History-aware Autonomous Exploration in Confined Environments using MAVs

Conference Paper

Author(s):
Witting, Christian; Fehr, Marius; Bähnemann, Rik; Oleynikova, Helen; Siegwart, Roland

Publication date:
2018

Permanent link:
https://doi.org/10.3929/ethz-b-000319871

Rights / license:
In Copyright - Non-Commercial Use Permitted

Originally published in:
https://doi.org/10.1109/IROS.2018.8594502

Funding acknowledgement:
608849 - European Robotics Challenges (EC)

History-aware Autonomous Exploration in Confined Environments using MAVs

Christian Witting[1], Marius Fehr[2], Rik Bähnemann[2], Helen Oleynikova[2], and Roland Siegwart[2]

***Abstract*— Many scenarios require a robot to be able to explore its 3D environment online without human supervision. This is especially relevant for inspection tasks and search and rescue missions. To solve this high-dimensional path planning problem, sampling-based exploration algorithms have proven successful. However, these do not necessarily scale well to larger environments or spaces with narrow openings. This paper presents a 3D exploration planner based on the principles of Next-Best Views (NBVs). In this approach, a Micro-Aerial Vehicle (MAV) equipped with a limited field-of-view depth sensor randomly samples its configuration space to find promising future viewpoints. In order to obtain high sampling efficiency, our planner maintains and uses a history of visited places, and locally optimizes the robot's orientation with respect to unobserved space. We evaluate our method in several simulated scenarios, and compare it against a state-of-the-art exploration algorithm. The experiments show substantial improvements in exploration time ($2\times$ faster), computation time, and path length, and advantages in handling difficult situations such as escaping dead-ends (up to $20\times$ faster). Finally, we validate the on-line capability of our algorithm on a computational constrained real world MAV.**

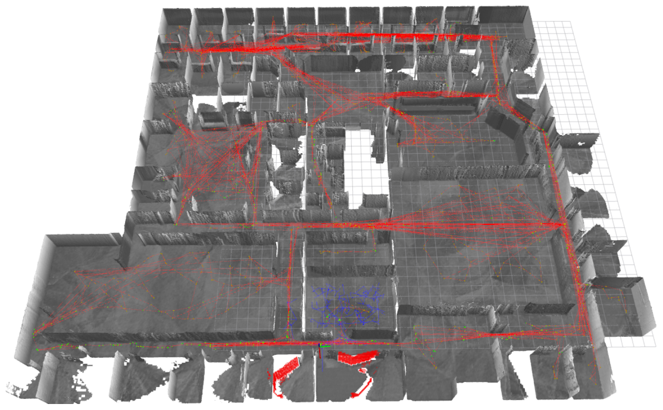


Fig. 1. A map build from sensor data of a partly explored simulation of the willowgarage building. The MAV's position is the axis marker in red, green and blue, with the RRT shown in dark blue. A sparse version of the history graph is shown in red.

I. Introduction

Autonomous mobile robot exploration has a wide variety of applications such as visual inspection tasks, search-and-rescue missions, 3D reconstruction, or mining [1], [2], [3], [4]. More specifically, Micro-Aerial Vehicles (MAVs) allow unique viewpoints and access to confined and narrow environments [5]. The common goal in all exploration applications is to efficiently explore an unknown environment completely.

The research in exploration is classically divided into two fundamental approaches: the frontier-based approach and the sampling-based approach. The frontier approach seeks to maximize the map coverage by identifying and exploring the boundaries between the known and unknown parts of a map, while the sampling-based approach randomly searches the robot's configuration space for sensor poses which maximize a given objective function.

We base our algorithm on the sampling-based approach as it has been proven successful in 3D exploration and can be formulated independently of the underlying objective function, e.g., it allows multi-objective optimization of exploration, scene reconstruction and localization [6], [7], [8]. Unfortunately, sampling-based planner performance deteriorates in large environments or confined scenarios featuring small openings or bottle-necks. In particular tree-based exploration has the tendency to get stuck in dead-end situations where the robot needs to reevaluate and revisit already traversed sections of the map in order to find new exploration gain. Thus a large amount of computation time is wasted on searching already visited places.

In our Next-Best View Planner (NBVP) we introduce three new components to cope with the curse of dimensionality. First, our planner reduces the sampling space by locally optimizing the orientation of the sensor instead of sampling it randomly. Second, we use path simplification and smoothing methods to shorten traversal time. Third, we maintain and use a history of previously visited positions as seeds of the Rapidly-exploring Random Tree (RRT) to quickly find informative regions when the mapped area increases, and the next gain is far away. Figure 1 shows a visualization of the proposed representation of this history.

We evaluate the proposed components by comparing our planner against a state-of-the art information gain exploration algorithm, both quantitatively and qualitatively in small- and large-scale simulation scenarios. Furthermore, we validate the on-board planning capability in a real world MAV experiment.

The main contributions resulting from this work are:

- Boosting the RRT planning performance by using a

[1]C. Witting is a Master student at the Faculty of Electrical Engineering, Technical University of Denmark, Lyngby, Denmark. witting.chr@gmail.com [2]M. Fehr, R. Bähnemann, H. Oleynikova, R. Siegwart are with the Autonomous Systems Lab (ASL), ETH Zürich, Zürich, Switzerland. mfehr, brik, oelena, rsiegwart@ethz.ch This research was supported in part by the European Communitys Seventh Framework Programme (grant number n.608849).

history of exploration potential as seeds.
- Increasing the sampling efficiency by maximizing the local sample gain w.r.t. orientation.
- Employing dynamics-aware trajectory optimization techniques for trajectory refinement.
- Evaluation and validation of our approach in both simulated and real world scenarios.

