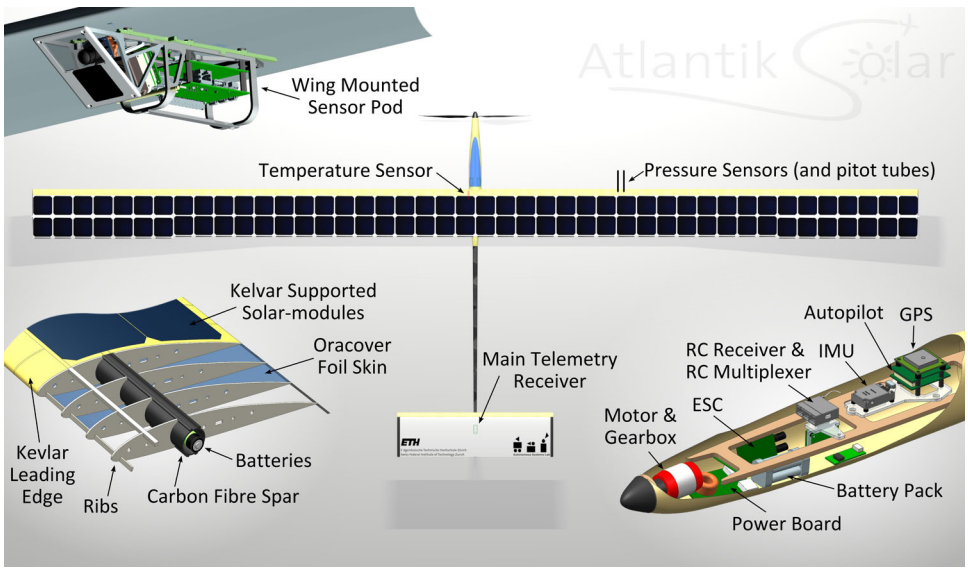


Fig. 11 AtlantikSolar system overview

Table 6 Summary of AtlantikSolar design and performance characteristics

Specification	Value/unit
Wing span	5.65 m
Mass	7.5 kg
Nominal cruise speed	9.7 m s^{-1}
Max. flight speed	20 m s^{-1}
Min. endurance (no payload)[a]	13 h
Design endurance (no payload)	10 days

[a] On battery-power only

algorithm estimates the local three-dimensional wind vector and employs an internal aircraft aerodynamics model to estimate the current sideslip angle and Angle of Attack (AoA), which can in turn be used by the flight controller to apply implicit flight regime limits (Oettershagen et al. 2014).

Subject to the availability of the aforementioned accurate state estimates, AtlantikSolar's flight control system features automatic tracking of waypoints. The baseline control is a set of cascaded PID-controllers for inner-loop attitude control (Stevens and Lewis 1992). Output limiters are applied to respect the aircraft flight envelope, dynamic pressure scaling of the control outputs is used to adapt to the changing moment generation as a function of airspeed and a coordinated-turn controller allows precise turning. Altitude control is based on a Total Energy Control System. Waypoint-following is performed using an extended version of the $\mathcal{L}_1$-nonlinear guidance logic (Park et al. 2004). The detailed implementation and verification of our autopilot is described in Oettershagen et al. (2015).

The AtlantikSolar UAV integrates a sensor pod (see Fig. 12) which features a grayscale (Aptina MT9V034) camera with a long-wavelength infrared (LWIR) camera (FLIR Tau 2) for thermal imaging, both mounted with an oblique FoV. An IMU (Analaog Devices ADIS16448) is also included. All sensors are integrated with a Skybotix VI-Sensor (Skybotix), allowing tight hardware synchronization and timestamping of the acquired data (Nikolic et al. 2014). Furthermore, an Intel Atom-based embedded computer (Kontron COMe-mBT10) is interfaced with the VI-Sensor and the PX4 autopilot. The on-board Atom computer further communicates with the PX4 in order to receive its data, and transmit waypoints. The acquired data is processed on-board and communication with the ground control station is achieved over Wi-Fi. In combination with this sensor pod, an of-the-shelf Sony HDR-AS100VW camera is also utilized. Within the framework of the experiments conducted in this study, the grayscale camera of the sensor pod (with pose annotations coming from the overall state estimation pipeline of the UAV) and the Sony camera (with position annotations coming from its integrated GPS) are employed. The data of these visible light cameras are combined with the pose estimates and fed to post-processing software (Pix4D) to derive accurate 3D reconstructions of the environment.

6.2.2 *Inspection path planning for the Marche-en-Famenne field-trials*

AtlanikSolar is an integral component of the large-scale ICARUS project on assisted search-and-rescue. Within that framework, this UAV participated during autumn 2014 in the field-trials event at Marche-en-Famenne, Belgium (ICARUS) with the role of large-scale mapping, long-term monitoring and communications relay. Geographical Information System (GIS) data were used to derive a first, rough, 8 facets ($N_{facets} = 8$) mesh of the area and subsequently two inspection paths were computed, one for the oblique view grayscale camera of the VI-Sensor with a fixed reference altitude set at an absolute value $z^r = 362$ m (corresponding to 120 m above the highest point to be inspected) and the other for the nadir-mounted Sony HDR-AS100VW with $z^r = 342$ m, while the

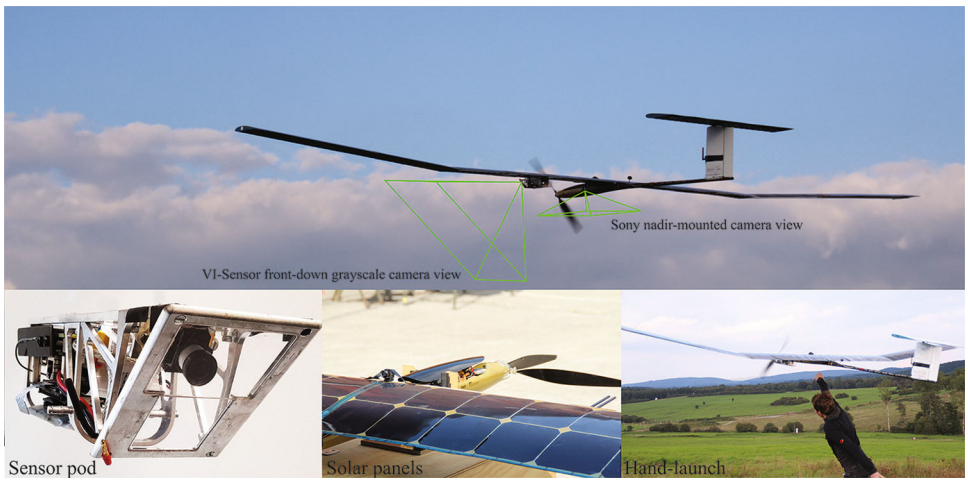


Fig. 12 The AtlantikSolar UAV with the sensor pod attached to its wings and further photos of the sensor pod, the solar cells and an instant of the hand-launching

Table 7 Marche-en-Famenne Inspection Scenario

N_{facets}	8, 8	∠incidence	30°
v_{FW}	9 m/s	$r_{\min}$	60, 60 m
FoV	[90, 50]°, [120, 120]°	Mouting pitch	50°, 90°
Range	Unconstrained	Height	120, 100 m

modeled minimum turning radius was $r_{\min} = 60$ m. Table 7 summarizes the parameters used for the two experiments.

Using the Pix4D software in combination with full-pose annotated images from the two camera systems, highly dense reconstructions were computed as shown in Figs. 13 and 14 for the oblique and the nadir view camera correspondingly. In both cases, the optimized reference inspection path, the real recorded UAV trajectory as well as an offline computed dense point cloud of the inspection area are shown such that the completeness of coverage is visually assessed. A video of the recorded result is available at https://youtu.be/qbwoKFJUm4k.

For both configurations, the proposed inspection planner manages to provide short distance paths that guarantee complete coverage while accounting for the motion constraints of the fixed-wing UAV.

