2024 IEEE International Conference on Robotics and Automation (ICRA 2024)
May 13-17, 2024. Yokohama, Japan

Autonomous 3D Exploration in Large-Scale Environments with Dynamic Obstacles

Emil Wiman†, Ludvig Widén, Mattias Tiger and Fredrik Heintz

***Abstract*— Exploration in dynamic and uncertain real-world environments is an open problem in robotics and it constitutes a foundational capability of autonomous systems operating in most of the real-world. While 3D exploration planning has been extensively studied, the environments are assumed static or only reactive collision avoidance is carried out. We propose a novel approach to not only avoid dynamic obstacles but also include them in the plan itself, to deliberately exploit the dynamic environment in the agent's favor. The proposed planner, Dynamic Autonomous Exploration Planner (DAEP), extends AEP to explicitly plan with respect to dynamic obstacles. Furthermore, addressing prior errors within AEP in DAEP has resulted in enhanced exploration within static environments. To thoroughly evaluate exploration planners in such settings we propose a new enhanced benchmark suite with several dynamic environments, including large-scale outdoor environments. DAEP outperforms state-of-the-art planners in dynamic and large-scale environments and is shown to be more effective at both exploration and collision avoidance.**

Fig. 1: Real-world 3D environments can be non-trivial to explore in detail due to large scales, geometrical complexity and the presence of dynamic obstacles wandering about.

I. Introduction

Real-world environments change over time. Be it due to construction, renovation, refurbishment, object relocation or deterioration. For robots to function effectively in the real-world they must possess the ability to explore their surroundings to build or maintain a 3D world model. Exploration is consequently a foundational capability as it enables the agent to navigate an a priori unknown environment effectively and enable the gathering of valuable information about the environment for any number of tasks.

Deliberate exploration is an open problem in robotics. The 3D exploration planning problem is to autonomously explore a potentially large and complex environment as quickly as possible, such that it is covered with a sensor configuration until desired coverage. The static environment case has been well-studied for applications such as volumetric exploration [1], surface inspection [2], object search [3], infrastructure modeling [4], weed classification [5] and 3D reconstruction [6] among others. However, most everyday environments of the real-world are occupied by people, pets, vehicles and other autonomous agents: The environments are dynamic, not static. Existing techniques do not take into account the presence of dynamic obstacles beyond simple obstacle avoidance behavior [7][8]. Even though it can be possible to force a region to be void of dynamic obstacles, it is often inconvenient and time-consuming. For instance, imagine trying to explore a busy city center like Times Square in New York. The process of removing dynamic obstacles from such a space is both laborious, costly and would cause major annoyance. Furthermore, clearing an area proves especially difficult if the scenario at hand is grand, as in Fig. 1. Moreover, these environments might need repeated exploration (changing world) which makes it even more inconvenient to clear the area. It would be far better to be able to effectively explore such environments in the presence of dynamic obstacles. Not to mention if time is of the essence.

We consider the problem of autonomous 3D exploration planning in the large-scale setting (Fig. 1) with the presence of dynamic obstacles. Both to avoid dynamic obstacles for safety reasons and to make the exploration more effective by exploiting how the environment changes. The contributions of this work are:

- An improved benchmark[1] for evaluating exploration planners in environments with dynamic obstacles. It comprises ten maps with varying sizes and complex geometries, reflecting challenging real-world environments. Docker [9] is used to provide high reproducibility and compatibility.
- A comprehensive evaluation of existing exploration planners utilizing the proposed benchmark. The planners are examined in both a static and dynamic setting to investigate their different strengths and weaknesses.

[1]https://github.com/LudvigWiden/daeplanner

This work was partially supported by the Wallenberg AI, Autonomous Systems and Software Program (WASP) funded by the Knut and Alice Wallenberg Foundation. Portions of this work were carried out using the AIOps/Stellar facilities funded by the Excellence Center at Linköping–Lund in Information Technology (ELLIIT). Thanks to WARA-PS for the Village Environment.

E. Wiman, L. Widén, M. Tiger and F. Heintz are with the Department of Computer Science, Linköping University, Sweden †Corresponding author: emil.wiman@liu.se

- A proposed planner (DAEP) that demonstrates both superior effectiveness and safety over state-of-the-art.

The paper's organization is as follows: In II, we introduce related research to contextualize the paper's contribution. The dynamic 3D exploration problem is defined in III. The presentation of the proposed method, DAEP, is in IV. The evaluation of this approach is detailed in V. Finally, conclusions in VI.

II. Related Work

Autonomous 3D exploration has been under active study for over two decades [10], with frontier exploration [10] as one of the first approaches to tackle 3D exploration. A frontier can formally be defined as the boundary between explored and unexplored regions of the environment. Frontier exploration is well established [11][8] but poses challenges as how to explore a local region effectively and how to acquire information gain when traveling between regions.