We organize the paper as follows. Section II presents related work. Section III introduces NBV planning and elaborates on the particular issues in exploration. In section IV we present our planning algorithm. In section V we benchmark the planner against an existing NBVP in simulation and show a real world application before we close the article with concluding remarks in section VI.

II. Related works

Exploration planning deals with the problem of finding a set of sensor poses along the border of an unknown volume such that eventually the whole volume is explored respecting some path cost, e.g., time, length or energy. As such, the problem is closely related to the art gallery problem, the traveling salesman problem, and the problem of finding a shortest path in 3D environments, which are NP-complete individually [9], [10], [11]. Additional planning constraints arise from the MAV's restricted computational and battery resources and its limitations in perception and actuation. The two most established heuristics in MAV exploration planning are frontier-based approaches and sampling-based information gathering approaches.

A. Frontier-based

The frontier-based approach is the classical approach to the exploration problem originally introduced in [12] who also extended it to multiple robots [13]. In a partly explored environment there exists a boundary between known and unknown free space which denotes the frontier. Frontier-based methods extract this boundary from a map and plan a path which visits the nearest boundary.

In [14] and [15] the frontier method has been compared to several different exploration algorithms, and [16] proposes a system which applies frontier exploration to MAVs. However, the MAV is kept at an constant height, and the comparisons are mainly made for the 2D case, whereas our method explores freely in 3D.

Traditionally, the frontier-based methods pick the closest frontier [12]. [17] takes the full 3D movement of the MAV into account when extracting the route to the next frontier. Their planner does not choose the closest frontier, but the one which requires the least change in velocity within the current field of view of the robot. While this approach also achieves faster exploration rates than the NBVP we are comparing against [6], our planner follows the sampling-based approach and thus potentially allows different exploration objectives.

B. Sampling-based

The opposing approach is sampling-based information gathering. Here the fundamental idea is to sample viewpoints in the explored map which could potentially contribute towards the exploration objective. This avoids explicit calculations of frontiers. Since the configuration space is sampled randomly, these planners also allow different optimization objectives without changing the underlying motion planning algorithm [18]. [19] for example uses this approach to generate MAV system identification trajectories.

The concept of NBVs was first introduced in [20], where the authors goal was to obtain a complete model of a scene by calculating a series of covering views. While not dealing explicitly with exploration, the idea of NBVs has been carried over into the exploration domain.

[6] uses NBVs in a 3D exploration algorithm. In a receding horizon fashion, their approach iterates between sampling accessible viewpoints in an RRT and executing the most informative path. Our algorithm builds up on theirs but introduces a memory of already visited spaces, local gain optimization, and trajectory optimization which leads to a significantly better sampling efficiency and shorter and faster explorations.

[7] extends [6]'s NBVP to account for uncertainty in the localization and mapping between viewpoints. In [8] the uncertainty and quality of NBVs has been incorporated with respect to a simultaneous localization and mapping (SLAM) framework. In our work we assume reliable state estimation, and focus on rapid exploration.

C. Other Approaches

[21] and [22] combine frontier- and sampling-based approaches. Their algorithms calculate the frontiers explicitly, but sample the poses from which frontiers can be observed. [21] also uses trajectory optimization methods to convert a piece-wise linear planned trajectory into a smooth path that obeys robot dynamics.

[23] uses particles defined by a stochastic differential equation to explore the environment. [24] calculates an explicit region-of-interest map based on the ideas of the NBVs.

[25] uses time-varying tensor fields to construct a topological skeleton of the map. Here the exploration gain is chosen based on the unknown area which can be scanned from the topological skeleton. This method resembles ours in maintaining the history of exploration potential. However, the tensor fields does not directly translate to the 3D case needed for MAV platforms which our work is based on.

III. Limitations of RRT-based NBV planning

In this section we introduce sampling-based NBV planning, discuss its limitations and motivate the proposed algorithmic changes.

A. Algorithm

3D NBV planning as introduced in [6] follows a receding horizon approach. Here the robot iterates between sampling new viewpoints uniformly in the current voxel map, and navigating towards the NBV.

In the initial step the robot grows an RRT from the current position in (x, y, z, θ)-space, where x, y and z describe the

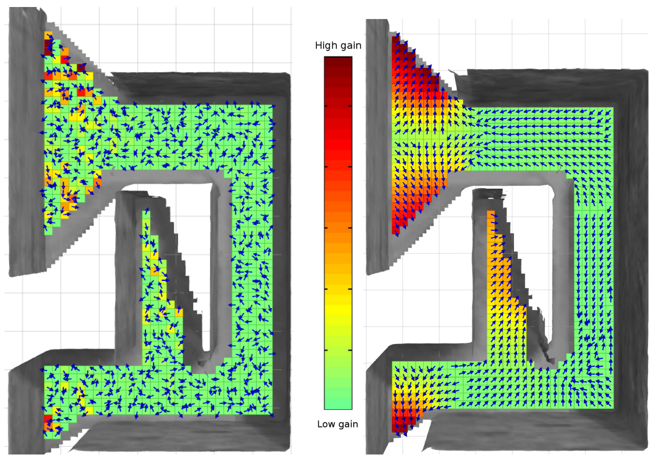


Fig. 2. The expected gain of a grid of poses on a slice of a map using (a) random sampling of the yaw, and (b) yaw optimization. The blue arrows represent the orientation. Grey shows the occluded voxels in the map (in this case the walls and floor).

position and θ describes the yaw orientation [26]. The RRT is the backbone of the planning part, and as such is responsible for finding a collision free straight-line path from the current position to a sample with gain.

