The International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences, Volume XLVIII-1/W1-2023
12th International Symposium on Mobile Mapping Technology (MMT 2023), 24–26 May 2023, Padua, Italy

FUELING GLOCAL: OPTIMIZATION-BASED PATH PLANNING FOR INDOOR UAVS IN AN AUTONOMOUS EXPLORATION FRAMEWORK

Marco Cella *, Francesco d'Apolito, Phillipp Fanta-Jende, Christoph Sulzbachner

AIT – Austrian Institute of Technology - Center for Vision, Automation and Control – Assistive and Autonomous Systems
(marco.cella, francesco.dapolito, phillipp.fanta-jende, christoph.sulzbachner)@ait.ac.at

KEY WORDS: Optimal Path Planning, B-Splines, Autonomous Exploration, Indoor UAV, Crisis and Disaster Management, Trajectory Estimation, SITL

ABSTRACT:

Exploration is a fundamental problem in robotics that requires robots to navigate through unknown environments to autonomously gather information about their surroundings while executing collision-free paths. In this paper, we propose a method for producing smooth paths during the exploration process in indoor environments using UAVs to improve battery efficiency and enhance the quality of pose estimation. The developed framework is built by merging two approaches that represent the state of the art in the field of autonomous exploration with UAVs. The overall exploration logic is given by GLocal, a paper that introduces an hybrid, *i.e.* both sampling-based and frontier-based, framework that is able to cope with the issue of odometry drift when exploring indoor environments due to the absence of absolute localization, *e.g.* through GNSS. The second paper is FUEL, which introduces a frontier-based exploration methodology which computes the drone's path as an optimized non-uniform B-Spline. The framework described in this paper borrows the optimized B-Spline trajectory generation from FUEL and implements it in GLocal. The presented system is evaluated in two different simulated environments, which show the pros and the cons of such method.

1. INTRODUCTION

Autonomous exploration with indoor Unmanned Aerial Vehicles (UAVs) has become an increasingly popular approach for mapping indoor environments, as can be seen in recent papers such as (Karam et al., 2022). Thanks to advancements in sensor technologies and robotics algorithms, such as SLAM, path planning, machine learning, and low-level control, UAVs are now able to navigate complex and cluttered indoor environments with greater ease than before. UAVs are ideal for mapping indoor environments due to their speed, agility, and accuracy. However, planning a safe and efficient path indoors can be difficult, particularly in scenarios such as Crisis and Disaster Management (CDM) where low visibility and obstacles such as furniture and rubble can impede the process. Furthermore, indoor settings in CDM missions might be unstructured, thus planarity and orthogonality assumptions on the environment can't be leveraged to support the trajectory generation process. Moreover, estimating the UAV's position and orientation can be difficult without the availability of Global Navigation Satellite System (GNSS) signals, forcing a reliance on relative state estimation. These factors necessitate the careful development of path planning algorithms that account for the unique characteristics of indoor environments in order to enable efficient and collision-free indoor navigation.

Despite advancements in autonomous exploration with indoor UAVs (Zhang et al., 2022), there is still a need to enhance mapping efficiency. This can be done by optimizing the trajectory generation process, thus making the UAV move more efficiently, covering the entire indoor environment more quickly and with fewer movements. This involves creating a path for the UAV that considers factors such as dynamic feasibility, collision avoidance, energy management, and execution time.

This paper aims at improving the state of the art introduced in GLocal (Schmid et al., 2021) by optimizing the execution of the path computed by the already existing Receding Horizon Rapidly Exploring Random Tree (RH-RRT) with a local trajectory planning method based on B-Splines that allows the quadrotor to fully utilize its dynamic capabilities. Figure 1 shows an example of a local B-Spline. This paper investigates non-uniform B-Splines instead of other techniques such as minimum-snap based trajectory generation (Mellinger and Kumar, 2011), approaches based on Deep Reinforcement Learning (Sun et al., 2021) and control-based methodologies (Kulathunga and Klimchik, 2022). This choice is motivated by their unique properties, which will be explained in Section 3, and their proven performance in computing trajectories for fast flight in 3D complex environments (Zhou et al., 2019). It contributes to the literature in two aspects:

- It improves the performance of a state of the art autonomous exploration algorithm.
- The non-uniform B-Spline approach to path planning built on top of the original GLocal framework adds to the scarce literature of fast autonomous exploration under odometry drift.