6.2.3 *Inspection path planning for the Rothenthurm Moor landscape*

The final inspection path planning problem considered was that of the 3D mapping of the moor landscape area of Rothenthurm in the Schwyz district of Switzerland. Once again, GIS data was employed to extract a first rough mesh of the environment (consisting of $N_{facets}^{nadir} = 19$ facets and $N_{facets}^{oblique} = 31$ facets) and subsequently, inspection paths were computed for both the nadir- and oblique-camera configurations of the AtlantikSolar UAV. While the altitude

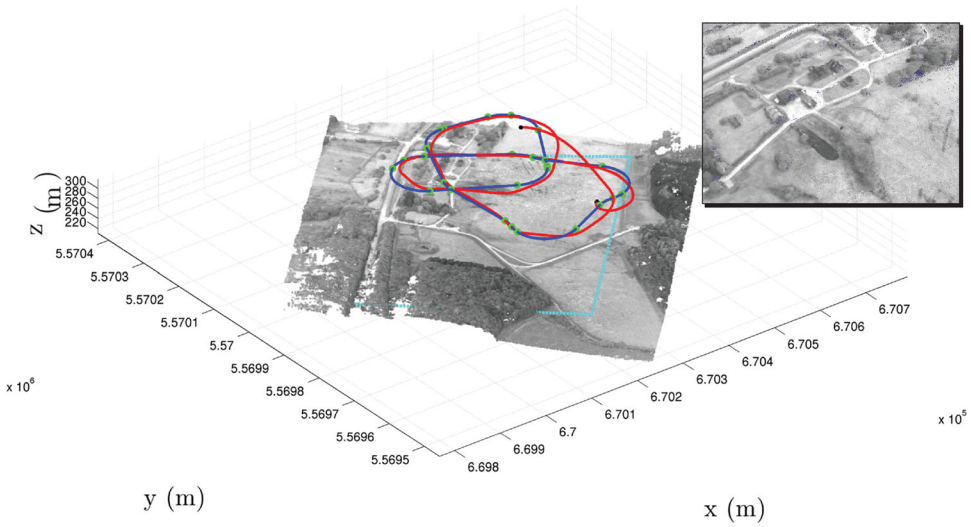


Fig. 13 Inspection path and point cloud for 3D reconstruction purposes using the front-down mounted view grayscale camera of the VI-Sensor on-board AtlantikSolar. *Blue line* represents the reference path, *green circles* are used to indicate the actual waypoints loaded to the autopilot and *red* is used for the vehicle response. The planner commands the vehicle to navigate such that the camera covers the whole desired area marked with *dashed cyan line*. A UTM31N coordinate system is employed (Color figure online)

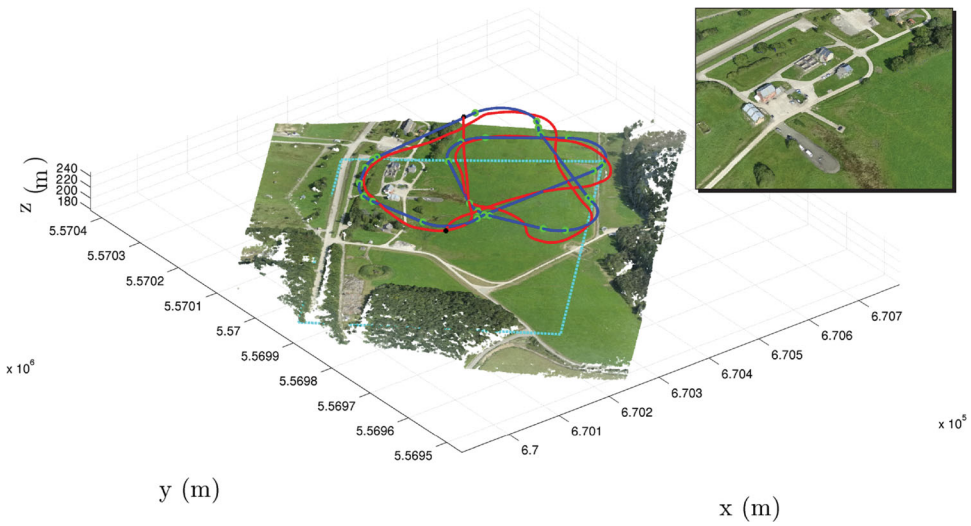


Fig. 14 Inspection path and 3D reconstruction results using the nadir mounted Sony HDR-AS100VW on-board AtlantikSolar. *Blue line* represents the reference path, *green circles* are used to indicate the actual waypoints loaded to the autopilot and *red* is used for the vehicle response. The planner commands the vehicle to navigate such that the camera covers the whole desired area marked with *dashed cyan line*. A UTM31N coordinate system is employed (Color figure online)

Table 8 Rothenthurm Moor landscape inspection scenario

N_{facets}	31, 19	∠incidence	30°, 45°
v_{FW}	9 m/s	$r_{\min}$	60, 60 m
FoV	[90, 50]°, [120, 120]°	Mouting pitch	50°, 90°
Range	Unconstrained	Height	[150, 170], [110, 130] m

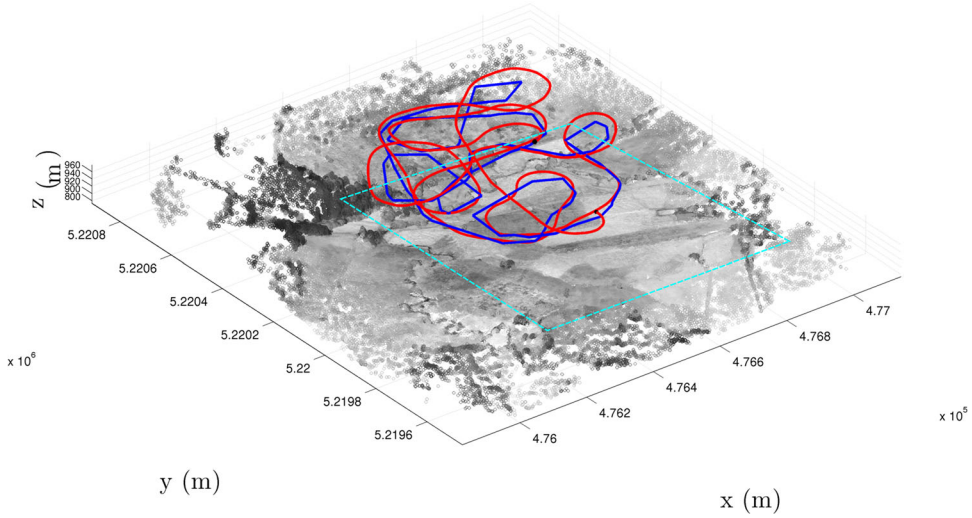


Fig. 15 Inspection path and point cloud for 3D reconstruction purposes using the front-down mounted view grayscale camera of the VI-Sensor on-board AtlantikSolar. *Blue line* represents the reference path that was computed within 145 s and fed as a reference to the autopilot via a very sparse sampling and *red* is used for the vehicle response. The planner commands the vehicle to navigate such that the camera covers the whole desired area marked with *dashed cyan line*. A UTM31N coordinate system is employed (Color figure online)

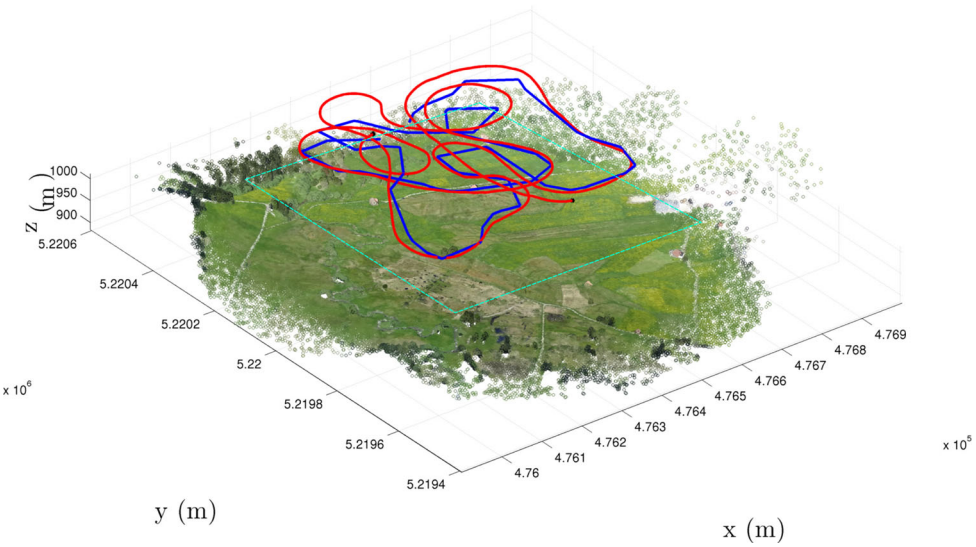


Fig. 16 Inspection path and point cloud for 3D reconstruction purposes using the nadir mounted Sony HDR-AS100VW on-board AtlantikSolar. *Blue line* represents the reference path that was computed within 51 s and fed as a reference to the autopilot via a very sparse sampling and *red* is used for the vehicle response. The planner commands the vehicle to navigate such that the camera covers the whole desired area marked with *dashed cyan line.* A UTM31N coordinate system is employed (Color figure online)