For each new sample that is connected to the nearest neighbor in the tree the algorithm calculates the expected information gain. In [6]'s NBVP, the immediate information gain is calculated by counting the number of visible unknown voxels in the sampled camera frustum exponentially discounted by its shortest distance to the current MAV position. The total gain of a node is the summation of all immediate gains along the RRT branch to the node.

This approach of calculating samples and adding them to the tree is repeated for a predefined number of iterations or until a sample with sufficient gain is found.

Once a potential gain is found, the MAV navigates edge by edge along the shortest path found in the RRT towards the node with the highest exploration gain. In the mean time a new RRT iteration begins, discarding all previous samples but the best branch. When a better branch is found the robot will update its goal pose.

B. Drawbacks

One of the limitations of the traditional NBVP for MAVs is the fact that the sampling takes place in the (x, y, z, θ) configuration space. While the sampling in x, y, z is obviously desired in order for the RRT algorithm to propagate through the mapped volume, the sampling of the yaw component limits the sample efficiency of the exploration. For a given sampled position near unobserved voxels it is unlikely that the planner will also sample a yaw orientation that faces the camera in the optimal direction into the unobserved region and thus creates a high gain. This is evident in Figure 2a which visualizes the expected gain when sampling the yaw randomly.

Additionally, RRTs have a bias toward large Voronoi regions[1]. Thus, they are sample efficient to span the Euclidean space $\mathbb{R}^3$ but take long to locally refine which is necessary to obtain different sensor viewpoints in the vicinity of the robot.

A second limitation of [6] is that the extracted waypoint list from the RRT is used directly as trajectory. However, this has the effect that the resulting movement is jagged due to the randomness of the RRT (illustrated in Figure 3a). Which in turn results in both long routes, and high start-stop energy consumption.

Last but not least, the RRT is a tree-based planning structure that always has its root in the current position of the MAV. This results in a behavior where the RRT needs to be discarded and recalculated after finishing a branch. As time passes and the mapped area increases, the distances to the next gain and thus the sampling time increases significantly. This issue is evident in Figure 11a where the next gain is far away from the current position and thus the RRT tree grows immensely large.

IV. The augmented NBVP

Based on the limitations identified in Section III-B we propose an augmented sampling-based NBVP. The main intuition behind our changes compared to [6] is to reduce the sampling space to interesting areas such that the RRT can quickly find a NBV. Our NBVP has a three-step sampling approach and a deterministic yaw policy. Further, it generates simplified and smooth trajectories and maintains a graph of potential RRT seeds in free space as a hot-start in dead-end situations.

The exploration algorithm is described in Algorithm 1. In the initial step the algorithm samples a random tree with potential NBVs within the free space of a user-defined vicinity of the MAV. Each new node gets assigned an exploration gain based solely on the number of unobserved voxels in the expected sensor frustum. If no view with a certain gain is found in the vicinity, the robot may be stuck in a dead-end and will reseed the RRT to the closest node with a potential in the history graph. If the algorithm is still unable to find a gain within the vicinity of the new seed, the sampling will extend to the full free workspace until a sample with gain is found.

Once a viewpoint with sufficient gain is found the MAV computes a smoothed trajectory to the node and repeats the sampling approach. Concurrent to the exploration, the robot updates the voxel map and maintains a potential seed graph. Figure 4 shows the system in comparison to the receding horizon based planner [6].

The mapping framework used for the exploration algorithm is based on Voxblox [27]. Voxblox takes a planning-centric approach to dense mapping, by maintaining both a Truncated Signed Distance Field (TSDF) and Euclidean Signed Distance Field (ESDF). We use this representation to perform ray casting in the frustum optimization, collision

[1]See `http://msl.cs.uiuc.edu/rrt/gallery_2drrt.html`.

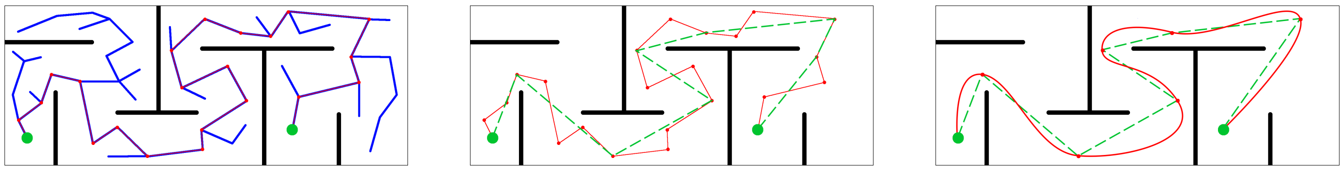

(a) (b) (c)

Fig. 3. Illustration of the trajectory smoothing stages. (a) Shows the RRT tree (blue) between the start and end node (green) together with the extracted raw branch (red). (b) Shows the minimum waypoint branch (green), and (c) shows the optimized polynomial trajectory in red.