2. RELATED WORK

A. Autonomous Exploration

The goal of Exploratory Path Planning (EPP) algorithms is to autonomously plan paths that cover an entire area that needs to be mapped. Multiple solutions have been proposed in the cases where the map's occupancy grid is known a priori, such as (Shen et al., 2021) and (Rottmann et al., 2021). These approaches that assume a known map are not suitable for completely unknown environments, since a global plan can't be computed a priori. Methods to tackle the exploration of

* Corresponding author

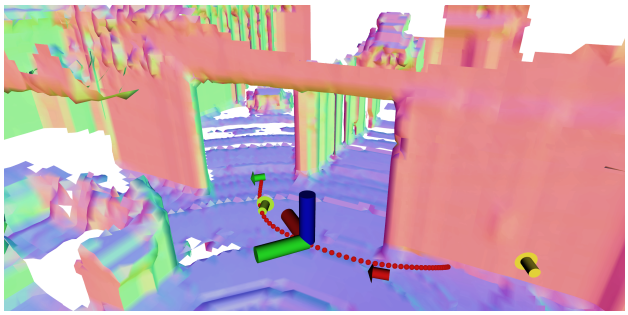

Figure 1. An example of the B-Spline trajectories that our algorithm produces in real-time. The colored (pink, light green and violet) area in the background represents the mesh of the explored volume, the small arrows the sampled viewpoints pointing towards the optimal yaw, the thick axes the flying UAV and the red curve the sampled B-Spline that connects the selected best viewpoints. The simulation is running in Gazebo.

unknown space have been researched since 1981 (Kuipers and Byun, 1991), when a method to autonomously explore a previously unknown 2D area while considering sensorimotor errors was proposed. Over the years, new and more efficient techniques have been researched and developed, and they can be divided in three main classes: sampling-based, frontier-based and hybrid. Most of the state of the art approaches assume perfect knowledge of the robot's position at all times in the map, ignoring the estimation uncertainties that surround real missions. For example, both the authors of (Duberg and Jensfelt, 2022) and (Schmid et al., 2020) use a motion capture system in their real world experiments to remove the errors introduced by state estimation drift. A solution to this problem has been proposed by (Zhang et al., 2022), in which the approach includes a Truncated Signed Distance Field (TSDF) mapping framework with real-time re-integration and frame pruning based on a set cover selection criterion. Key frames are re-integrated with updated camera poses to rectify the map distortions caused by the localization drift whenever a loop closure is detected. The authors also introduced a novel key frame selection method based on set cover formulation and space partitioning improving real-time performance. Furthermore, the localization drift is reduced by implementing an active loop closing strategy in the path planning process. Another paper that proposed a solution to this issue is GLocal (Schmid et al., 2021), the paper on which our methodology builds upon. It introduces the idea of actively tackling the pose drift problem that naturally arises in real-word scenarios of indoor exploration. The core idea is to combine various layers of mapping and planning to achieve the goal. A sliding window map is used to overcome the problem of inaccurate state estimation, which might lead to collisions with the environment. This map is built exclusively on up-to-date sensor data so that reliable collision checking can be performed. Sliding window maps are periodically stored as sub-maps, which are then organized in an overall global map. Whenever a sub-map is added to the global one, its traversability graph is computed and connected to the graphs of overlapping sub-maps. To ensure collision-free path planning even under pose drift, the local planner samples viewpoints only within the sliding window, which are then connected forming a graph. The computed most informative path is then executed in a Receding Horizon - Next Best View (RH-NBV) fashion. If none of the sampled viewpoints returns a high enough information gain, the planner plans a path over the frontiers in the global map. The assumption behind this step is that even small frontiers can lead to discovering large unknown areas. Thus, the closest frontier according to the Euclidean distance is chosen as a goal. To reach it while still guaranteeing a collision-free path, the trajectory is computed over the previously saved traversability graph of each submap. They show that this approach leads to safe exploration even under strong odometry drift.