change of the landscape is 15m from the lowest to the highest point, the relative altitude of the UAV with respect to the highest point is constrained by $z^r = [150, 170]$ m for the case of the nadir-view camera. Similarly, relative altitude bounds of $z^r = [110, 130]$ m were set for the case of the oblique-view camera. Table 8 summarizes the mission parameters. Once again, using the Pix4D software with the full-pose annotated images from the two camera systems, very dense reconstructions in the form of point clouds, triangular meshes and digital surface maps were derived and can be found in the public dataset in Bircher et al. (2015a).

Figures 15 and 16 present the reference and recorded flight paths on top of the derived point clouds for the case of the oblique- and nadir-view cameras correspondingly. A video with a segment of the flight relevant to this mission is available at https://youtu.be/DDu5rHYCDkg.

7 Code and dataset release

Enhancing this publication, an open-source toolbox and a richful dataset are available. More specifically, the code

release can be found online at Bircher and Alexis (2015) (available and continuously updated) and once compiled, provides a ROS interface to run the proposed structural inspection path planning algorithm. The released toolbox considers the world as modelled via a triangular mesh, supports both holonomic as well as nonholonomic aerial robots and contains a short set of tuning parameters. Installation instructions, explanation of the parameters as well as the interface, instructions of usage and visualization alongside with demo scenarios, example experimental results and guidelines for further development are provided within the *wiki* page of the online repository.

Furthermore, a continuously updated dataset can be found online (Bircher et al. 2015a). This contains the results presented in the previous Sections while it gets continuously updated over time. All datasets contain dense point clouds derived using the proposed autonomous inspection planner for the flight path, a camera system and the Pix4D software (Pix4D) for post-processing of the pose-annotated images. The point clouds and the 3D-triangular meshes are saved in *.PLY* format and one may load them with any relevant software. Furthermore, for the case of the AtlantikSolar UAV test-flights equipped with the Sony HDR-AS100VW camera orthophoto data along with Google Earth KML files are provided.

8 Conclusions

A new and efficient 3D structural inspection path planning algorithm was proposed within this work. The algorithm is capable of computing optimized paths for both holonomic and nonholonomic vehicles and supports multiple cameras while respecting further mission constraints. The proposed strategy was thoroughly evaluated regarding its capacity to handle complex and large-scale 3D structures. Employing two different aerial robots, a rotorcraft as well as a fixed-wing vehicle, the algorithm was practically tested in four different scenarios relevant with both holonomic and nonholonomic configurations, nadir- and oblique-camera views for close-proximity as well as large-scale 3D mapping applications. With the support of state-of-the-art 3D reconstruction software, the recorded inspection data were postprocessed and as shown practical full coverage, very dense and high-quality point clouds and triangular meshes were derived. The implementation of the proposed algorithm is publicly released and can be found at the— continuously updated—repository in Bircher and Alexis (2015) while the postprocessed data are available at Bircher et al. (2015a). Future work may include extension to moveable cameras with multiple degrees of freedom as well as planning for multi-agent inspection tasks. Furthermore, we believe that it is promising to investigate methods that potentially allow the arrival to better solutions (e.g. using simulated annealing concepts for the viewpoint resampling) as well as to combine such a global planner with local, on-line reactive exploration planners that can deal with model uncertainty or serious deviations from the nominal trajectory.

Acknowledgments This work has received funding from the European Union's Horizon 2020 Research and Innovation Programme under the Grant Agreement No.644128, AEROWORKS.

References

Acar, E. U., Choset, H., & Lee, J. Y. (2006). Sensor-based coverage with extended range detectors. *IEEE Transactions on Robotics*, *22*(1), 189–198.

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.

Atkar, P., Conner, D. C., Greenfield, A., Choset, H., & Rizzi, A. A. (2004). Uniform coverage of simple surfaces embedded in R 3 for auto-body painting. In *Sixth workshop on the algorithmic foundations of robotics, Utrecht/Zeist, The Netherlands*.

Autopilot, P. (2015). http://pixhawk.org/.

Bircher, A., & Alexis, K. (2015) Structural inspection planner code release (Online). https://github.com/ethz-asl/StructuralInspectionPlanner.

Bircher, A., Alexis, K., Burri, M., Kamel, M., Oettershagen, P., Omari, S., Mantel, T., & Siegwart, R. (2015a) Structural inspection planner dataset release (Online). https://github.com/ethz-asl/StructuralInspectionPlanner/wiki/Example-Results.

Bircher, A., Alexis, K., Burri, M., Oettershagen, P., Omari, S., Mantel, T., & Siegwart, R. (2015b). Structural inspection path planning via iterative viewpoint resampling with application to aerial robotics. In *IEEE international conference on robotics and automation (ICRA)* (pp. 6423–6430).

Boon, K., & Lovelace, D. C. (2014). *The domestic use of unmanned aerial vehicles*. Oxford: Oxford University Press.

Burri, M., Nikolic, J., Hurzeler, C., Caprari, G., & Siegwart, R. (2012). Aerial service robots for visual inspection of thermal power plant boiler systems. In *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.

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

Dantzig, G., Fulkerson, R., & Johnson, S. (1954). Solution of a large-scale traveling-salesman problem. *Journal of the Operations Research Society of America*, *2*(4), 393–410.

Domahidi, A. (October 2012). FORCES: Fast optimization for real-time control on embedded systems. http://forces.ethz.ch.

Ferreau, H., Kirches, C., Potschka, A., Bock, H., & Diehl, M. (2014). qpOASES: A parametric active-set algorithm for quadratic programming. *Mathematical Programming Computation*, *6*(4), 327–363.

Gabriely, Y., & Rimon, E. (2002). Spiral-stc: An on-line coverage algorithm of grid environments by a mobile robot. In *Proceedings of IEEE international conference on robotics and automation, ICRA'02.* (vol. 1, pp. 954–960). IEEE