Algorithm 1 Exploration

```
1:  while Map is not explored do
2:      Set sampling bounds to root vicinity
3:      while No gain found in RRT do
4:          if Initial sampling fails then
5:              Seed RRT with closest node in history
6:              Update sampling bounds
7:          end if
8:          if Reseeded sampling fails then
9:              Increase sampling bounds to full free space
10:         end if
11:         Sample within bounds
12:         Grow RRT
13:     end while
14:     Extract best branch from RRT
15:     Simplify and smooth trajectory
16:     Carry out trajectory
17: end while
```

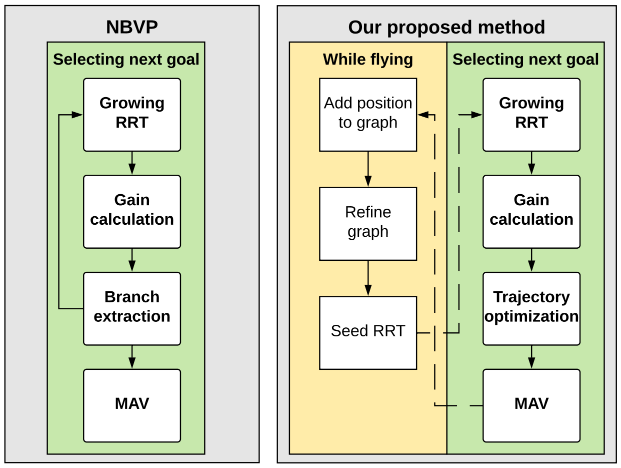


Fig. 4. A comparison of the system structure of the traditional NBVP [6] and our proposed method.

checking in trajectory optimization and to refine the history graph.

The three-step sampling approach increases the chances of finding a close NBV and escape dead-ends early. In the following we will elaborate on the extensions in orientation optimization, trajectory smoothing, and history maintainance.

A. Local gain optimization

Instead of sampling in (x,y,z,θ)-space, we propose to sample only in Euclidean (x,y,z)-space and set θ deterministically to the optimal direction for each sample. The optimal yaw direction is found by ray casting the sensor frustum in the set of N discrete orientations $\theta \in \{-\pi, -\pi+\Delta\theta, \dots, \pi-\Delta\theta\}$, where the discrete step size should ideally be set to $\Delta\theta = \frac{r}{R}$ to ensure that all voxels within the view frustum are considered. However, to balance the computational complexity with accuracy this value was increased to 5° in the experiments here. Here r is the voxel size of the map in meters, and R is the desired projection range in meters.

We approximate the viewing frustum as a circular sector of a cylinder, with radius R, height $H = 2R\sin\left(\frac{v_{\text{fov}}}{2}\right)$ and central angle h_{fov}, where v_{fov} and h_{fov} are the vertical and horizontal field of view of the camera respectively.

To compute the exploration gain for all N frusta, we first precompute the gain of N vertical slices over the whole cylinder originating from each sensor pose. For each of these slices we cast rays from the sensor origin to the voxels on the vertical boundary line of the cylinder. The gain of the slice is the summation of the number of unknown voxels along the rays. If a ray intersects with an obstacle in the map it is stopped short, in order to not count occluded unknown areas behind obstacles.

The total gain for each direction is found by summing the slice gains over a window with angular width h_{fov}. The direction with the largest gain is then the optimal direction to face for that specific sample.

Figure 2 shows a visual comparison of the expected gain between randomly sampled orientations and optimized orientations where it can be seen how the gain is large when close to the regions with unexplored parts. As mentioned in Section III-B it is obvious from Figure 2a that the random sampling misses a lot of the gain in several cases leading to degraded performance. Furthermore, it can be seen from Figure 2b that the yaw optimization results in finding viewpoints that consistently point towards the frontiers of the map to achieve map coverage.

B. Solution selection and trajectory generation

In the original NBVP, [6], the MAV chooses a straight-line path according to the distance and immediate gains along the branch of the tree (see Figure 3a). In our solution, we do not consider intermediate gain or distance along the path towards the NBV. Our planner directly navigates to the first viewpoint with sufficient exploration gain. Since we locally optimize the orientation, the first informative sampled pose is expected to automatically be the closest gain to the seed of the RRT. This simplification allows discarding intermediate waypoints and performing trajectory optimization.

The first step takes the jagged branch and converts it into a minimum straight-line trajectory. The sparse waypoint list is then interpolated by continuous polyonomial trajectories as presented in [28] and implemented in [29][2]. This results in short, smooth, and dynamically feasible trajectories as illustrated in Figure 3.

C. History maintenance

As mentioned in the drawbacks in Section III-B, the RRT exploration degrades in large and confined areas when the robot has to find its way towards distant information sources. In this section we specifically address this issue by introducing a graph in free space that stores knowledge about the spatial distribution of information in already explored parts of the map. The nodes of the graph are used to reseed the RRT when no gain is found in the vicinity of the current root. Algorithm 2 shows the full history graph algorithm that runs concurrently to the exploration algorithm.

The history graph nodes consist of positions in free space that are sampled along the travelled path. For each node, we store a measure of the exploration potential while the MAV is navigating. This potential is a representation of exploration gain which is nearby, but not necessarily in view, and is specific to the chosen exploration objective.

To update a node, we perform a Breadth First Search (BFS) over the free map voxels[3] in its vicinity to count the number of voxels with a potential gain (in our case, we essentially count reachable frontier voxels). Here the BFS ensures that there exist a collision-free but not necessarily direct connection to the gain. By doing this for each node in the history graph, the potential for all the previously visited positions is kept up to date.

To mitigate the effects of changes in the explored map (i.e. due to mapping or localization inaccuracies), we attempt to maximize the clearance to obstacles of nodes in the history graph. This is done by ascending the obstacle distance gradient in the Voxblox ESDF map when refining the poses pushing them away from the nearest obstacle. This effectively warps the graph into the points of equal distance to the obstacles ensuring the connections remain collision free, approximating a Voronoi graph of the map.

As the exploration progresses, some of the nodes in the history graph become redundant when they collapse onto the same position during the refinement. These are continuously combined while keeping the connectivity during the maintenance step.

Figure 9 shows a complete history graph in an explored scenario.