Next, work on the next-best-view (NBV) problem [12] from computer vision, enabled autonomous NBV exploration planning [13] which focuses directly on the sensor coverage problem. This made efficient local exploration with receding-horizon NBV planning (RH-NBVP) [1] possible. It works by combining NVB sampling with rapidly exploring random trees (RRT) [14] which produce traversable paths between the robot pose and candidate viewpoints. By executing only the first edge from the RRT and repeating the expansion process, the planner adapts to new information.

Autonomous exploration planner (AEP) [15] combines both paradigms where RH-NBVP is used as a local exploration strategy and frontier exploration for global planning. This combination has proved successful, especially in large-scale environments where RH-NBVP may suffer from premature termination. AEP utilizes Gaussian processes [16] to effectively estimate the potential information gain. The potential information gain translates to the unmapped volume that can be collected from a pose and it is an important principle we build upon in this work.

AEP's selection of interesting viewpoints does not necessarily consider the structure of interest when planning which may lead to inefficient potential information gain estimation. This is addressed by [17], which proposes a novel informed sampling-based approach that leverages surface frontiers to sample viewpoints only where high information gain is expected, leading to faster exploration. This approach has been shown to outperform AEP in realistic static exploration scenarios. However, the code is not available and it has consequently not been included in our evaluation.

The first limited steps towards autonomous exploration planning for dynamic obstacles are dynamic frontiers [8] and the dynamic exploration planner (DEP) [7]. The former extends 2D frontier exploration with a new type of frontier that represents one or several dynamic obstacles. These can for example be (detected) people who stand in front of a door opening. These dynamic frontiers will be visited later when people hopefully have moved, unlocking new unexplored regions.

DEP [7] instead builds a probabilistic roadmap (PRM) [18] incrementally, which is used for reactive collision avoidance by finding a path around dynamic obstacles when collisions are imminent. Consequently, obstacle collisions are potentially reduced, but obstacles have no other impact on the autonomous exploration itself. DEP denotes this ability to handle dynamic obstacles as a replanning [19] functionality.

III. Problem Statement

The problem to consider can be formalized as follows. Given a 3D volume $V \subset \mathbb{R}^3$, the objective of the agent is to explore this volume as completely as possible while avoiding collisions with dynamic obstacles. The volume V consists of two components, namely the free volume $V_{free}(t)$ and the occupied volume $V_{occupied}$. Here $V_{occupied}$ refers to the static environment where dynamic obstacles have been excluded. Note that the free volume is subject to temporal change, meaning that at time t the volume might be occupied by a dynamic obstacle. Initially, all poses $\mathbf{p} \in V \subset \mathbb{R}^3$ are unmapped. Thus the objective is to build an internal representation, M, that resembles $V_{occupied}$ as closely as possible by exploring the environment. Moreover, the agent must compute feasible routes that avoid the trajectory of the dynamic obstacles while simultaneously avoiding sub-optimal views in the environment to minimize the exploration time and path length. Due to the highly uncertain and dynamic setting of the environment, this must be solved online.

IV. Proposed Approach

We propose the Dynamic Autonomous Exploration Planner (DAEP), which builds upon AEP and introduces a crucial correction as well as several important modifications and improvements. See Fig. 2 for an overview of DAEP. The coming sections highlight the most important components of DAEP.

A. Introduction to AEP

A brief introduction will be given of AEP, to help the reader grasp the concepts of DAEP. Initially, will the local planner create an RRT-tree, where the static information gain is estimated in each pose utilizing the static score function [15]. Subsequently, the pose with the highest score will be chosen as goal node and the agent will travel to that pose by following the structure of the RRT-tree. If no local nodes with sufficient information gain can be found, then the global planner will take over. It will search for the closest frontier which has been cached earlier in the exploration process by expanding an RRT*-tree [20]. If no frontiers exist, then exploration is completed, otherwise the agent will travel to the closest frontier and continue the exploration process.

B. Predictor

To operate in a dynamic environment, a predictor component is needed to estimate the future trajectory of dynamic obstacles. Here, a Kalman filter [21] has been employed with a constant velocity motion model [22]. The state vector is

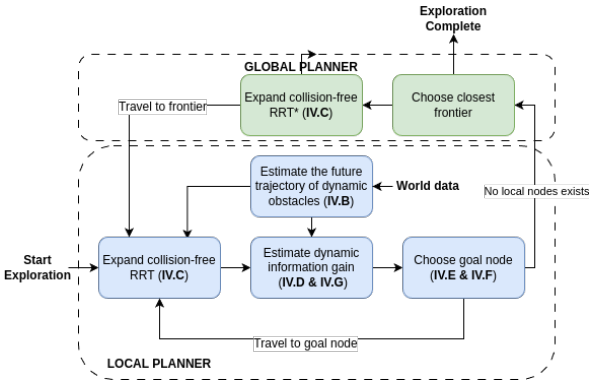


Fig. 2: Schematic overview of DAEP with references to Sec. (IV-B)-(IV-G) for functionality descriptions for each component.