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

González-Baños, H. (2001). A randomized art-gallery algorithm for sensor placement. In *Proceedings of the seventeenth annual symposium on Computational geometry* (pp. 232–240). ACM.
Grabcad. https://grabcad.com/library/generic-direct-drive-wind-turbine-with-77-dot-5-meter-conventional-tower.
Helsgaun, K. (2000). An effective implementation of the lin-kernighan traveling salesman heuristic. *European Journal of Operational Research*, *126*(1), 106–130.
Hert, S., Tiwari, S., & Lumelsky, V. (1996). A terrain-covering algorithm for an auv. In *Underwater Robots* (pp. 17–45) Springer.
Hover, F. S., Eustice, R. M., Kim, A., Englot, B., Johannsson, H., Kaess, M., et al. (2012). Advanced perception, navigation and planning for autonomous in-water ship hull inspection. *The International Journal of Robotics Research*, *31*(12), 1445–1464.
ICARUS: Unmanned search and rescue. http://www.fp7-icarus.eu/.
Karaman, S., & Frazzoli, E. (2011). Sampling-based algorithms for optimal motion planning. *The International Journal of Robotics Research*, *30*(7), 846–894.
Kroll, A., Baetz, W., & Peretzki, D. (May 2009). On autonomous detection of pressured air and gas leaks using passive ir-thermography for mobile robot application. In *International conference on robotics and automation, 2009. ICRA '09. IEEE* (pp. 921–926).
Leutenegger, S. (2014). Unmanned solar airplanes: Design and algorithms for efficient and robust autonomous operation. *Ph.D. Dissertation*, ETH Zurich.
Leutenegger, S., Melzer, A., Alexis, K., & Siegwart, R. (2014). Robust state estimation for small unmanned airplanes. In *IEEE Multiconference on systems and control (MSC)*, Antibes, France.
Leutenegger, S., Melzer, A., Alexis, K., & Siegwart, R. (2014). Robust state estimation for small unmanned airplanes. In *IEEE multiconference on systems and control*.
Lin, S., & Kernighan, B. W. (1973). An effective heuristic algorithm for the traveling-salesman problem. *Operations research*, *21*(2), 498–516.
Luo, C., Yang, S. X., Stacey, D. A., & Jofriet, J. C. (2002). A solution to vicinity problem of obstacles in complete coverage path planning. In *Proceedings of IEEE international conference on robotics and automation, ICRA'02* (vol. 1, pp. 612–617). IEEE.
Metni, N., & Hamel, T. (2007). A UAV for bridge inspection: Visual servoing control law with orientation limits. *Automation in Construction*, *17*(1), 3–10.
Nikolic, J., Rehder, J., Burri, M., Gohl, P., Leutenegger, S., Furgale, P. T., & Siegwart, R. Y. (2014). A Synchronized Visual-inertial sensor system with FPGA pre-processing for accurate real-time SLAM. In *IEEE International conference on robotics and automation (ICRA)*.
Noth, A. (2008). Design of solar powered airplanes for continuous flight. *Ph.D. dissertation*, ETH Zurich.
Oettershagen, P., Melzer, A., Leutenegger, S., Alexis, K., & Siegwart, R. (2014). Explicit model predictive control and $\mathcal{L}_1$-navigation strategies for fixed-wing UAV path tracking. In *22nd Mediterranean conference on control & automation (MED)*.
Oettershagen, P., Melzer, A., Mantel, T., Rudin, K., Lotz, R., Siebenmann, D., Leutenegger, S., Alexis, K., & Siegwart, R. (May 2015). A solar-powered hand-launchable uav for low-altitude multi-day continuous flight. In *IEEE International conference on robotics and automation (ICRA)* (accepted).
O'rourke, J. (1987). *Art gallery theorems and algorithms* (Vol. 57). Oxford: Oxford University Press.
Papadopoulos, G., Kurniawati, H., & Patrikalakis, N. M. (2013). Asymptotically optimal inspection planning using systems with differential constraints. In *IEEE International conference on robotics and automation (ICRA)* (pp. 4126–4133). IEEE.
Park, S., Deyst, J., & How, J. P. (2004). A new nonlinear guidance logic for trajectory tracking. In *AIAA guidance, navigation, and control conference and exhibit* (pp. 16–19).
Pix4D. http://pix4d.com/.
RotorS: An MAV gazebo simulator. https://github.com/ethz-asl/rotors_simulator.'
Skybotix, A. G. http://www.skybotix.com/.
Stevens, Brian L., & Lewis, Frank L. (1992). *Aircraft control and simulation*. Hoboken: Wiley.
Swisstopo swissAlti3D height map. http://www.swisstopo.admin.ch/internet/swisstopo/en/home/products/height/swissALTI3D.html.
Zelinsky, A., Jarvis, R. A., Byrne, J., & Yuta, S. (1993). Planning paths of complete coverage of an unstructured environment by a mobile robot. *Proceedings of International Conference on Advanced Robotics*, *13*, 533–538.

Andreas Bircher is 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 April 2014. His research interests lie in the area of path planning for autonomous robots, specifically, coverage and inspection 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 PhD 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 obtained his PhD in the field of aerial robotics control and collaboration from the University of Patras, Greece in 2011. Noteworthy, his PhD research was supported by the Herakleitos II Excellence Fellowship. Being awarded a Swiss Government Fellowship, after successfully defending his PhD Thesis he then moved to Switzerland and ETH Zurich. Since 2012 he holds the position of Senior Researcher at the Autonomous Systems Lab, ETH Zurich. His research interests lie in the fields of control, navigation, optimization and path-planning focusing on aerial robotic systems. He is the author or co-author of more than 40 scientific publications and has received several best paper awards and distinctions including the IET Control Theory & Applications Premium Award 2014. Kostas Alexis has participated and organized several large-scale multi-million research projects with broad international involvement and collaboration.

Michael Burri obtained his MSc from ETH Zurich with a specialization in Robotics, Systems and Control and since then he has been working actively in the area of autonomous aerial robotics as a PhD researcher of the Autonomous Systems Lab at ETH Zurich. His current primary focus is related with robust and ego-motion estimation and overall advanced autonomous navigation of aerial robotics.

Philipp Oettershagen received a M.Sc. in Aerospace Engineering from California Institute of Technology in 2010 and a Diploma (Dipl.-Ing.) in Aerospace Engineering from University of Stuttgart, Germany, in 2011. He is currently a research assistant and PhD-student at ETH Zurich's Autonomous Systems Lab, where he is leading the AtlantikSolar Unmanned Aerial Vehicle (UAV) project. Besides the conceptual design and system engineering of high-performance solar-powered UAVs, he is interested in using in-flight estimations of the often highly cluttered and dynamic meteorological environment for trajectory planning and UAV decision making.

Sammy Omari received the master's degree in mechanical engineering from ETH Zurich, Zurich, Switzerland, in 2012, where he is currently working toward the Ph.D. degree with a thesis concerning intelligent control and teleoperation of unmanned aerial vehicles in the Autonomous System Laboratory. During his master's degree work he was involved in the development of the flybox platform. Mr. Omari received the deVigier Young Entrepreneur Award for his involvement with the company Skybotix.

Thomas Mantel is 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 2013. His research interests lie in the area of solar-powered unmanned aerial vehicles, sensor systems and long-range communications.

Roland Siegwart is full professor for autonomous systems at ETH Zurich since July 2006. He has a Diploma in Mechanical Engineering (1983) and PhD 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.

2015 IEEE International Symposium on Intelligent Control (ISIC)
Part of 2015 IEEE Multi-Conference on Systems and Control
September 21-23, 2015. Sydney, Australia

Uniform Coverage Structural Inspection Path–Planning for Micro Aerial Vehicles

Kostas Alexis[1], Christos Papachristos[2], Roland Siegwart[2] and Anthony Tzes[2]

***Abstract*— This work proposes a new path–planning framework that provides uniform coverage of 3D structures by employing an iterative strategy to improve the inspection path that benefits from remeshing–techniques, while the first full solution is computed very fast. The resulting paths inspect each detail of the structure from a distance that directly depends to the local geometrical complexity of the structure and viewpoints are selected accordingly. For each admissible set of viewpoints, a Traveling Salesman Problem is solved and leads to the inspection route that the algorithm outputs at each iteration. The proposed path–planning algorithm is initially evaluated in extensive simulation studies. Finally, an experimental case study using a Micro Aerial Vehicle and a realistic mockup model of a power transformer is conducted, validating the advantages of the proposed scheme in conducting fine-quality inspection.**

I. Introduction

Aerial robotics are constantly proving their potential and their capabilities to assist in a large variety of civilian applications of great importance. Among others, the emerging field of autonomous aerial structural inspection is gaining significant interest as aerial robots can provide new, unique, high–fidelity, versatile and cost–effective alternatives to the so–far achieved state of the art that relies on manned systems. Examples already include efforts for automated infrastructure inspection using Micro Aerial Vehicles (MAVs) [1] such as bridge monitoring or risk detection in boiler power plants using cameras [2] or nondestructive testing [3, 4].

However, despite the interest that this field has attracted, important challenges still have to be addressed. From an applications perspective, there is a clear need for fast computation of inspection paths that provide full coverage of the desired structure, pay sufficient focus on its details and are in general of short length. As inspection operations requirements are complex, end–users naturally demand to be able to control the level of inspection fidelity according to the initial knowledge, the geometry of the structure or the time available for the execution of a mission.