[2]See `github.com/ethz-asl/mav_trajectory_generation`.

[3]It should be noted that the BFS is employed on the map representation (Voxblox), not the history graph.

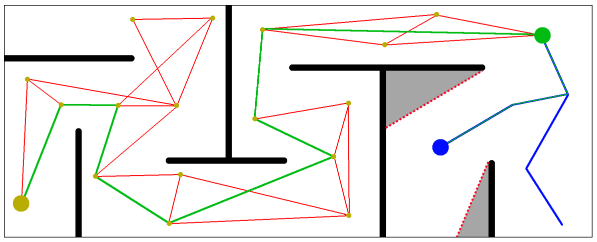


Fig. 5. Illustration of the history seeding. Gray denotes unknown area with the current frontier in red. The large golden circle is the current position, and the red graph is the history graph. The golden nodes are previous poses which do not have any exploration potential, while the green node is the closest that does. It has thus been chosen as seed for the RRT tree (shown in blue). The green branch is the best branch through the history graph and RRT to the large blue node which has an exploration gain.

D. Seeding

As described in Algorithm 1 the RRT is reseeded whenever the robot cannot find informative poses in its vicinity. In this case the closest node with potential is extracted from the history graph, and used as a seed for the RRT. This puts the root of the RRT in the vicinity of unexplored gain, and thus the sampling time is significantly reduced, as the RRT no longer needs to sample all the way from the current position. The shortest path from the current position to the root of the RRT can then be extracted from the history graph. Figure 5 illustrates this process.

A comparison of a specific scenario in a dead-end can be seen on Figures 11a and 11b.

Algorithm 2 History maintenance

```
 1: while Robot moving do
 2:     Save and connect current pose to graph
 3:     for Each node in graph do
 4:         Refine position of node with gradient from ESDF
 5:         if Connectivity is broken then
 6:             Discard refinement
 7:         end if
 8:         Recalculate potential
 9:         if No potential then
10:             Add to set without potential
11:         end if
12:         Combine collapsed nodes
13:     end for
14: end while
```

V. Results

We tested our algorithm both in simulation and on a real platform. In simulation we benchmark it in a small and a large maze scenario against [6]'s sampling-based NBVP, stressing the reduced computation time, exploration time and exploration path length of our approach. In the real-world experiment, we validate the online-capability of the planner.

The specific details of the platforms used for the evaluation in this work has been highlighted in Table I.

Scenario	Platform	Processor	Vision sensor
Simulation	Gazebo 7.0 + RotorS [30]	Intel i7-4700MQ @ 2.4 MHz	Simulated VI-sensor
Real world	Astec Firefly	Astec Mastermind, Intel Core i7 processor	VI-sensor @ 752×480 px [31]

TABLE I

DETAILS OF THE PLATFORMS USED IN THIS WORK.

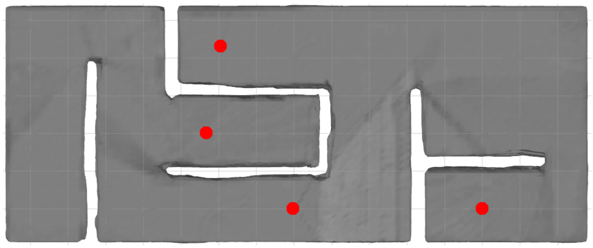


Fig. 6. The map of the small test scenario ($15.5\,\mathrm{m} \times 6.5\,\mathrm{m}$). The red dots are the different starting positions used for the simulations.

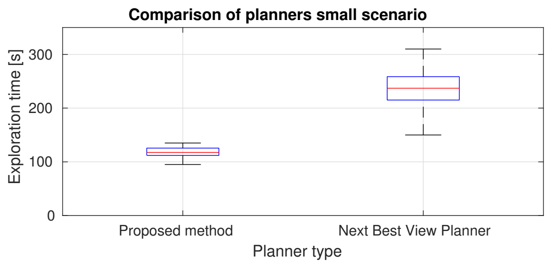


Fig. 7. The boxplot for 20 exploration runs in the small scenario with our planner without history and [6] at the same maximum velocity $V_{max} = 1.2\,\mathrm{m\,s^{-1}}$. Our improvements clearly reduces the exploration time and variance of the approach.

An accompanying video of the proposed method in a range of these experiments can be found on:
`https://youtu.be/Rp2bIH_e9ig`

A. Small scenario

We use the small maze scenario shown in Figure 6 to quantitatively compare the two approaches. As both the approaches are stochastic, they are run multiple times to account for the variance in the approach. In this scenario the algorithms were executed 20 times each from several different starting spots. We omitted the reseeding in our algorithm in this scenario as it is too small to take effect. Figure 7 presents the resulting exploration times. The proposed method is on average approximately $2\times$ faster, and has a significantly smaller variance in the exploration time, showing the merits of the proposed yaw optimization, trajectory smoothing, and vicinity sampling. Especially the yaw optimization contributes to the more deterministic behavior of the algorithm.