given as $\begin{pmatrix} x & y & v_x & v_y \end{pmatrix}$, where x and y are the position of the dynamic obstacle at time point k and v_x and v_y are the velocities in the x and y direction at time point k. Measurements are supplied by the simulation environment regarding the current state of the dynamic obstacles. The Kalman filter provides a future distribution of the position of the dynamic obstacle, with a mean and bounded covariance [23][24] for each time point k. These can be utilized to handle the uncertainty in the dynamic environment and thus help construct collision-free paths.

C. Time-based RRTs

Including a predictor component enables the agent to construct paths that avoid the future trajectories of the dynamic obstacles. This is done by introducing time as a state in the RRT-tree construction, similarly as [25]. Each node is assigned a time of arrival, namely the time at which the agent is estimated to reach a certain node. By comparing the time of arrival with the future trajectory for each dynamic obstacle it can be determined whether or not the node is collision-free in the future. This technique is used in both the local and global planner.

D. Dynamic Information Gain

Due to the environment being dynamic, it is no longer guaranteed that the estimated potential information gain will be acquired upon arrival to a certain view. This is since dynamic obstacles may block the view upon arrival, hence decreasing the information gain acquired. To address these issues, a dynamic score function $s(\mathbf{p}, t)$ (Sec. IV-F) has been introduced. This function utilizes dynamic information gain $d(\mathbf{p}, t)$ to produce better decisions in the dynamic environment. Inspecting Fig. 3, the dynamic information gain is simply computed as the difference between the blue rays and the red rays (note that the red rays start within the gray square). This dynamic information gain can be estimated during the construction of the RRT and hence assigned to each node. Note that the blue rays blocked by the white circle will be included in the available information gain upon arrival at the origin of the blue rays.

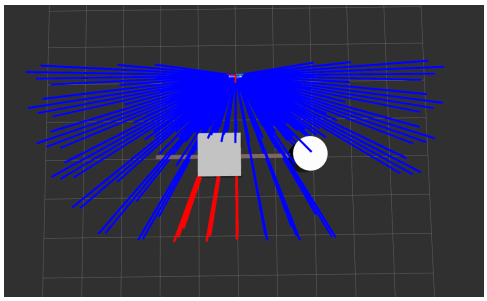


Fig. 3: Dynamic information gain visualized with one dynamic obstacle. The white circle represents the current position of the dynamic obstacle while the gray square represents its future bounding box according to the predictor component. Arriving at the point where the blue rays originate, the line of sight will be obstructed, resulting in the invisibility of the red rays as a consequence.

E. Dynamic Frequency Map

Dynamic obstacles tend to not navigate uniformly. Usually they follow designated paths or roads. This can be leveraged to enhance decision-making in a dynamic environment. By constructing a heat map of the environment and updating it with position information of dynamic obstacles from the simulation environment, a distribution of the historical position of dynamic obstacles can be obtained. Since they follow designated paths, regions will be formed in the heat map. This Dynamic Frequency Map, $DFM(\mathbf{p})$, can then be utilized to increase the priority for areas that have previously shown significant occupancy but are presently unoccupied according to the most recent prediction.

F. Dynamic Score Function

To aid the agent in the decision-making process, a new dynamic score function has been implemented that extends the score function of AEP (Sec. IV-A) with a temporal (Sec. IV-D) and a statistical (Sec. IV-E) component. The dynamic score in pose $\mathbf{p}$ is

$$s(\mathbf{p}, t) = d(\mathbf{p}, t) \cdot \underbrace{e^{-\lambda \cdot c(\mathbf{p})}}_{(0,1]} \cdot (1 + (\zeta \cdot \underbrace{DFM(\mathbf{p})}_{[0,1]})) \quad (1)$$

where the dynamic score $s(\mathbf{p}, t)$ for a specific pose $\mathbf{p}$ at time t is determined by the dynamic information gain $d(\mathbf{p}, t)$ scaled by the cost $c(\mathbf{p})$ associated with traveling to that pose. Also, the dynamic scores receive a potential boost $(1+(\zeta \cdot DFM(\mathbf{p}))$. Here, λ and ζ are tuning parameters. The dynamic score consists of scaling the dynamic information gain by distance, prioritizing closer poses with higher scores. If a pose is currently unoccupied but has been historically occupied, it will receive a boost in score which will re-direct the agent to explore this new priority region.