Within the literature, two main families of 3D inspection path planning algorithms may be identified [5, 6], namely those that: a) separate the problem into that of finding a *(possibly minimal)* set of viewpoints and then compute an optimal route among them [7] and b) those that aim to optimize the viewpoint configurations and the overall path in a unified step [8]. In general, methods that follow the separated approach are characterized by lower computational complexity while the second category emphasizes on optimality at the cost of increased computational times. Previous work of a subset of the authors aimed to make a step towards iteratively optimized solutions that retain a low computational load [1].

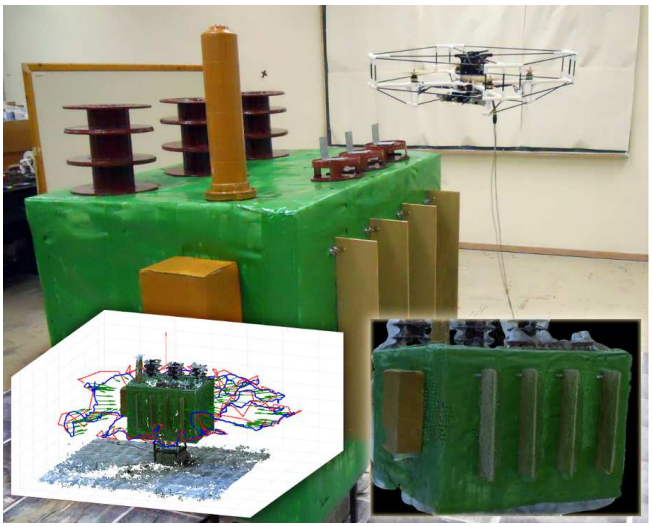


Fig. 1: Indicative experimental application of the proposed algorithm employing a micro aerial vehicle, for the case of a power transformer mockup.

The focus of this work is put on a very different direction, and its goal is not limited to that of finding full coverage paths with short distance. Motivated from application needs, the proposed Uniform Coverage 3D structure Inspection Path–Planner (UC3D–IPP) rather tries to: a) compute viewpoints that provide full coverage but also *uniform* focus into each detail of the structure by adjusting the perception distance and orientation, b) compute the first *almost uniform* coverage paths very fast and with scalable processing requirements, c) employ an iterative framework that makes use of a set of geometrical rules to improve coverage uniformity while keeping the amount of viewpoints small and the path short, while finally d) allows to define the minimum acceptable inspection detail and levels of uniformity. To achieve these goals, the framework benefits from the properties of mesh representations of the 3D structure on which it uses remeshing techniques, while the distance of each viewpoint from the corresponding mesh face is a function of the local complexity of the structure. Once an admissible set

This work has received funding from the European Union's Horizon 2020 Research and Innovation Programme under the Grant Agreement No. 644128, AEROWORKS.

[1]The authors are with the Autonomous Systems Lab at ETH Zurich, Leonhardstrasse 21, 8092, Zurich, Switzerland. email: konstantinos.alexis@mavt.ethz.ch

[2]The authors are with the University of Patras, Eratosthenous 6 Str. Patras, Greece.

of viewpoints is chosen, the optimal route among them is computed by solving a Traveling Salesman Problem employing the efficient Lin–Kerninghan–Helsgaun heuristic. The algorithm initializes with a downsampled version of the mesh to find a first solution and subsequently higher fidelity meshes are loaded, more viewpoints are added and new inspection routes are computed. The overall framework was tested in multiple simulation and experimental test–cases and Figure 1 illustrates the algorithm's experimental application employing a micro aerial robotic system. Jointly with this paper, a dataset is released and is available in [9] and contains additional evaluation studies while it will be further updated and maintained in the future.

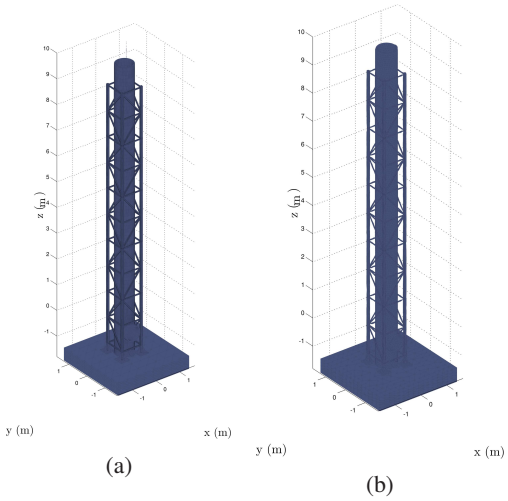
The remainder of the paper is organized as follows. In Section II the methods to represent the 3D structure are discussed followed by the description of the basic version of UC3D–IPP in Section III. The iterative UC3D–IPP is presented in Section IV and some heuristical adaptations are introduced in Section V. Simulation evaluation studies are presented in Section VI followed by experimental studies in Section VII. Finally, conclusions are drawn in Section VIII.

II. Representation of 3D Structures for Inspection Path–Planning

The selection of the representation method of the 3D structure has a key role in the implementation of any inspection path–planner. Within this work, the representation of the structure is fundamental for some of the algorithm main capabilities. More specifically, *uniform* triangular meshes are employed to represent the 3D structure model for which a full coverage inspection path is planned. The initial mesh representation of the structure to be inspected is processed using the generic remeshing algorithm proposed in [10] while iterative reduction steps follow. Figures 2 and 3 provide example results of this remeshing strategy applied on models relevant to structural inspection path–planning. Specifically, the effect of heavy resampling steps applied in the case of the model of an industrial chimney and a model of the Hoa Hakananai'a statue are considered.

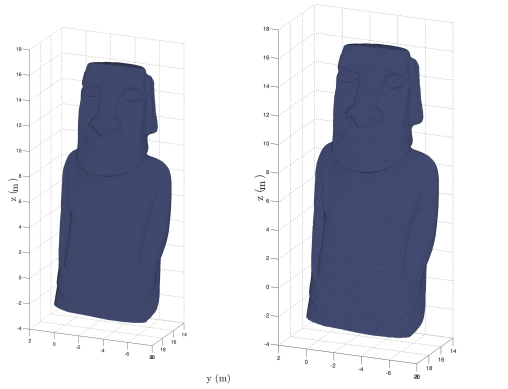
III. Uniform Coverage Path-Planning

The specific focus of the Uniform Coverage 3D structure Inspection Path–Planner (UC3D–IPP) is to provide a framework which computes inspection paths that a) provide full coverage of the desired structure and b) the inspection distances are a function of the structural fidelity and complexity per subset of the inspection manifold so that uniformity in 3D reconstruction tasks is more probable. The proposed framework provides first inspection paths extremely fast (and with controllable level of computational load) while it employs an *iterative* framework to improve the solution or consider parts of the structural information lost during the mesh downsampling process. The minimum desired focus on the structural details can be initially set by providing the most downsampled mesh model to be considered. The envisaged use of the algorithm is both in the sense of long–term offline planning of high–fidelity inspections as well as close to real–time inspection path–planning in the field. For the latter case, the capability of the algorithm to provide its first solution very fast is fundamental.



(a) (b)

Fig. 2: Initial model of an industrial chimney (141388 vertices and 275964 faces) and a subsequent step of 25% reduction of the number of vertices using the employed model resampling technique.



(a) (b)

Fig. 3: Initial model of the Hoa Hakananai'a statue (63223 vertices and 126446 faces) and a subsequent step of 25% reduction of the number of vertices using the employed model resampling technique.

A. Basic UC3D–IPP

This section deals with the explanation of the *basic* UC3D–IPP. With the term *basic* we refer to the fundamental algorithmical block that computes uniform coverage paths against a given mesh of a structure which might not be the actual structure model but rather a downsampled version of its mesh representation. Derivation of such simplified versions can be acquired by utilizing the mesh resampling techniques described in Section II.

Let $\mathcal{G}_i, \mathcal{F}_i$ be the set of all vertices υ_j and faces $\mathbf{f}_j$ of a specific mesh model and $\Gamma = [\phi^C, \theta^C, \psi^C]$ and $\Pi = [\Delta x^C, \Delta y^C, \Delta z^C]$ the camera mounting configuration in terms of rotation and position deviation from the vehicle center of mass. Then the basic UC3D–IPP follows the steps

described in Algorithm 1. Within that – and from now on – $\mathcal{V}$ represents the set of computed viewpoints.