B. Large scenario

In the larger maze scenario shown in Figure 9 we compare both approaches qualitatively. The resulting statistics from a single experiment can be seen in Figure 8. It shows that the proposed method finds a solution that is both faster and

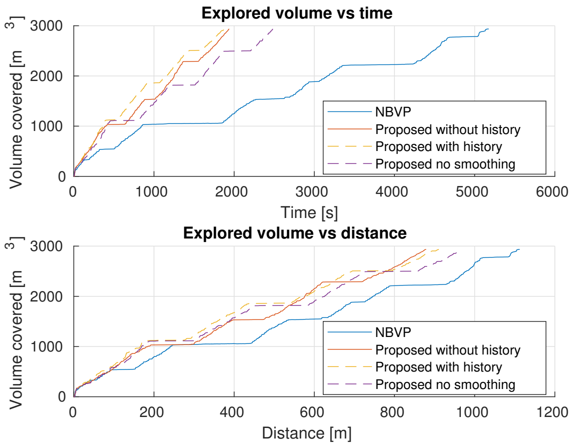


Fig. 8. A comparisons with the NBVP and the proposed method in the large scenario with $V_{max} = 1.2\,\mathrm{m\,s^{-1}}$. The proposed method clearly outperforms the NBVP. Furthermore, trajectory optimization results in faster and shorter exploration.

shorter. It also shows the advantage of trajectory simplification and smoothing which results in shorter trajectories with consistent velocity.

A more thorough comparison of the proposed method and the history graph can be seen in Figure 10a. Here we ran the large scale experiment at higher velocities which is feasible for the controller with the trajectory optimization but infeasible with [6]'s waypoint following. The plot shows that the history graph results in reduction of the time needed to explore the large scenario.

C. Computation time

As mentioned in Section III-B one of the issues with the NBVP was its performance in dead-ends. We have specifically evaluated this scenario here in the case of the large experiment. Here the MAV's ability to escape the dead-end was evaluated. The results in Figure 11, show how the seeding of the RRT in this case reduced the computation time significantly from around 35 s to around 1.4 s. Furthermore, Figure 11a shows the excessive tree structure from the RRT without history is shown in blue. The equivalent tree in Figure 11b is much smaller.

Figure 10b shows the computation time for the 10 experiments on the large scale scenario and the corresponding exploration results has been portrayed in Figure 10a. The plot shows a significant reduction of the expected maximum computation time per iteration from 38 s to 2.2 s.

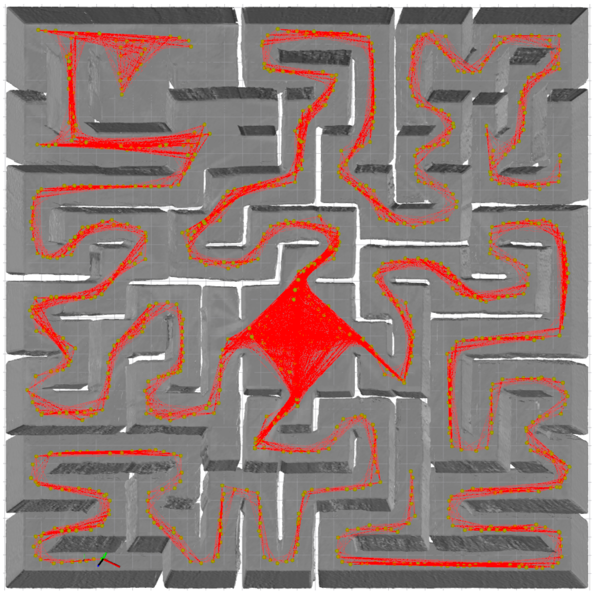
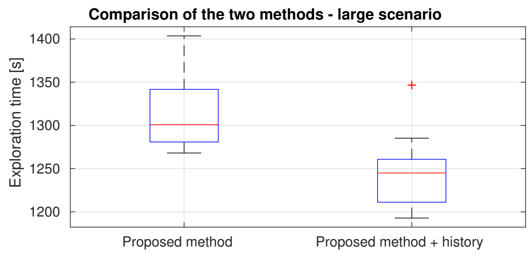
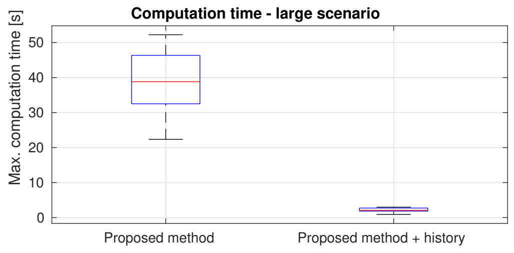


Fig. 9. The fully explored map of the large scenario (30 m × 30 m), together with the history graph. Red shows the edges in the history graph, the yellow nodes indicate the nodes with no potential. The current position of the MAV is marked with an Axis in the lower left corner.



(a)



(b)

Fig. 10. The statistics for 10 experiments for each method in the large scenario at $V_{max} = 4.5\,\mathrm{m\,s^{-1}}$. (a) shows an average reduction in exploration time of 5% by using the history graph. (b) shows a reduction of the worst case computation time per iteration from 38 s to 2.2 s in the experiments.

D. Real world experiment

The algorithm was also put to test in the real world. Here it was tested in a small room to validate that it could be run onboard a MAV. Everything was run on-board with the robot using a limited field-of-view depth sensor and VoxBlox [27] for mapping. The on-board state estimation was done with Rovio [32] with the recently developed localization extension Rovioli [33], and thus no external sensing was used.

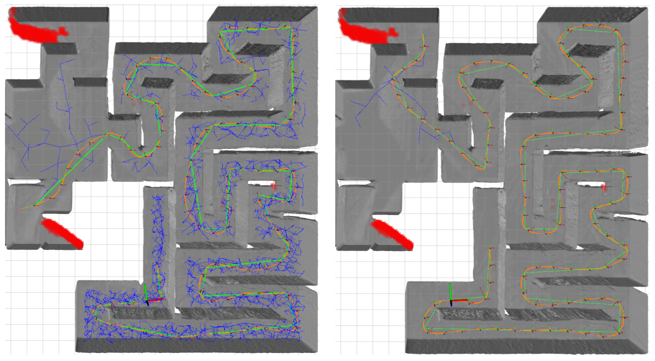
The experiment ran in a semi-autonomous fashion. This was done by first letting the safety pilot map a small section of the room to give a starting point for the algorithm. As the room which the experiments was conducted in was open the history feature was disabled, as there was always a potential in the MAV's current position (i.e. there was always directly reachable frontier voxels from the position of the robot) it would not have had any effect.