B. Optimization-Based Trajectory Planning

To improve GLocal's explored volume over time, the original receding-horizon planning logic is replaced with a method based on FUEL (Zhou et al., 2021). It involves the computation of a smooth minimum-time non-uniform B-Spline trajectory that enables the UAV to navigate through the sampled viewpoints of the RRT by fully utilizing its dynamic capabilities. Many optimization-based trajectory generation methods have been proposed and implemented in the field of robotics, and existing methods can be broadly categorized as hard-constrained or soft-constrained methods. Hard-constrained methods, such as minimum-snap trajectory generation (Mellinger and Kumar, 2011), represent trajectories as piecewise polynomials and generate them by solving a quadratic programming problem. These methods ensure global optimality but are limited by conservative kinodynamic constraints that can result in slow trajectories. Soft-constrained methods such as (Park et al., 2021), on the other hand, use non-linear optimization to generate smooth trajectories while also considering safety constraints. However, they suffer from local minima and have no strong guarantees of convergence nor kinodynamic feasibility. Other methods take a two-step pipeline approach, where a collision-free path is planned first, and then the smoothness and time-allocation of the relevant trajectory are optimized based on the shape of the path. For instance, B-Splines, Bézier curves, and piecewise polynomials are commonly used to represent shape-constrained trajectories in cluttered environments. Sampling-based (Karaman and Frazzoli, 2011) and searching-based (Likhachev et al., 2003) methods are used to plan a collision-free path, while gradient-based methods (Ratliff et al., 2009) are employed to guarantee smoothness and dynamical feasibility. However, these methods separate the trajectory shape and the trajectory parametrization, which can lead to sub-optimal or unfeasible trajectories since the system behaviour over time is not taken into consideration. This makes it important to choose appropriate methods that consider the system's dynamics when generating trajectories. To address these issues, some recent methods consider both the trajectory shape and the dynamics at the same time. For example, (Usenko et al., 2017) uses an optimization-based method which cost function considers smoothness, dynamic constraints, and obstacle avoidance.

In this paper, we propose an autonomous exploration framework that expands the original approach of GLocal (Schmid et al., 2021) with a real-time optimization-based trajectory generation process based on the work of (Zhou et al., 2021) to enable faster exploration in indoor environments.

3. PROPOSED METHOD

Figure 2 describes the overall structure of our approach. Everything begins from the data coming from the onboard sensors the drone is equipped with, *i.e.* a 3D lidar with a 360°

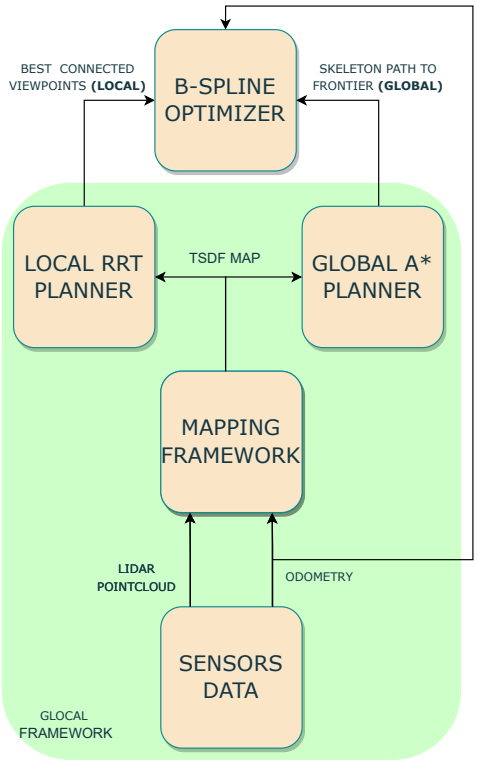

Figure 2. The structure of the proposed algorithm.

horizontal field of view (FOV) and a 59° vertical FOV and an Inertial Measurement Unit (IMU). The mapping and sampling components are kept almost identical to the original implementation of GLocal in order to maintain the safety guarantees when exploring environments under sever odometry drift. The only difference is that, given the use of the 360° lidar mounted with no pitch relative to the UAV's frame of reference, the yaw optimization process carried out while sampling the RRT is not needed anymore. This optimization was based on (Witting et al., 2018), where the yaw that optimizes the amount of unmapped voxels in the sensor frustum for each sampled viewpoint is computed. By removing this component, each sampling iteration becomes more efficient. Once the original planner is done with sampling the RRT and computing the optimal local path, instead of executing the first segment and then sampling the tree once again in a RH fashion, our method looks one more step ahead by also selecting the second waypoint in the best RRT branch. As will be shown in Section 4, this choice of control points has both pros and cons, namely the exploration of more space at the expenses of the overall traveled distance. Then, the current pose of the UAV and the two connected waypoints become the initial control points used to initialize the B-Spline, as shown in a simplified drawing in Figure 3.