G. Yaw Angle Booster

Initial experiments revealed that AEP occasionally omits certain areas near the map boundary, resulting in significant

TABLE I: Experimental Parameters

Parameter	Value	Parameter	Value
Linear Velocity	0.5 [m/s]	Collision Box [m^3]	[0.4, 0.4, 0.1]
Angular Velocity	1 [rad/s]	Horizontal FoV	103.2 [deg]
Map Resolution	0.2 [m]	Vertical FoV	77.4 [deg]
RRT Ext. range	1 [m]	Camera Range	5 [m]
Dyn. Obs. Lin. Vel.	0.35 [m/s]	Dyn. Obs. Ang. Vel.	1 [rad/s]
λ	0.75	ζ	0.5
α	6		

gaps in the environment representation, also reported by [17]. It was found that AEP only accounts for volume inside the pre-defined bounding box, leading to negligent exploration near the borders. DAEP addresses this by artificially boosting information gain near borders using a constant multiplier α. This enhancement has improved the exploration greatly close to the borders over AEP.

V. Experimental Evaluation

To evaluate the performance of DAEP compared to other planners, especially for realistic scenarios with dynamic obstacles, a benchmark has been developed (V-A). DAEP and three competing planners (RH-NBVP [1], AEP [15], DEP [7]) are evaluated on the benchmark. The planners undergo initial evaluation in several static environments to assess in the classic sense, followed by assessment in dynamic environments. Finally, DEP and DAEP are evaluated on large-scale dynamic environments to see how they scale.

A. Benchmark

The benchmark[1] include ten dynamic scenarios (Tab. II) which can also be run with no dynamic obstacles. Six of the worlds are from [7] where we have added difficult dynamic obstacles (people walking) to the previous static worlds *Cafe*, *Maze* and *Apartment*, made them more difficult in *Field*, and kept them as-is in *Auditorium* and *Tunnel*. In the new scenarios *Crosswalks* has 4 people crossing back-and-forth, *Patrol* has eight people moving on patrol paths and *Exhibition* has people moving along the walls at a close poster-viewing distance. The large-scale scenario *Village* (Fig. 4) is a high-res scan of a cotton village with surrounding greenery, with people walking around. The benchmark code itself consists of a Docker solution which simplifies getting started and extending the benchmark in the future. It simplifies running all planners on a single machine, despite requirements on different versions of ROS [26] and other conflicting dependencies. Integration of the four planners RH-NBVP, AEP, DEP, and DAEP is provided. The scenarios are simulated with Gazebo 9 [27], using the same simulated quadcopter and Realsense D435 depth camera as in [7] with the camera parameters from Tab. I. The controller supplied by [7] has been employed in all planners, to avoid alterations of the motion planning. OctoMap [28] is used as a representation of the internal map. The following experiment procedure has been followed:

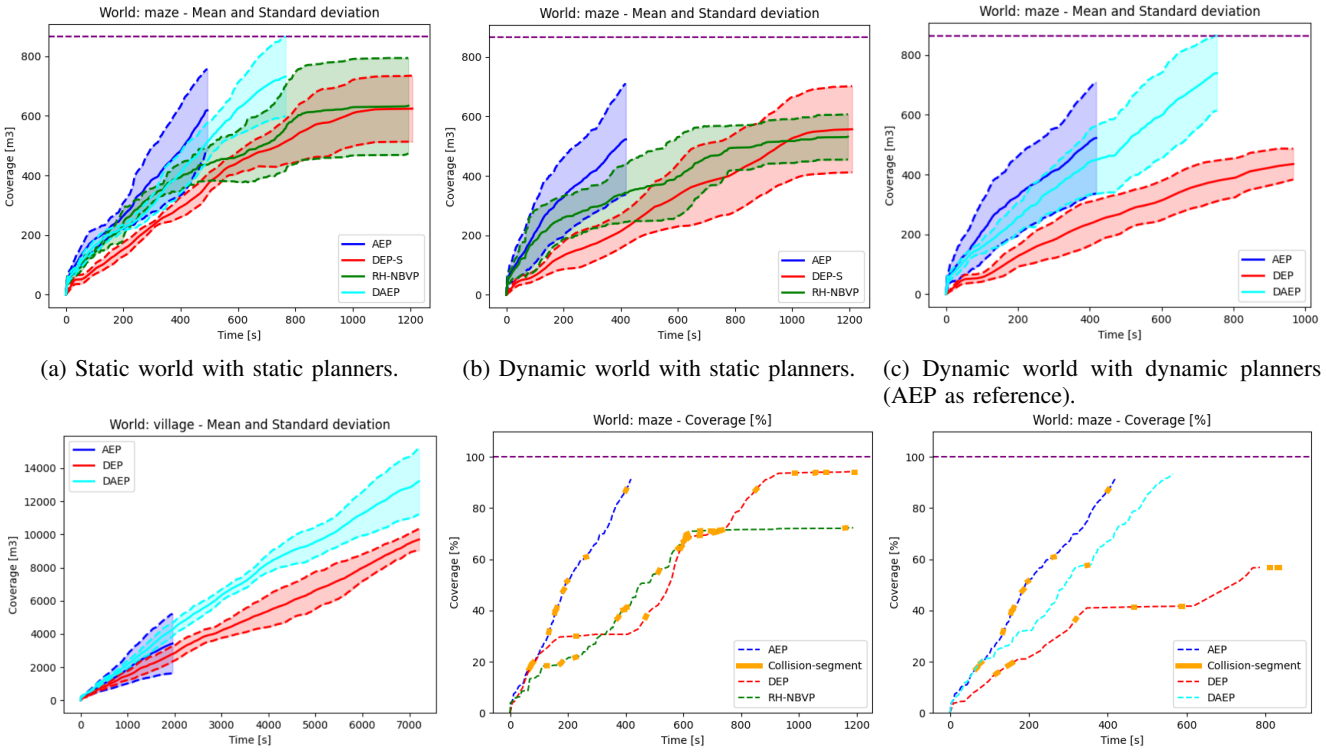
- For each experiment run: A planner, world and mode (with or without dynamic obstacles) is chosen.
- The agent starts in one of five different start locations in the specified world with zero yaw.
- The agent travels 1 meter vertically up in the air. A 360-degree rotation is performed to gain initial information about the environment and to ensure free space in the representation to start exploring from.
- The exploration algorithm starts and exploration begins.
- The exploration continues until the planner signals being finished, or the hard time limit (20 min) is reached.