On each iteration the chosen trajectory was manually approved by a human supervisor. This was done from a safety perspective such that the safety pilot can always operate from a safe position.

A figure of the constructed Voxblox map in this scenario can be seen on Figure 12.

VI. Conclusion

In this work we presented an improved sampling-based NBVP for autonomous exploration. Statistical simulations show a reduction in exploration time by a factor of 2 over an existing sampling-based planner. Our planner overcomes the curse of dimensionality of sampling-based exploration by guiding the sampling towards informative regions. This was accomplished by introducing a history of interesting places to hot-start the exploration algorithm, which reduces the worst-case computation time to find the NBV from 38 s to 2.2 s in our experiments. We formulated a geometrically optimized orientation policy which reduces the sampling space and thus the variance of the approach. The planned trajectories were optimized to be fast, short, and dynamically feasible which allows exploration velocities of up to $V_{max} = 4.5\,\mathrm{m\,s^{-1}}$ and reduces the overall exploration time. Furthermore, we showed on-board capability in a real exploration experiment on a MAV.



(a) (b)

Fig. 11. Our method running on the large maze scenario, with a dead-end situation highlighted. Here the red part is the frontier, blue is the RRT, and green is the best branch. In (a) The tree is grown using pure RRT growing from the current position ($t_{comp} = 35.3\,\mathrm{s}$). In (b) the RRT is reseeded using the history graph ($t_{comp} = 1.4\,\mathrm{s}$) resulting in a much smaller tree.

References

[1] S. Thrun, S. Thayer, W. Whittaker, C. Baker, W. Burgard, D. Ferguson, D. Hahnel, D. Montemerlo, A. Morris, Z. Omohundro, C. Reverte, and W. W, "Autonomous exploration and mapping of abandoned mines," *IEEE Robotics Automation Magazine*, vol. 11, pp. 79–91, Dec 2004.

[2] D. Calisi, A. Farinelli, L. Iocchi, and D. Nardi, "Autonomous exploration for search and rescue robots," *Wit Trans B*, vol. 94, pp. 305–314, 2007.

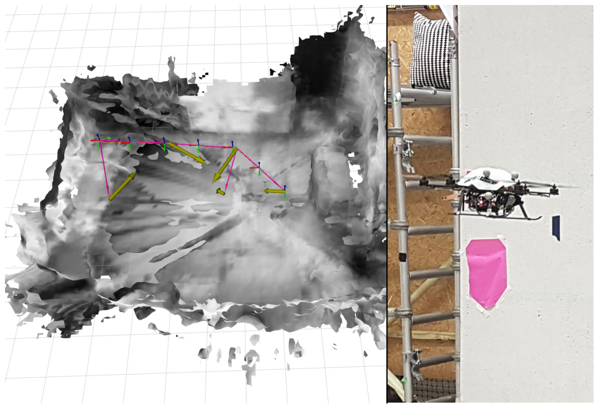


Fig. 12. The map constructed in the real world scenario together with the MAV platform used for the experiment. The dimensions of the room was approximately 9 m × 6 m. The pink arrows show edges in the RRT and the yellow arrows show next best sensor poses and gains.

[3] L. Yoder and S. Scherer, "Autonomous exploration for infrastructure modeling with a micro aerial vehicle," *Springer Tracts in Advanced Robotics*, vol. 113, pp. 427–440, 2016.

[4] J. Rosenblatt, S. Williams, and H. Durrant-Whyte, "A behavior-based architecture for autonomous underwater exploration," *Information Sciences*, vol. 145, no. 1-2, pp. 69–87, 2002.

[5] Z. Fang, S. Yang, S. Jain, G. Dubey, S. Roth, S. M. Maeta, S. T. Nuske, Y. Wu, and S. Scherer, "Robust autonomous flight in constrained and visually degraded shipboard environments," *Journal of Field Robotics*, September 2016.

[6] A. Bircher, M. Kamel, K. Alexis, H. Oleynikova, and R. Siegwart, "Receding horizon next-best-view planner for 3d exploration," *Proceedings - Ieee International Conference on Robotics and Automation*, vol. 2016-, pp. 7487281, 1462–1468, 2016.

[7] C. Papachristos, S. Khattak, and K. Alexis, "Uncertainty-aware receding horizon exploration and mapping using aerial robots," *Proceedings - Ieee International Conference on Robotics and Automation*, pp. 7989531, 4568–4575, 2017.

[8] H. H. Gonzalez-Banos and J.-C. Latombe, "Navigation strategies for exploring indoor environments," *International Journal of Robotics Research*, vol. 21, no. 10-11, pp. 829–848, 2002.

[9] J. O'Rourke and K. Supowit, "Some np-hard polygon decomposition problems," *IEEE Transactions on Information Theory*, vol. 29, no. 2, pp. 181–190, 1983.

[10] C. H. Papadimitriou, "The euclidean travelling salesman problem is np-complete," *Theoretical computer science*, vol. 4, no. 3, pp. 237–244, 1977.

[11] J. Canny and J. Reif, "New lower bound techniques for robot motion planning problems," in *Foundations of Computer Science, 1987., 28th Annual Symposium on*, pp. 49–60, IEEE, 1987.