B-Splines are piecewise polynomials uniquely defined by their degree d, a set of N control points $[\mathbf{P}_1, \mathbf{P}_2, \ldots, \mathbf{P}_N]$ and a knot vector $[k_0, k_1, \ldots, k_M]$, where $\mathbf{P}_i \in \mathbb{R}^3$, $k_i \in \mathbb{R}$ and $M = N + d$. A B-Spline trajectory is parameterized by time t, where $t \in [t_d, t_{M-d}]$. In a uniform B-Spline, each knot span $\Delta k_i = t_{i+1} - t_i$ has the same Δt. By normalizing $t \in [t_j, t_{j+1})$, $t_j < t_{j+1}$ to $u(t) = \frac{t-t_j}{t_{j+1}-t_j}$, the matrix representation of the B-

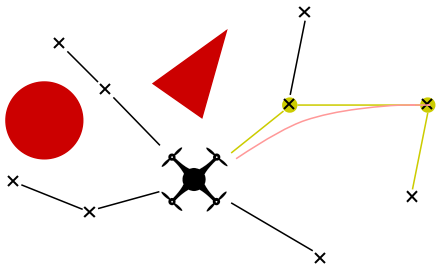

Figure 3. A simplified 2D representation of how the local B-Splines are planned. The black crosses and the related links are the sampled RRT. As per the GLocal methodology, these points are only sampled in the free visible space, so there will never be a point behind the obstacles (the red shapes). The highlighted path is the one with the highest cumulative gain, and the highlighted waypoints are the one used to plan the B-Spline, the pink curve.

Spline can be used to evaluate the position at time t:

$$\mathbf{p}(u(t)) = \begin{bmatrix} 1 & u(t) & u^2(t) \ldots u^d(t) \end{bmatrix} \mathbf{M}^{d+1} \begin{bmatrix} \mathbf{P}_{j-d} \\ \mathbf{P}_{j-d+1} \\ \mathbf{P}_{j-d+2} \\ \ldots \\ \mathbf{P}_j \end{bmatrix} \quad (1)$$

As shown in (Kaihuai Qin, 1998), for a cubic B-Spline with d = 3, M^{d+1} is:

$$\mathbf{M}^4 = \frac{1}{3!} \begin{bmatrix} 1 & 4 & 1 & 0 \\ -3 & 0 & 3 & 0 \\ 3 & -6 & 3 & 0 \\ -1 & 3 & -3 & 1 \end{bmatrix} \quad (2)$$

The two main properties of B-Splines relevant to path planning are the *continuity* and the *convex hull*.

- *Continuity*: the continuity at the knots is at most C^{d-2}, where d represents the degree of the B-Spline. So, a cubic B-Spline is C^2 continuous at the junction points between the polynomials of order *d-1* that form the spline itself
- *Convex hull*: a B-Spline curve is fully contained in the convex hull of its control points, *i.e.* the smallest convex polygon containing all the points. A representation of this property is shown in Figure 4

If uniform B-Splines have the same knot spans Δt_j throughout the entire curve, non-uniform B-Splines allow for different Δt_js.

Once the set of control points has been selected, a smooth path that passes through them has to be planned. For this purpose, B-Splines are an ideal tool. To do so, the selected subset of control points $[\mathbf{P}_d, \mathbf{P}_{d+1}, \ldots, \mathbf{P}_{N-d}]$ and the knot spans are optimized based on factors such as smoothness of the path, velocity and acceleration constraints and fit to the original waypoints as described by the FUEL paper (Zhou et al., 2021). A cost function that encapsulates all the desired characteristics of the final trajectory is setup and used in the optimization problem. The C^2 continuity property of the cubic B-Spline results in smooth paths both in velocity and in acceleration,

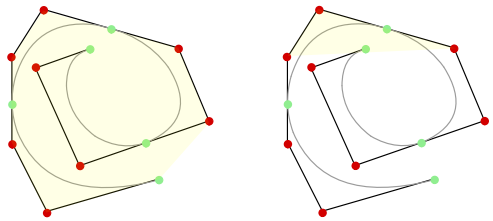

Figure 4. Representation of the convex hull property of a B-Spline. The red dots are the control points, the green ones are the knots, the gray curve is the spline and the yellow areas are the convex hulls of the control points. The image on the right shows that the property is also valid for a sub-set of control points.

a desired property that avoids jagged and sudden movements of the drone. Finally, to make sure that the resulting spline is also collision-free, the convex hull property is leveraged: if every convex hull of the B-Spline is collision-free, then the B-Spline is collision-free as well. The original FUEL approach explicitly incorporates an obstacle distance element in the cost function, while our method omits it, as the GLocal sampling logic guarantees that waypoints lie solely within known free space during local trajectory planning. This process would not be possible without the *differential flatness* property of quadcopters, as shown in (Mellinger and Kumar, 2011). Differential flatness is the extension of the concept of controllability in linear dynamical systems to nonlinear ones. Thanks to this property, the initial trajectory that will then be optimized can be represented with three independent polynomial functions, p_x, p_y, p_z.