Algorithm 1 Basic UC3D–IPP Inspection path planner: The computation of the viewpoints is done according to the description in section III-A.2 and for the computation of the tour an implementation of the Lin-Kernighan Heuristic [11] is employed.

$\mathcal{P} \leftarrow$ *ExtractPolygons*$(\mathcal{G}_i, \mathcal{F}_i)$
for all $\mathbf{p}_k \in \mathcal{P}$ **do**
  $\mathbf{v}_k \leftarrow$ *ComputeViewpoint*$(\mathbf{p}_k, \Gamma, \Pi)$
  $\mathcal{V} \leftarrow \mathcal{V} \cup \mathbf{v}_k$
for all $\mathbf{v}_n \in \mathcal{V}$ **do**
  for all $\mathbf{v}_m \in \mathcal{V}$ **do**
    $\mathbf{C}(n,m) \leftarrow$ *ConnectionDistance*$(\mathbf{v}_n, \mathbf{v}_m)$
$\mathbf{r}_i \leftarrow$ *ComputeViewpointsRoute*$(\mathbf{C}(n,m))$
return $\mathbf{r}_i$

The execution of this algorithm relies on the functions *ExtractPolygons*(), *ComputeViewpoint*(), *ConnectionDistance*() and *ComputeViewpointsRoute*() which are described in the following subsections.

1) Mesh Faces Polygon Extraction: The function *ExtractPolygons*() executes the operation of extracting the polygons that correspond to the triangular faces of the model.

2) Viewpoint Configuration Computation: Viewpoint computation per mesh face is done based on pure geometric calculations. The complete procedure consists of three main and one optional step, namely: a) loading of the baseline camera viewing polyhedron model, b) rotation and translation of this polyhedron based on the mounting configuration of the camera (ϕ^C, θ^C, ψ^C and $\Delta x^C, \Delta y^C, \Delta z^C$) and finally c) translational move and corresponding enlargement/downsizing of this polyhedron so that the mesh facet becomes at least exactly visible (either exactly or based on a minimum allowed viewing distance). In case, the mesh face is big enough, so that the maximum viewing distance is violated during this process, then it is divided into two smaller faces and two viewpoint configurations are computed. This process may be repeated until valid viewpoints are found.

3) Viewpoint–to–Viewpoint Connecting Distance Computation: The computation of the connecting paths and corresponding distances among the different viewpoints that have to be visited is done via the combination of a Boundary Value Solver (BVS) for the considered vehicle dynamics with a collision–free point–to–point path–planner that employs the RRT* algorithm [12] and executes a finite N_{RRT} number of iterations of it. The overall procedure that leads to collision–free connections between each viewpoint with all the others is summarized in Algorithm 2. Within that, *BVS*$_T$() represents the boundary value solver for the vehicle configuration T which may be holonomic or nonholonomic. Similarly, the function RRT$^\star_T$ denotes the execution of an RRT* planner that implements ray–checking for collisions against the mesh represented by the vertices and faces sets $\mathcal{V}, \mathcal{F}$ and may account for possible nonholonomic limitations in terms of minimum turning radius and maximum ascending/descending rates. Regarding the employed BVS it is noted that within this paper only holonomic assumptions are considered (for example for rotorcraft–type MAVs like quadrotors, hexacopters or conventional helicopters) augmented with yaw rate constraints.

Algorithm 2 Computation of the connecting path between the viewpoint configuration $\mathbf{v}_n$ and $\mathbf{v}_m$.

$p_{\mathbf{v}_n \to \mathbf{v}_m} \leftarrow$ *BVS*$_T(\mathbf{v}_n, \mathbf{v}_m)$
if *CheckCollision*$(p_{\mathbf{v}_n \to \mathbf{v}_m}, \mathcal{V}, \mathcal{F})$ == **TRUE** **then**
  $p_{\mathbf{v}_n \to \mathbf{v}_m} \leftarrow \text{RRT}^\star_T(\mathbf{v}_n, \mathbf{v}_m, \mathcal{V}, \mathcal{F}, N_{RRT})$
return $p_{\mathbf{v}_n \to \mathbf{v}_m}$

4) Complete Viewpoints Visiting Route Computation: Once the complete set of viewpoint configurations $\mathcal{V}$ has been found, the computation of the optimal inspection route that visits all of them and returns to the initial position corresponds to the solution of the Traveling Salesman Problem (TSP) with graph vertices the selected viewpoints and edges that correspond to the cost–to–go of each viewpoint–to–viewpoint connection. In order to enable fast computation of such a solution, the function *ComputeViewpointsRoute*() employs an efficient implementation of the Lin–Kernighan–Helsgaun (LKH) heuristic [11, 13]. It is known that local search with k–change neighborhoods, k–opt as it is called, is one of the most widely used heuristic algorithms to solve TSP.

IV. Iterative UC3D–IPP

As mentioned, among the main goals of this work is to provide an inspection path–planning framework that has the capacity to provide uniform solutions for the provided mesh model and is able to iterate and improve the solution as long as this first solution –based on a more rough mesh model– is not considered sufficient and mission time is available. Towards this direction, the algorithm can iteratively load improved –higher fidelity– mesh models and not fully recompute a path (as the *basic* UC3D–IPP) but modify the existing by adding a new set of viewpoints that will lead to an almost equally uniform coverage based on the updated mesh model. This advanced *Iterative* Uniform Coverage 3D structure Inspection Path–Planner implements the additional functionality depicted in Algorithm 3 once a first step of the *basic* UC3D–IPP has executed and has found its initial solution with viewpoint configurations $\mathcal{V}^{basic}$.

Implementation of the iterative UC3D–IPP relies on the newly introduced function *IsCoveredUniformly*(). For each triangular face of the new and higher–fidelity mesh, this function performs three main steps, namely: a) computes the inspection polyhedron of each viewpoint, b) checks if any of the existing viewpoint configurations provides coverage of this face and if so c) checks if the inspection distance is around the value (with a tunable threshold) it should be if the viewpoint configuration was computed initially using *ComputeViewpoint*(). If any of the results of this test is

Algorithm 3 Itarative UC3D–IPP Inspection path planner: New viewpoints are added on top of the basic UC3D–IPP solution as long as faces of the higher fidelity mesh are not covered or covered from distances that exceed a threshold of uniformity. Subsequently, a new inspection route is solved employing the LKH–heuristic. Within this algorithm, $\mathcal{V}^i$ represents the set of computed viewpoints within the i–th iteration of *iterative* the UC3D–IPP and $\mathcal{V}^{basic}$ refers to the last computed set of viewpoints of the *basic* UC3D–IPP.

$\mathcal{V}^{i-1} \leftarrow \mathcal{V}^{basic}$
$\mathcal{V}^{i} \leftarrow \mathcal{V}^{i-1}$
$\mathcal{P}_i \leftarrow$ ExtractPolygons$(\mathcal{G}_i, \mathcal{F}_i)$
for all $\mathbf{p}_{k,i} \in \mathcal{P}_i$ **do**
  if IsCoveredUniformly$(\mathbf{p}_{k,i}, \mathcal{V}^{i-1})$ == **FALSE then**
    $\mathbf{v}_{k,i} \leftarrow$ ComputeViewpoint$(\mathbf{p}_{k,i})$
    $\mathcal{V}^{i} \leftarrow \mathcal{V}^{i} \cup \mathbf{v}_{k,i}$
for all $\mathbf{v}_n \in \mathcal{V}^i$ **do**
  for all $\mathbf{v}_m \in \mathcal{V}^i$ **do**
    $\mathbf{C}(n,m) \leftarrow$ ConnectionDistance$(\mathbf{v}_n, \mathbf{v}_m)$
$\mathbf{r}_i \leftarrow$ ComputeViewpointsRoute$(\mathbf{C}(n,m))$
return $\mathbf{r}_i$

negative, then a new viewpoint is computed and added to the set of viewpoints corresponding to this algorithmical iteration $\mathcal{V}^i$, while otherwise the set of viewpoints is not updated. Once any new viewpoint configurations that ensure that the new mesh model is fully and uniformly covered, a new optimal route that visits all the updated viewpoint configurations is computed using the LKH heuristic.