During experiments the default parameters for each planner[1] have been used. The experiment parameters in Tab. I have been supplied by [7] except for λ, ζ and α which were manually tuned until sufficient behavior. The same parameters have been used in all experiments. Each experiment is repeated five times (i.e. five runs) and the results are reported as $\mu \pm \sigma$ over all specified scenarios' mean run. The different performance measures used are **C**: Coverage [%], **T**: Exploration Time [s], **PL**: Path Length [m], **PT**: Planning Time [s] and **NOC**: Number Of Collisions. Planning Time denotes the accumulated time it takes for the system to construct a new path. While planning, the agent is stationary. Exploration Time denotes the total time it takes for the agent to complete the exploration. Note that the abbreviation DEP refers to DEP with its re-plan functionality enabled while DEP-S denotes that it is disabled.

TABLE II: Worlds part of the benchmark.

World	Origin	Volume [m³]	No. Dyn. Obs.
Cafe	[7]	510	2
Maze	[7]	865	5
Apartment	[7]	1627	12
Tunnel	[7]	1100	2
Field	[7]	1440	8
Auditorium	[7]	798	3
Exhibition	NEW	450	22
Crosswalks	NEW	450	4
Patrol	NEW	800	8
Village	NEW	40057	15

Fig. 4: Top-down view of the *Village* area. The red bounding box illustrates the volume to be explored. The area is roughly 1 hectare and populated with multiple dynamic obstacles. The paths of the dynamic obstacles are highlighted in green where the lines indicate that the dynamic obstacle moves back and forth while a loop indicates that the dynamic obstacles circulate.



(a) Static world with static planners.

(b) Dynamic world with static planners.

(c) Dynamic world with dynamic planners (AEP as reference).

(d) Exploration in the *Village* environment with dynamic obstacles. AEP as reference.

(e) Collisions for static planners.

(f) Collisions for dynamic planners.

Fig. 5: Upper row: Exploration progress in *Maze*. The maximum volume is 865 m^3. Here (D)AEP halts the exploration where the envelope stops. Lower row: Exploration progress in the *Village* environment to the left and collision graphs in the middle and to the right for *Maze*. The collision graphs present the run that accumulated the most coverage. Orange segments indicate collisions and their duration.

B. Static Planners in a Static Environment

The planners are evaluated on static versions of the first six worlds (same as [7]) to compare their performance in the classical sense. The aggregated results are shown in Tab. IIIa and coverage over time is shown for *Maze* in Fig. 5a. DAEP is employed in this experiment to evaluate its performance after the correction from Sec. IV-G, even though it contains some overhead to handle the dynamic environment. Including it will hopefully show the impact of this fix in the static setting compared to AEP. DEP is also employed without its re-planning ability, thus it lacks any overhead. From Fig. 5a it can be observed that among the static planners, AEP manages to explore the environment quicker than both DEP-S and RH-NBVP while acquiring a similar amount of final volume. However, DAEP outperforms all planners in this scenario in terms of coverage. The observations regarding the static planners are reflected in Tab. IIIa, where AEP dominates in terms of exploration time, while DEP-S accumulates the largest volume on average. However, DAEP proves to find more coverage than its static competitors despite its overhead. It produces similar results in terms of path length compared to AEP while doubling the planning time due to the overhead. All planners face challenges achieving 100% coverage due to drone size restrictions and difficulties adjusting a bounding box to a world.