[12] B. Yamauchi, "A frontier-based approach for autonomous exploration," *1997 Ieee International Symposium on Computational Intelligence in Robotics and Automation - Cira '97, Proceedings*, pp. 146–151, 1997.

[13] B. Yamauchi, "Frontier-based exploration using multiple robots," *Proceedings of the Interantional Conference on Autonomous Agents*, pp. 47–53, 1998.

[14] D. Holz, N. Basilico, F. Amigoni, and S. Behnke, "Evaluating the efficiency of frontier-based exploration strategies," *Joint 41st International Symposium on Robotics and 6th German Conference on Robotics 2010, Isr/robotik 2010*, vol. 1, pp. 36–43, 2010.

[15] M. Julia, A. Gil, and O. Reinoso, "A comparison of path planning strategies for autonomous exploration and mapping of unknown environments," *Autonomous Robots*, vol. 33, no. 4, pp. 427–444, 2012.

[16] F. Fraundorfer, L. Heng, D. Honegger, G. H. Lee, L. Meier, P. Tanskanen, and M. Pollefeys, "Vision-based autonomous mapping and exploration using a quadrotor mav," *Proceedings of the ... Ieee/rsj International Conference on Intelligent Robots and Systems*, pp. 4557–4564, 2012.

[17] T. Cieslewski, E. Kaufmann, and D. Scaramuzza, "Rapid exploration with multi-rotors: A frontier selection method for high speed flight," *2017 Ieee/rsj International Conference on Intelligent Robots and Systems (iros)*, pp. 2135–2142, 2135–2142, 2017.

[18] G. A. Hollinger and G. S. Sukhatme, "Sampling-based robotic information gathering algorithms," *The International Journal of Robotics Research*, vol. 33, no. 9, pp. 1271–1287, 2014.

[19] R. Bähnemann, M. Burri, E. Galceran, R. Siegwart, and J. Nieto, "Sampling-based motion planning for active multirotor system identification," in *Robotics and Automation (ICRA), 2017 IEEE International Conference on*, pp. 3931–3938, IEEE, 2017.

[20] C. Connolly, "The determination of next best views," *Proceedings. 1985 Ieee International Conference on Robotics and Automation*, vol. 2, pp. 432,433,434,435, 432–435, 1985.

[21] B. Charrow, G. Kahn, S. Patil, S. Liu, K. Goldberg, P. Abbeel, N. Michael, and V. Kumar, "Information-theoretic planning with trajectory optimization for dense 3d mapping," *Robotics: Science and Systems*, vol. 11, 2015.

[22] A. Visser, Xingrui-Ji, M. Van Ittersum, L. A. Gonzlez Jaime, and L. A. Stancu, "Beyond frontier exploration," *Lecture Notes in Computer Science (including Subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, vol. 5001, pp. 113–123, 2008.

[23] S. Shen, N. Michael, and V. Kumar, "Autonomous indoor 3d exploration with a micro-aerial vehicle," *Proceedings - Ieee International Conference on Robotics and Automation*, pp. 6225146, 9–15, 2012.

[24] R. Grabowski, P. Khosla, and H. Choset, "Autonomous exploration via regions of interest," *Ieee International Conference on Intelligent Robots and Systems*, vol. 2, pp. 1691–1696, 2003.

[25] K. Xu, L. Zheng, Z. Yan, G. Yan, E. Zhang, M. Niessner, O. Deussen, D. Cohen-Or, and H. Huang, "Autonomous reconstruction of unknown indoor scenes guided by time-varying tensor fields," *Acm Transactions on Graphics*, vol. 36, no. 6, p. a202, 2017.

[26] J. J. Kuffner and S. M. LaValle, "Rrt-connect: An efficient approach to single-query path planning," in *Robotics and Automation, 2000. Proceedings. ICRA'00. IEEE International Conference on*, vol. 2, pp. 995–1001, IEEE, 2000.

[27] H. Oleynikova, Z. Taylor, M. Fehr, R. Siegwart, and J. Nieto, "Voxblox: Incremental 3d euclidean signed distance fields for on-board mav planning," *2017 Ieee/rsj International Conference on Intelligent Robots and Systems (iros)*, pp. 1366–1373, 1366–1373, 2017.

[28] C. Richter, A. Bry, and N. Roy, "Polynomial trajectory planning for aggressive quadrotor flight in dense indoor environments," in *Robotics Research*, pp. 649–666, Springer, 2016.

[29] M. Burri, H. Oleynikova, M. W. Achtelik, and R. Siegwart, "Real-time visual-inertial mapping, re-localization and planning onboard mavs in unknown environments," in *Intelligent Robots and Systems (IROS), 2015 IEEE/RSJ International Conference on*, pp. 1872–1878, IEEE, 2015.

[30] F. Furrer, M. Burri, M. Achtelik, and R. Siegwart, *Robot Operating System (ROS): The Complete Reference (Volume 1)*, ch. RotorS—A Modular Gazebo MAV Simulator Framework, pp. 595–625. Cham: Springer International Publishing, 2016.

[31] 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 - Ieee International Conference on Robotics and Automation*, pp. 6906892, 431–437, 2014.

[32] M. Bloesch, M. Burri, S. Omari, M. Hutter, and R. Siegwart, "Iterated extended kalman filter based visual-inertial odometry using direct photometric feedback," *International Journal of Robotics Research*, vol. 36, no. 10, pp. 1053–1072, 2017.

[33] T. Schneider, M. Dymczyk, M. Fehr, K. Egger, S. Lynen, I. Gilitschenski, and R. Siegwart, "maplab: An open framework for research in visual-inertial mapping and localization," 2017.