V. Collision Checking Speed–up

As it will be revealed within Section VI, collision checking is among the most computationally expensive procedures within the UC3D–IPP algorithm. This is due to the fact that every possible connecting edge of the tour has to be evaluated in order to assess if it leads to a collision with any of the mesh faces. Simple heuristic approximations allow faster collision–checking while guaranteeing safety. This is specifically done by replacing the mesh vertices and faces used for collision checking with those derived by computing the convex hull of a slightly magnified mesh which is then heavily simplified. Apart from magnification of the employed mesh, utilization of bounding boxes to approximate complex structures like construction networks can also have significant impact.

VI. Evaluation Studies in Simulation

Thorough evaluation of the proposed inspection path–planning framework was conducted for the case of an aerial robot capable of flying holonomic trajectories. The aforementioned 3D structure mesh models were employed as test–cases and several simulation runs were executed. In particular, the performance of the algorithm being executed on meshes with different levels of fidelity along with the performance of the iterative improvement steps and the

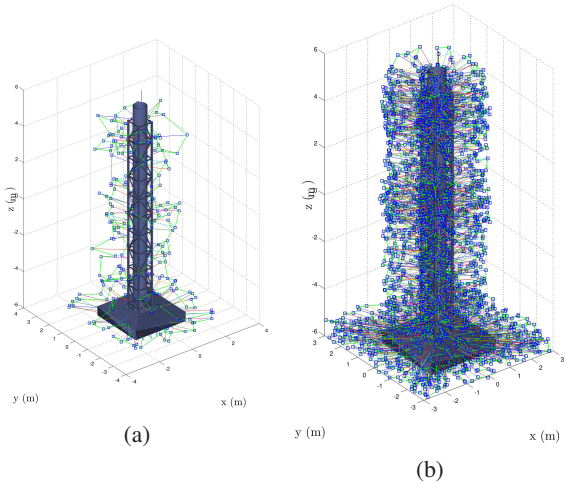


Fig. 4: Two executions of the basic UC3D–IPP algorithm for a more and a less simplified version of the industrial chimney mesh model consisting of (234 vertices, 460 faces) and (1415 vertices, 2730 faces) respectively.

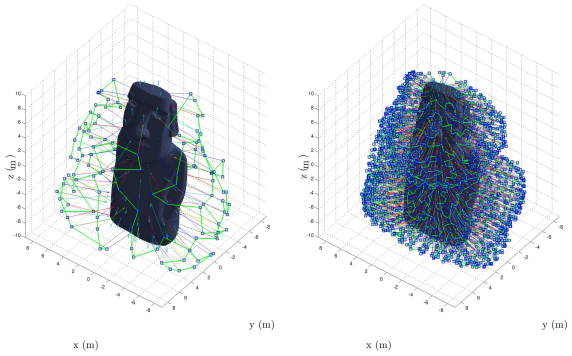
heuristic adaptations was evaluated. The results are presented below and as shown the UC3D–IPP does provide fast solutions that ensure uniform coverage for the given initial mesh models and an almost uniform, iteratively improved, coverage given that more computation time is available. Jointly with this publication, a dataset is released and is available in [9]. It contains additional evaluation studies and will be continuously updated and maintained.

A. Results using the basic UC3D–IPP

Considering the industrial chimney and Hoa Hakananai'a models presented in section II, two executions of the basic UC3D–IPP were ran for different levels of simplification of the initial mesh model. The inspection results are presented in Figures 4, 5. For all these cases, a field of view equal to 40 degrees in both directions was considered along with a mounting configuration of $\Gamma = [0, -12, 0]$deg and $\Pi = [0, 0, 0]$m. The number of $RRT^\star$ iterations is set to $N_{RRT} = 500$ while for these simulations none of the heuristics mentioned in Section V were employed. Observing the computation load analysis shown in Figure 6 indicates that the total time needed is short and for rough representations it only takes a few seconds to come up with a first solution. As further analyzed in Figure 6, the component that consumes most of the time is actually the assembly of the cost matrix as during that, the collision checking against each and every facet of the mesh (or other possible obstacles of the environment) has to take place.

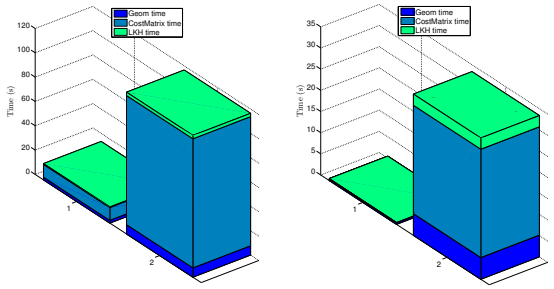
B. Results using the Iterative UC3D–IPP

As described in Section IV, the iterative adaptation of the proposed uniform inspection path–planner can improve the solution by loading higher fidelity mesh models and additively inserting new sets of viewpoints that "refine and repair" the path leading to an almost equally uniform coverage now based on the updated and more detailed mesh



(a) (b)

Fig. 5: Two executions of the basic UC3D algorithm for a more and a less simplified version of the HoA-Hakananai'a mesh model consisting of (61 vertices, 118 faces) and (987 vertices, 1970 faces) respectively.



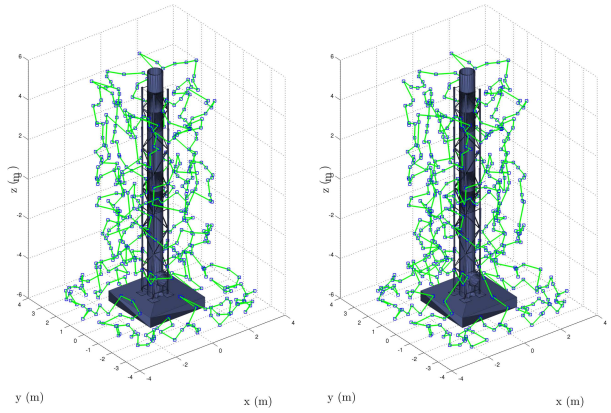
(a) (b)

Fig. 6: Analysis of the computational needs of each of the fundamental basic–UC3D algorithm. As shown the overall execution time is small even for meshes of significant complexity while the most expensive operation is the assembly of the cost–matrix which contains checking for collisions and replanning using RRT* as long as the direct solution of the BVS turns out not to be collision–free.

model. The new added viewpoints at each iteration either "fill the gaps" by providing coverage for structural details lost in previous steps of mesh simplification or improve the uniformity of coverage in case a significant deviation from the expected inspection distance is observed. This iterative version of the algorithm was evaluated in simulation studies. The model of the industrial chimney is employed and Figure 7 presents the initial and the two subsequent improved paths. As highlighted in Figure 8, the iteratively found solutions are computed with a relatively small additional computation overhead and the cost of less uniform solution comes with the benefit of a solution that is computed faster compared with the case that the updated mesh models would be used directly with the basic UC3D–IPP.

VII. Experimental Studies

The proposed UC3D–IPP algorithm was additionally evaluated experimentally for the case of a power–transformer structure. A realistically-sized mockup model of an actual power transformer was used, while the execution of the inspection process was performed by an aerial robot capable of autonomous perception of its environment, estimation of



(a) (b)

Fig. 7: Initial execution of the basic–UC3D on a highly simplified mesh and a subsequent execution of the iterative–UC3D for a mesh model of much higher fidelity. As shown, only a few extra viewpoints were needed and the solution is sufficiently simpler than the execution of the basic–UC3D on the more complex mesh.

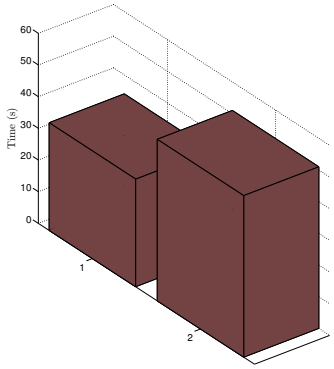


Fig. 8: Execution time for the basic–UC3D on the more rough and simplified mesh (427 vertices, 840 faces) and execution time of the path–improving iterative–UC3D for the more complex and higher–fidelity mesh model (1415 vertices, 2730 faces) with a setting of 20% acceptable deviation of the expected uniform inspection distance. Only a small increase in execution time is observed.

its motion and navigation [14–17], and equipped with an onboard $RGB-D$ image and depth sensor. More specifically, the utilized platform is a quadrotor UAV with a protective outer frame (for safety in case of contact when in very short range to its surroundings), custom–developed for the purposes of autonomous aerial structural inspection.