C. Static Planners in a Dynamic Environment

Next, we investigate how AEP, DEP-S, and RH-NBVP are impacted by the presence of dynamic obstacles. The first six worlds are now filled with dynamic obstacles. The aggregated results can be found in Tab. IIIb. Coverage over time is shown for *Maze* (Fig. 5b) as a representative example. Coverage variance increased for all planners (Tab. IIIb), possibly due to dynamic obstacles limiting sight and access to certain areas. Collisions increased (Fig. 5e), which would be disastrous in real-world scenarios. The findings in Fig. 5e are reinforced by Tab. IIIb to occur in general. All planners find less coverage compared to Tab. IIIa. Similarly for exploration time and planning time, except for the exploration and planning time of RH-NBVP. Finally, each planner collides at least four times on average for each run.

D. Dynamic Planners in a Dynamic Environment

Introducing the dynamic planners in the dynamic environment should address the issues presented in Sec. V-C. Here, DEP and DAEP are employed in the dynamic environment, with AEP as a reference. The experiments have been conducted in the first six worlds, as well as in *Exhibition*, *Crosswalks* and *Patrol*. The aggregated results can be found in Tab. IIIc. A representative example is shown in Fig. 5c with associated collision rate (Fig. 5f) for *Maze*. From Fig.

TABLE III: Results from Different Scenarios and Planners

	AEP (**DAEP**)	DEP-S	RH-NBVP
C	84.05 ± 9.53 (**86.37** ± 10.6)	85.84 ± 11.28	80.55 ± 12.92
T	**541.14** ± 401.68 (655.13 ± 349.25)	1017.35± 248.02	1128.76± 168.18
PL	143.76 ± 110.25 (**143.59** ± 73.33)	254.81 ± 70.53	211.25 ± 37.6
PT	55.64 ± 51.02 (119.2 ± 57.94)	28.33 ± 18.39	**1.83** ± 0.33

(a) Results from static environments. Results from DAEP is in parentheses.

	AEP	DEP-S	RH-NBVP
C	80.29 ± 15.43	**84.63** ± 13	77.79 ± 13.25
T	**460.68**± 373.53	953.91± 253.45	1158.47± 99.91
PL	**111.9** ± 77.05	223.25± 52.8	204.82 ± 30.34
PT	29.95 ± 24.21	25.8 ± 15.14	**2.04** ± 1.02
NOC	**4.07** ± 3.83	6.47 ± 5.36	6.63 ± 4.59

(b) Results from dynamic environments with static planners.

	AEP (Reference)	DEP	**DAEP**
C	83.56 ± 13.58	81.85 ± 18.37	**91.25** ± 6.25
T	368.78 ± 335.05	775.03 ± 301.81	**589.59** ± 346.44
PL	91.7 ± 70.39	170.96 ± 65.15	**136.43** ± 66.37
PT	26.11 ± 21.33	**53.7** ± 39.56	79.79 ± 40.29
NOC	3.36 ± 3.41	6.38 ± 4.38	**0.31** ± 0.63

(c) Results from dynamic environments with dynamic planners.

	AEP (Reference)	DEP	**DAEP**
C	8.56 ± 4.49	24.22 ± 1.66	**32.96** ± 4.98
T	1094.6± 531.83	**7175.78**± 17.73	7201.08± 2.27
PL	213.03 ± 104.36	**893.66** ± 27.65	1280.65± 22.49
PT	57.08 ± 30.26	2053.47± 163.9	**930.01** ± 155.52
NOC	0 ± 0	1.4 ± 1.0198	**0** ± 0

(d) Results from large-scale dynamic environment *Village*.

	-	DEP	**DAEP**
C	-	60.93	**68.05**
T	-	35880.8	**31241.5**
PL	-	**2978.4**	5132.73
PT	-	20068.1	**5763.29**
NOC	-	25	**0**

(e) *Village* over 10 hours in a dynamic setting with DAEP and DEP.

5c it is prominent that DAEP explores the environment faster and considerably more meticulously than DEP. Noticeably, it also explores for a longer period than AEP and thus, manages to gather more volume. Furthermore, AEP and DEP continue to collide frequently, while DAEP collides rarely (e.g. only once in Fig 5f). The results provided in Tab. IIIc demonstrate that DAEP manages to accumulate more coverage than DEP and AEP on average. Additionally, it does so with a reduced average exploration time and path length compared to DEP. Finally, the number of collisions has decreased significantly for DAEP, compared to DEP and AEP.