This same procedure for generating and optimizing the B-Spline is also used on the global paths computed by GLocal, this time using the A* waypoints as control points for the B-Spline. Once either the local or the global smooth path has been computed, the state machine that governs the overall system enters a dedicated state where the drone follows the spline while making sure that no collisions are about to happen. Slightly before concluding the execution of the path, the next RRT is sampled and the new spline is computed. By doing so, the UAV can keep flying in a continuous trajectory and stopping only when necessary. Furthermore, while following the longer global splines, it keeps expanding the RRT in the background to constantly check if it should stop following the path to pursue a newly discovered viewpoint with a high exploration value. This logic ensures that the drone is always reactive to new discoveries.

4. EXPERIMENTS AND RESULTS

Originally, the GLocal framework was based on a simulation built on AirSim [1] and Unreal Engine 4 [2]. Given the requirements of the overall project in which this paper has been developed, the code has been adapted to run on a custom simulation environment. To validate the developed methodology, a simulation based on the Software in the Loop (SITL) simulation provided by ArduPilot [3], the Gazebo 11 simulator [4], the Robot Operating System (ROS) [5] and the MAVROS package [6] has been implemented. To use these tools together for UAV simulation, a virtual environment in Gazebo that mimics the physical environment in which the UAV will operate is created. In this paper, the chosen testing environments are two:

- The Amazon Web Services (AWS) small house [7] shown in Figure 5
- The AWS hospital environment [8] shown in Figure 5

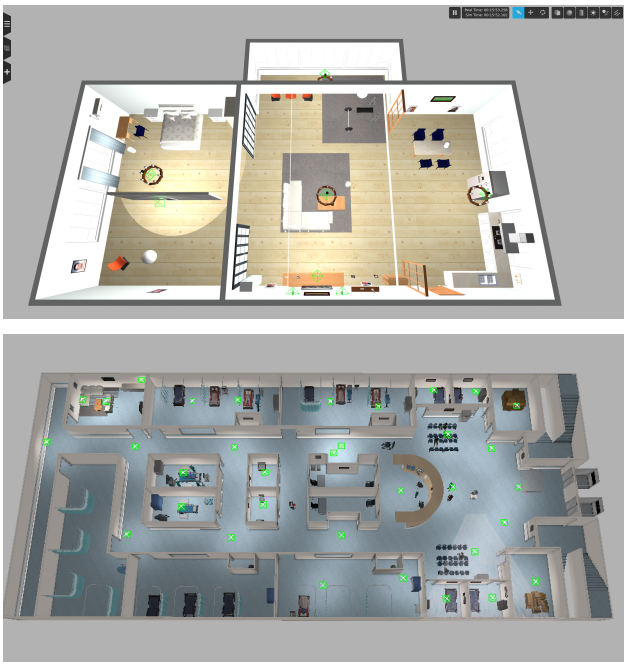

Figure 5. The house and the hospital simulated worlds used for testing the exploration algorithm.

The algorithm was developed for a mission that may require exploration of an indoor environment, and therefore the simulated worlds have been selected as suitable options to represent such an environment. Furthermore, the latter has been chosen because of its long corridors and its cluttered environment to test the long range exploration performance of the algorithm.
Once the world has been created, the ArduPilot SITL is used to simulate the UAV's firmware and control system within the Gazebo environment. The chosen UAV is a realistic replica of the 3DR IRIS Quadcopter [9]. MAVROS facilitates communication between ROS and simulated UAVs, enabling the development and testing of algorithms using the ROS platform.