In particular, the mesh model of a power transformer mock–up structure swas employed by the UC3D–IPP algorithm. For the executed path it was greatly downsampled to a version of 69 vertices and 134 faces and the computed path is shown in Figure 9. The same Figure further illustrates the recorded flight path which indicates the overall good tracking bahavior of the system. It is noted, that the computed path was conducted as a trajectory with average linear velocity $v_t = 0.5$m/s. For the computation of the inspection path, the camera mounting configuration $\Gamma = [0, -15, 0]$deg, $\Pi = [0.075, 0, 0.025]$m and a minimum inspection distance of 0.35m were considered.

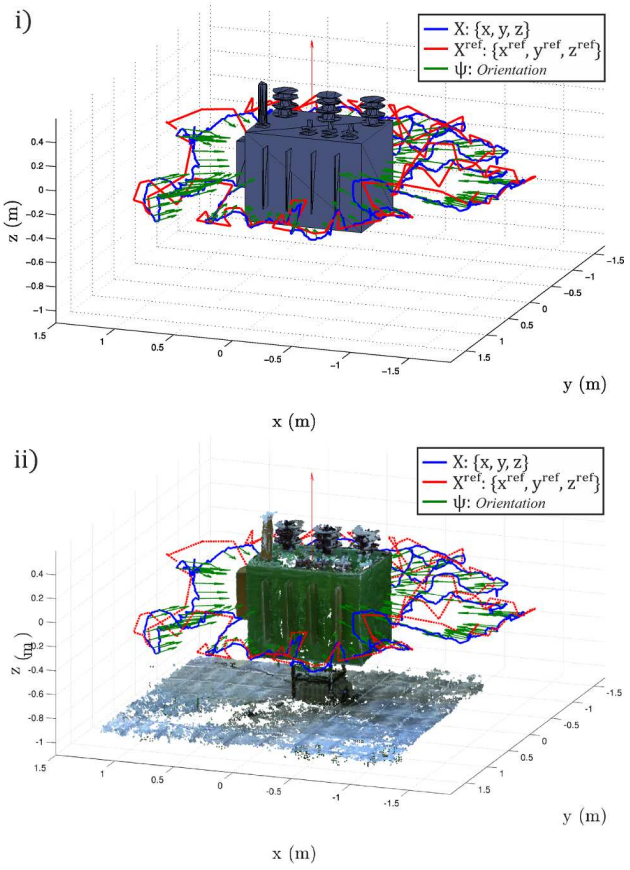


Fig. 9: The uniform coverage inspection path for the case of the power transformer mockup structure and the experimentally recorded flight path using an advanced aerial robotic system equipped with the Kinect RGB-D sensor: i) Model mesh, and ii) extracted dense Point Cloud, illustrate good reconstruction correspondence.

The quality of the inspection path was evaluated using 3D reconstruction techniques using the VGA–quality video footage of the Kinect camera. Despite the low resolution of the camera, the reconstruction depicts that the achieved inspection quality is very high. Details of the structure are clearly visible, and even the texture of the surface –which holds major importance in inspection operations– is clearly discernable. The dense pointcoud and reconstructed mesh of the power transformer mockup are available in the dataset that is jointly released with this paper [9], while part of the on–board video footage can be found in the following url: `https://youtu.be/Gg9qsF3y8IU`

VIII. Conclusions

A new algorithm for 3D structural inspection path–planning was proposed and presented within this paper. The UC3D–IPP method provides full coverage of the 3D structure as modeled by a mesh and ensures uniform (or almost uniform) focus on the geometrical details by appropriately selecting the inspection distance. It provides a framework that computes first solutions extremely fast at the cost of relying on slightly more rough models derived using state–of–the–art mesh simplification techniques. However, as long as more computational time is available, UC3D–IPP will provide new and improved solutions either via a completely new calculation or via a fast iterative strategy that still guarantees full coverage with a small/tunable deviation from inspection distance uniformity. Extensive simulations and experimental evaluation studies using an aerial robot were employed to validate the concepts of the algorithm and showcase its capabilities.

References

[1] A. Bircher, K. Alexis, M. Burri, P. Oettershagen, S. Omari, T. Mantel and R. Siegwart, "Structural inspection path planning via iterative viewpoint resampling with application to aerial robotics," in *IEEE International Conference on Robotics and Automation (ICRA)*, May 2015, pp. 6423–6430.

[2] M. Burri, J. Nikolic, C. Hurzeler, G. Caprari, and R. Siegwart, "Aerial service robots for visual inspection of thermal power plant boiler systems," in *Applied Robotics for the Power Industry, 2012 2nd International Conference on*, 2012.

[3] G. Darivianakis, K. Alexis, M. Burri, and R. Siegwart, "Hybrid predictive control for aerial robotic physical interaction towards inspection operations," in *Robotics and Automation (ICRA), 2014 IEEE International Conference on*, May 2014, pp. 53–58.

[4] K. Alexis, C. Huerzeler, and R. Siegwart, "Hybrid predictive control of a coaxial aerial robot for physical interaction through contact," *Control Engineering Practice*, vol. 32, no. 0, pp. 96 – 112, 2014. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0967066114001762

[5] H. Choset, "Coverage for robotics–a survey of recent results," *Annals of mathematics and artificial intelligence*, vol. 31, no. 1-4, pp. 113–126, 2001.

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

[7] B. Englot and F. S. Hover, "Sampling-based sweep planning to exploit local planarity in the inspection of complex 3d structures," in *Intelligent Robots and Systems (IROS), 2012 IEEE/RSJ International Conference on*. IEEE, 2012, pp. 4456–4463.

[8] G. Papadopoulos, H. Kurniawati, and N. Patrikalakis, "Asymptotically optimal inspection planning using systems with differential constraints," in *Robotics and Automation (ICRA), 2013 IEEE International Conference on*, 2013.

[9] K. Alexis, C. Papachristos, R. Siegwart, A. Tzes, "Uniform Coverage Inspection Path-Planning Datasets." [Online]. Available: https://sites.google.com/site/unicov3d/

[10] S. Valette and J.-M. Chassery, "Approximated centroidal voronoi diagrams for uniform polygonal mesh coarsening," *Computer Graphics Forum*, vol. 23, no. 3, pp. 381–389, 2004.

[11] K. Helsgaun, "An effective implementation of the lin–kernighan traveling salesman heuristic," *European Journal of Operational Research*, vol. 126, no. 1, pp. 106–130, 2000.

[12] S. Karaman and E. Frazzoli, "Incremental sampling-based algorithms for optimal motion planning," *arXiv preprint arXiv:1005.0416*, 2010.

[13] S. Lin and B. W. Kernighan, "An effective heuristic algorithm for the traveling-salesman problem," *Operations research*, vol. 21, no. 2, pp. 498–516, 1973.

[14] C. Papachristos, K. Alexis and A. Tzes, "Model predictive hovering–translation control of an unmanned tri–tiltrotor," in *2013 International Conference on Robotics and Automation*, Karlsruhe, Germany, 2013, pp. 5405–5412.

[15] C. Papachristos, K. Alexis and A. Tzes, "Efficient force exertion for aerial robotic manipulation: Exploiting the thrust-vectoring authority of a tri-tiltrotor uav," in *Robotics and Automation (ICRA), 2014 IEEE International Conference on*, May 2014, pp. 4500–4505.

[16] C. Papachristos, K. Alexis, and A. Tzes, "Dual–authority thrust–vectoring of a tri–tiltrotor employing model predictive control," *Journal of Intelligent & Robotic Systems*, pp. 1–34, 2015. [Online]. Available: http://dx.doi.org/10.1007/s10846-015-0231-1

[17] C. Papachristos, D. Tzoumanikas, and A. Tzes, "Aerial robotic tracking of a generalized mobile target employing visual and spatio—temporal dynamic subject perception," in *Intelligent Robots and Systems, 2015. IROS 2015. IEEE/RSJ International Conference on*, Sept–Oct 2015.