E. Large-Scale Environments

Finally, we investigate how the planners scale to realistic large-scale outdoor scenarios. Here, we use *Village*, see Fig. 4. The collected findings for the 2-hour experiment are shown in Tab. IIId and the exploration progress is depicted in Fig. 5d. Interestingly, it can be observed, from Fig. 5d, that AEP halts the exploration after only 2000 seconds. Correspondingly, this can be noticed in Tab. IIId where AEP collects significantly less coverage. It was found that AEP was restricted in the sampling process of new coordinates, where coordinates in a certain interval could never be sampled, thus leading to absent exploration. DAEP resolves this by making the interval dynamic so it adapts to the size of the current world. Moreover, DAEP manages to find a larger amount of volume on average compared to DEP, while completely avoiding collisions. After the 2-hour experiment only about 32% of the *Village* environment was mapped by DAEP. Hence, DAEP and DEP were allowed to continue to explore for a total of 10 hours to push their limits (this experiment was only conducted once). The results can be found in Tab. IIIe and the corresponding representation of the world is depicted in Fig. 6. Comparing the real world in Fig. 4 with the representation in Fig. 6, DAEP is observed to have managed to capture the essential structures and details of the environment. Examining Tab. IIIe, it shows that roughly 68% of the environment has been mapped after 10 hours while avoiding collisions completely. DAEP outperforms DEP both in terms of coverage but also in number of collisions. Also, observe here that DAEP only plans roughly 18% of the total exploration time while DEP plans 56% of the total exploration time.

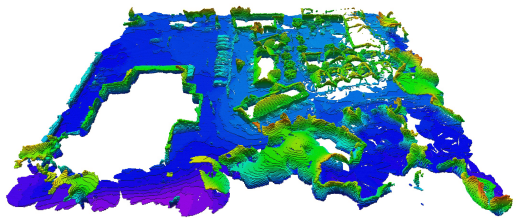


Fig. 6: Constructed representation of the *Village* environment from DAEP.

VI. Summary & Conclusion

We propose DAEP, a novel approach for autonomous 3D exploration with dynamic obstacles. DAEP is an extension of AEP with several improvements to handle the presence of dynamic obstacles. A predictor component has been added to facilitate the construction of time-based RRTs, used for safe and collision-free path planning. Furthermore, a novel dynamic score function has been proposed to facilitate efficient navigation in a dynamic environment. Here, the dynamic information gain has been used to predict the potential information gain upon arrival to a new view, while the DFM score has been used to increase priority to areas that are frequently populated by obstacles. DAEP outperform both static and dynamic competitors in the experiments. It demonstrates the ability to explore large-scale environments effectively and safely compared to previous dynamic planners. In future work, we propose to combine the exploration planner with a safe motion planner [29][30] for real-world testing in *Village*.

REFERENCES

[1] A. Bircher, M. Kamel, K. Alexis, H. Oleynikova, and R. Siegwart, "Receding horizon "next-best-view" planner for 3d exploration," in *2016 IEEE International Conference on Robotics and Automation (ICRA)*, 2016, pp. 1462–1468.

[2] A. Bircher, M. S. Kamel, K. Alexis, H. Oleynikova, and R. Siegwart, "Receding horizon path planning for 3d exploration and surface inspection," *Autonomous Robots*, vol. 42, 02 2018.

[3] T. Dang, C. Papachristos, and K. Alexis, "Autonomous exploration and simultaneous object search using aerial robots," in *2018 IEEE Aerospace Conference*, 2018, pp. 1–7.

[4] L. Yoder and S. A. Scherer, "Autonomous exploration for infrastructure modeling with a micro aerial vehicle," in *International Symposium on Field and Service Robotics*, 2015.

[5] M. Popović, G. Hitz, J. Nieto, I. Sa, R. Siegwart, and E. Galceran, "Online informative path planning for active classification using uavs," in *2017 IEEE International Conference on Robotics and Automation (ICRA)*, 2017, pp. 5753–5758.

[6] S. Song and S. Jo, "Surface-based exploration for autonomous 3d modeling," in *2018 IEEE International Conference on Robotics and Automation (ICRA)*, 2018, pp. 4319–4326.

[7] Z. Xu, D. Deng, and K. Shimada, "Autonomous uav exploration of dynamic environments via incremental sampling and probabilistic roadmap," *IEEE Robotics and Automation Letters*, vol. 6, no. 2, pp. 2729–2736, 2021.

[8] V. Cavinato, T. Eppenberger, D. Youakim, R. Siegwart, and R. Dubé, "Dynamic-aware autonomous exploration in populated environments," in *2021 IEEE International Conference on Robotics and Automation (ICRA)*, 2021, pp. 1312–1318.