First, the house environment is evaluated. The results are shown in Figure 6. Three different metrics have been chosen to compare the performances of the two algorithms: observed volume $[m^3]$, distance traveled over time $[m]$ and CPU usage. Their relevance can be easily motivated, given that the volume observed over time is the primary objective of the exploration process,

[1] https://microsoft.github.io/AirSim/
[2] https://www.unrealengine.com/en-US
[3] https://ardupilot.org/dev/docs/sitl-simulator-software-in-the-loop.html.
[4] https://staging.gazebosim.org/home
[5] https://www.ros.org/
[6] http://wiki.ros.org/mavros
[7] https://github.com/aws-robotics/aws-robomaker-small-house-world
[8] https://github.com/aws-robotics/aws-robomaker-hospital-world
[9] https://www.arducopter.co.uk/iris-quadcopter-uav.html

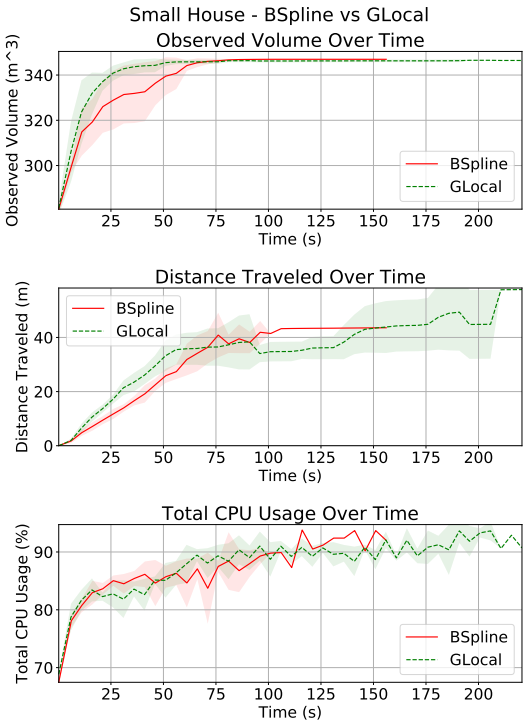

Figure 6. Comparative results of the two algorithms running in the small house environment. The mean and the standard deviation of each metric are plotted. GLocal represents the performance of the original implementation, whereas BSpline is our approach. Each method is interrupted when there are no more frontiers left to explore. The results were averaged over 5 experiments.

minimizing the traveled distance is a desirable outcome in terms of efficiency, and CPU utilization is crucial for real-time execution of the algorithm. In the small environment of the house, it can be seen how the original implementation of GLocal initially outperforms our method with respect to the amount of observed volume as it converges to a higher amount more quickly. However, it's worth noticing how the B-Spline algorithm concludes the exploration approximately 50 seconds earlier. This result is explainable by the smoothness and continuity of the generated paths. Furthermore, the overall traveled distance is slightly lower in our approach, a result that can be seen as a consequence of the shorter exploration time. Lastly, The CPU usage is comparable between the two algorithms, a significant result given the added complexity of the nonlinear optimization problem which generates the B-Splines at every iteration. More simulations were run in the hospital environment, and the results are shown in Figure 7. Given that fully exploring this much larger environment would take significantly more time, a different approach has been taken with these results; rather than waiting for the environment to be fully explored, a fixed time window has been chosen, and the metrics have been assessed over this period. The results show a marginal improvement in the observed volume of approximately 100 m^3, which is not much but still quite important in the context of autonomous exploration. On the other hand, this extra explored space is counterbalanced by a higher traveled distance. These outcomes can be interpreted by looking at the simulations: in this bigger and more complex environment, the longer planning horizon of the B-Spline algorithm was often led to explore nooks

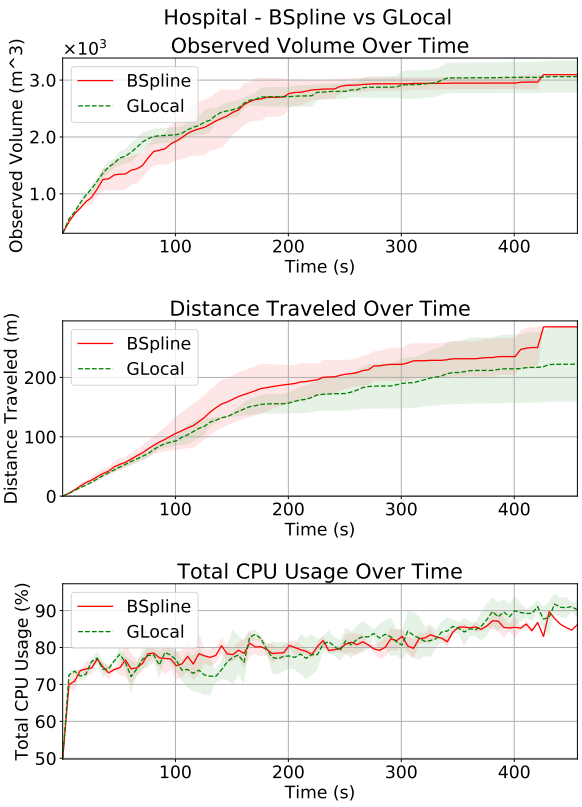

Figure 7. Comparative results in the hospital world over a fixed time window. The results were averaged over 3 experiments.

and crannies that the GLocal algorithm simply observed by exploring their immediate surroundings and immediately moving on. This behaviour had a smaller impact in the previous experiments in the house given the smaller overall volume. Lastly, it's worth noticing how the CPU usage of the two algorithms is still comparable, shining a light on the real-time capabilities of this approach.