[9] D. Merkel, "Docker: lightweight linux containers for consistent development and deployment," *Linux journal*, vol. 2014, no. 239, p. 2, 2014.

[10] B. Yamauchi, "A frontier-based approach for autonomous exploration," in *Proceedings 1997 IEEE International Symposium on Computational Intelligence in Robotics and Automation CIRA'97. 'Towards New Computational Principles for Robotics and Automation'*, 1997, pp. 146–151.

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

[12] C. Connolly, "The determination of next best views," in *IEEE International Conference on Robotics and Automation (ICRA)*, vol. 2, Mar 1985, pp. 432–435.

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

[14] S. M. Lavalle, "Rapidly-exploring random trees: A new tool for path planning," *TR 98-11, Computer Science Dept., Iowa State University*, 1998.

[15] M. Selin, M. Tiger, D. Duberg, F. Heintz, and P. Jensfelt, "Efficient autonomous exploration planning of large-scale 3-d environments," *IEEE Robotics and Automation Letters*, vol. 4, no. 2, pp. 1699–1706, 2019.

[16] C. Williams and C. Rasmussen, "Gaussian processes for regression," *Advances in neural information processing systems*, vol. 8, 1995.

[17] Y. Kompis, L. Bartolomei, R. Mascaro, L. Teixeira, and M. Chli, "Informed sampling exploration path planner for 3d reconstruction of large scenes," *IEEE Robotics and Automation Letters*, vol. 6, no. 4, pp. 7893–7900, 2021.

[18] L. Kavraki, P. Svestka, J.-C. Latombe, and M. Overmars, "Probabilistic roadmaps for path planning in high-dimensional configuration spaces," *IEEE Transactions on Robotics and Automation*, vol. 12, no. 4, pp. 566–580, 1996.

[19] M. Wzorek, J. Kvarnström, and P. Doherty, "Choosing path replanning strategies for unmanned aircraft systems," in *Proceedings of the International Conference on Automated Planning and Scheduling*, vol. 20, 2010, pp. 193–200.

[20] S. Karaman and E. Frazzoli, "Sampling-based algorithms for optimal motion planning," *International Journal of Robotic Research - IJRR*, vol. 30, pp. 846–894, 06 2011.

[21] R. E. Kalman, "A new approach to linear filtering and prediction problems," *Transactions of the ASME–Journal of Basic Engineering*, vol. 82, no. Series D, pp. 35–45, 1960.

[22] X. Rong Li and V. Jilkov, "Survey of maneuvering target tracking. part i. dynamic models," *IEEE Transactions on Aerospace and Electronic Systems*, vol. 39, no. 4, pp. 1333–1364, 2003.

[23] O. Andersson, M. Wzorek, and P. Doherty, "Deep learning quadcopter control via risk-aware active learning," *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 31, no. 1, Feb. 2017. [Online]. Available: https://ojs.aaai.org/index.php/AAAI/article/view/11041

[24] O. Andersson, M. Wzorek, P. Rudol, and P. Doherty, "Model-predictive control with stochastic collision avoidance using bayesian policy optimization," in *2016 IEEE International Conference on Robotics and Automation (ICRA)*, 2016, pp. 4597–4604.

[25] A. Sintov and A. Shapiro, "Time-based rrt algorithm for rendezvous planning of two dynamic systems," in *2014 IEEE International Conference on Robotics and Automation (ICRA)*, 2014, pp. 6745–6750.

[26] M. Quigley, K. Conley, B. Gerkey, J. Faust, T. Foote, J. Leibs, R. Wheeler, and A. Ng, "Ros: an open-source robot operating system," vol. 3, 01 2009.

[27] N. Koenig and A. Howard, "Design and use paradigms for gazebo, an open-source multi-robot simulator," in *2004 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS) (IEEE Cat. No.04CH37566)*, vol. 3, 2004, pp. 2149–2154 vol.3.

[28] A. Hornung, K. M. Wurm, M. Bennewitz, C. Stachniss, and W. Burgard, "OctoMap: An efficient probabilistic 3D mapping framework based on octrees," *Autonomous Robots*, 2013, software available at https://octomap.github.io. [Online]. Available: https://octomap.github.io

[29] M. Tiger, D. Bergström, A. Norrstig, and F. Heintz, "Enhancing lattice-based motion planning with introspective learning and reasoning," *IEEE Robotics and Automation Letters*, vol. 6, no. 3, pp. 4385–4392, 2021.

[30] O. Andersson, O. Ljungqvist, M. Tiger, D. Axehill, and F. Heintz, "Receding-horizon lattice-based motion planning with dynamic obstacle avoidance," in *2018 IEEE Conference on Decision and Control (CDC)*, 2018, pp. 4467–4474.