The real strength of the B-Spline method is represented by the smoother trajectories that it produces. As previously stated, a third degree spline is C^2 continuous, which leads to a continuous linear acceleration profile.

Table 1. Comparison of mean, standard deviation and absolute maximum value of the three dimensional accelerations of the UAV using the two different algorithms.

	Acceleration $[ms^{-2}]$					
	GLocal			B-Spline		
	x	y	z	x	y	z
Mean	0.054	0.015	0.049	0.033	0.035	0.012
SD	0.841	0.978	0.725	0.401	0.390	0.275
|Max|	2.988	3.422	4.801	3.011	2.783	1.757

As shown in Table 1, the standard deviation of the acceleration is at least 47% lower while following the B-Spline trajectories. This results shows how the GLocal approach produces more jagged movements given by the start and stop that happens whenever a waypoint is reached. Smoother linear acceleration trajectories are desirable for many reasons, one of them

being the quality of the pose estimation of the robot in Active SLAM (A-SLAM) frameworks. Continuous movements are required to enhance the estimation quality, as sudden changes in trajectory might confuse the underlying motion model used to propagate the pose estimation. Moreover, the enhancement of pose estimation quality has a positive impact on the overall performance of global mapping consistency. Furthermore, it can be argued that more conservative acceleration profiles lead to an increased power efficiency. This assumption can be done by looking at Equation 3, which relates the required thrust to the linear acceleration, as shown in (Kumar et al., 2020).

$$\begin{bmatrix} \ddot{x} \\ \ddot{y} \\ \ddot{z} \end{bmatrix} = q \otimes \frac{1}{m} \begin{bmatrix} F_2 s\theta_2 + F_4 s\theta_4 \\ -F_1 s\theta_1 - F_3 s\theta_3 \\ F_1 c\theta_1 + F_2 c\theta_2 + F_3 c\theta_3 + F_4 c\theta_4 \end{bmatrix} \otimes q^* - \begin{bmatrix} 0 \\ 0 \\ g \end{bmatrix} \tag{3}$$

In the equation, $\ddot{x}, \ddot{y}, \ddot{z}$ represent the linear accelerations along the respective axes, q the quaternion that rotates the acceleration array from the body frame to the inertial frame of reference, s and c are respectively $sine$ and $cosine$, θ_i are the tilt angles of the rotors, and F_i are the propeller thrust forces. Finally, m and g are the mass of the drone and the gravity acceleration. It's easy to see that stronger accelerations require higher thrust from the rotors, which is one of the main culprits when it comes to battery drain.

5. CONCLUSION AND FUTURE WORK

This paper has explored the implementation of a planning algorithm based on optimized non-uniform B-Splines in an autonomous exploration framework. At every iteration, a non-linear optimization problem that computes the optimal, with respect to factors such as curve smoothness and dynamical feasibility, control points and knot spans of a B-Spline curve that explores the next best view waypoints sampled by the underlying exploration algorithm. This approach has been shown to slightly improve on the baseline algorithm GLocal in small scenarios such a small apartment by reducing the exploration time and the overall traveled distance while maintaining comparable performances with respect to CPU usage. When exploring bigger and more complex environments with many obstacles and different rooms, our methodology produces similar performance in terms of the selected metrics while marginally increasing the explored volume. A significant improvement over the baseline technique are the C^2 continuous trajectories, which could lead to enhanced performances with respect to continuous pose estimation in A-SLAM frameworks and battery management. Future work on this method includes the development of a different strategy to select the initial control points that define the spline and the implementation of a more continuous velocity profile over the entire mission. Furthermore, this method will be included in a bigger A-SLAM framework to enable complete autonomous exploration of unknown environments. In this A-SLAM context, the assumption that was made on the improvement of mapping accuracy will be tested.

6. ACKNOWLEDGMENTS

Funded by the European Union

REFERENCES

Duberg, D., Jensfelt, P., 2022. UFOExplorer: Fast and Scalable Sampling-Based Exploration With a Graph-Based Planning Structure. *IEEE Robotics and Automation Letters*, 7(2), 2487–2494.

Kaihuai Qin, 1998. General matrix representations for B-splines. *Proceedings Pacific Graphics '98. Sixth Pacific Conference on Computer Graphics and Applications (Cat. No.98EX208)*, IEEE Comput. Soc, Singapore, 37–43.

Karam, S., Nex, F., Chidura, B. T., Kerle, N., 2022. Microdrone-Based Indoor Mapping with Graph SLAM. *Drones*, 6(11), 352.

Karaman, S., Frazzoli, E., 2011. Sampling-based algorithms for optimal motion planning. *The international journal of robotics research*, 30(7), 846–894.

Kuipers, B., Byun, Y.-T., 1991. A Robot Exploration and Mapping Strategy Based on a Semantic Hierarchy of Spatial Representations. *Robotics and Autonomous Systems*, 8(1), 47–63.

Kulathunga, G., Klimchik, A., 2022. Optimization-based Motion Planning for Multirotor Aerial Vehicles: a Review. *arXiv preprint arXiv:2208.14647*.

Kumar, R., Bhargavapuri, M., Deshpande, A. M., Sridhar, S., Cohen, K., Kumar, M., 2020. Quaternion feedback based autonomous control of a quadcopter uav with thrust vectoring rotors. *2020 American Control Conference (ACC)*, 3828–3833.

Likhachev, M., Gordon, G., Thrun, S., 2003. Ara*: Anytime a* with provable bounds on sub-optimality. *Proceedings of (NeurIPS) Neural Information Processing Systems*, 767 – 774.

Mellinger, D., Kumar, V., 2011. Minimum snap trajectory generation and control for quadrotors. *2011 IEEE International Conference on Robotics and Automation*, 2520–2525.

Park, Y., Kim, W., Moon, H., 2021. Time-Continuous Real-Time Trajectory Generation for Safe Autonomous Flight of a Quadrotor in Unknown Environment. *Applied Sciences*, 11(7), 3238.

Ratliff, N., Zucker, M., Bagnell, J. A., Srinivasa, S., 2009. CHOMP: Gradient optimization techniques for efficient motion planning. *2009 IEEE International Conference on Robotics and Automation*, IEEE, Kobe, 489–494.

Rottmann, N., Denz, R., Bruder, R., Rueckert, E., 2021. A Probabilistic Approach for Complete Coverage Path Planning with low-cost Systems. *2021 European Conference on Mobile Robots (ECMR)*, IEEE, Bonn, Germany, 1–8.

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

Schmid, L., Reijgwart, V., Ott, L., Nieto, J., Siegwart, R., Cadena, C., 2021. A Unified Approach for Autonomous Volumetric Exploration of Large Scale Environments under Severe Odometry Drift. *IEEE Robotics and Automation Letters*, 6(3), 4504–4511.

Shen, Z., Agrawal, P., Wilson, J. P., Harvey, R., Gupta, S., 2021. Cppnet: A coverage path planning network. *OCEANS 2021: San Diego–Porto*, IEEE, 1–5.

Sun, H., Zhang, W., Yu, R., Zhang, Y., 2021. Motion Planning for Mobile Robots—Focusing on Deep Reinforcement Learning: A Systematic Review. *IEEE Access*, 9, 69061-69081.

Usenko, V., von Stumberg, L., Pangercic, A., Cremers, D., 2017. Real-Time Trajectory Replanning for MAVs using Uniform B-splines and a 3D Circular Buffer. *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 215–222.

Witting, C., Fehr, M., Bähnemann, R., Oleynikova, H., Siegwart, R., 2018. History-Aware Autonomous Exploration in Confined Environments Using MAVs. *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 1–9.

Zhang, Y., Zhou, B., Wang, L., Shen, S., 2022. Exploration with global consistency using real-time re-integration and active loop closure. *2022 International Conference on Robotics and Automation (ICRA)*, IEEE, 9682–9688.

Zhou, B., Gao, F., Wang, L., Liu, C., Shen, S., 2019. Robust and efficient quadrotor trajectory generation for fast autonomous flight. *IEEE Robotics and Automation Letters*, 4(4), 3529–3536.

Zhou, B., Zhang, Y., Chen, X., Shen, S., 2021. Fuel: Fast uav exploration using incremental frontier structure and hierarchical planning. *IEEE Robotics and Automation Letters*, 6(2), 779–786.