IEEE Access
Multidisciplinary ⋮ Rapid Review ⋮ Open Access Journal

Received 9 August 2023, accepted 24 September 2023, date of publication 28 September 2023, date of current version 4 October 2023.

Digital Object Identifier 10.1109/ACCESS.2023.3320645

RESEARCH ARTICLE

ASEP: An Autonomous Semantic Exploration Planner With Object Labeling

ANA MILAS, (Member, IEEE), ANTUN IVANOVIC, (Student Member, IEEE), AND TAMARA PETROVIC, (Member, IEEE)

Faculty of Electrical Engineering and Computing, University of Zagreb, 10000 Zagreb, Croatia

Corresponding author: Ana Milas (ana.milas@fer.hr)

This work was supported in part by Project VIRTUALUAV—Development of a System of Unmanned Aerial Vehicles (UAVs) Controlled in Virtual Environments funded by the European Regional Development Fund under Grant KK.01.2.1.02.0197, and in part by the Scientific Project Strengthening Research and Innovation Excellence in Autonomous Aerial Systems—AeroSTREAM [1] supported by European Commission HORIZON-WIDERA-2021-ACCESS-05 Program under Grant 101071270. The work of Ana Milas was supported in part by the Young Researchers Career Development Project–Training of Doctoral Students of the Croatian Science Foundation funded by the European Union from the European Social Fund.

ABSTRACT In this paper, we present a novel 3D autonomous exploration planner called the Autonomous Semantic Exploration Planner (ASEP), designed for GPS-denied indoor environments. ASEP combines real-time mapping, exploration, navigation, object detection, and object labeling onboard an Unmanned Aerial Vehicle (UAV) with limited resources. The planner is based on a frontier exploration strategy that utilizes semantic information about the environment in the exploration policy. The policy is extended to incorporate both geometric and semantic information provided by a deep convolutional neural network (DCNN) for semantic segmentation. This semantically-enhanced exploration algorithm directs the exploration toward the quick labeling of all objects of interest in the environment. An extended path planning algorithm continuously checks for path validity, enabling safe navigation in challenging environments. The overall system is designed to be modular and easily extended or replaced with custom modules. The proposed planner is evaluated and analyzed in both simulation and real-world environments using a UAV. Experimental studies demonstrate the effectiveness of the ASEP strategy compared to state-of-the-art methods. Results show that the objects in the environment are explored faster and total exploration time is reduced while the computational time remains consistent regardless of the semantic segmentation processing involved.

INDEX TERMS Autonomous exploration, semantic segmentation, UAV, path planning, object labeling.

I. INTRODUCTION

Autonomus exploration using UAVs has gained significant attention in recent years due to its numerous advantages, including operations in inaccessible or hazardous environments, increased efficiency and cost-effectiveness, mapping and monitoring capabilities, disaster response and search-and-rescue operations. When autonomously exploring indoor environments using a UAV with limited resources, where GPS signals may not be available, it can represent a significant challenge. Industries such as warehousing, logistics, inspection, or maintenance can benefit from the advancements in autonomous UAV exploration to optimize operations, reduce costs, and improve safety.

Using the semantic data from the environment in autonomous exploration is a marginally researched area so far. Generally, it includes simultaneous object detection using semantic segmentation and autonomous exploration. The result is a map of a previously unknown area with labeled objects of interest. Within this work, our objective is to determine the position and semantic label of objects of interest (hereafter referred to only as label objects of interest)

The associate editor coordinating the review of this manuscript and approving it for publication was Heng Wang.

in warehouse environments during exploration. These objects of interest may include shelves, boxes, and doors, among others, while the set of objects of interest is determined prior to exploring the environment. By using the information about the objects in the environment, our exploration system can effectively identify key regions with a high concentration of relevant objects, resulting in faster labeling of objects of interest, and at the same time building a map of the previously unknown environment. The obtained map with labeled objects can be used then for navigation, detailed visual inspections of equipment and infrastructure, counting boxes in warehouses, etc. Exploration algorithms aim to explore completely or partially unknown environments, usually as fast as possible, considering the data from the environment. State-of-the-art methods are focused on extracting data from metric maps, such as the widely used OctoMap [2] and including this data into the exploration policy. The exploration methods are mostly divided into frontier-based [3] and sampling-based [4]. In the literature, most criteria considered by exploration strategies refer only to metric information, i.e., information that can be derived from metric maps that a robot builds. Recently, a few exploration strategies proposed using semantic features to evaluate candidate locations and select the next best goal [5], [6], [7], which will be explained in detail in the following section. Semantic features from the environment can be mapped into the OctoMap so that each voxel is assigned additional information describing its semantic label. This approach is introduced in [8] and used in [5]. On the contrary, to avoid altering the OctoMap structure and make the approach applicable to other map representations, semantic features can be labeled in the 3D environment to have the position and semantic label, as described in this paper. Presented in this way, semantic features can be easily used in the exploration strategy to speed up object labeling. To the best of authors' knowledge, there is no semantically-enhanced exploration algorithm that directs the exploration to quickly label all objects of interest on the map.

Thus, we present an autonomous UAV exploration planner called Autonomous Semantic Exploration Planner (ASEP), that enables real-time mapping, exploration, navigation, object detection and labeling in GPS-denied indoor environments. By leveraging onboard sensors and processing capabilities, a UAV can detect and identify objects of interest, such as equipment, products, or inventory, in real time, providing valuable information for the exploration algorithm. We provide the integration of real-time localization, mapping and semantic segmentation onboard a UAV equipped with an RGB-D camera. The semantically segmented object from the image frame is projected on a 3D map of the environment. The proposed approach takes advantage of the semantic information extracted in 3D so that a new utility function is introduced to guide the UAV toward the objects in the environment. The proposed planner is thoroughly evaluated in both simulation and real-world environments and compared with state-of-the-art methods. The results demonstrate the proposed strategy is capable of exploring unknown environments and labeling objects effectively.

A. CONTRIBUTIONS

The key contributions of this work are summarized as follows:

- An exploration strategy that utilizes the previously introduced frontier-based method [9] to generate candidates, and combine semantic utility functions to iteratively explore the unknown environment and label all the objects of interest. This includes an extension of the information gain function to incorporate not only geometric but also semantic information of the environment.
- A 3D object labeling during exploration. This includes extraction of semantically segmented objects from 2D images and processing camera depth point cloud to estimate the position of the object in the environment.
- An overall system for mapping, exploration, path planning and navigation that is modular and can be extended or replaced by a custom module. The system utilizes a low-cost sensing system and ensures exploration and object labeling onboard a UAV with limited resources.

B. ORGANIZATION

The paper is organized so that in Section II we give an overview of the state-of-the-art of 3D exploration methods and position our work in relation to them. Section III describes problems solved within this paper while Section IV introduces system and sensor models used in the proposed method. Section V is the core of the paper and contains details of the proposed planner. Results of simulations performed with a UAV and their analysis are presented in Section VI while Section VII shows our experiment setup and results in a real-world indoor environment. The paper is concluded in Section VIII.

II. RELATED WORK

In the context of warehousing, UAVs are used to collect data and provide certain information for tasks such as inventory management, monitoring stockpiles, and inspecting hard-to-reach areas within a warehouse facility. UAVs used in warehouse exploration missions should be able to effectively navigate the environment and gather information. To achieve this, UAVs should be able to localize themselves, detect objects, explore and map the environment. This paper mainly focuses on environment exploration and object labeling on the map. Thus, in this section, state-of-the-art methods for autonomous exploration are overviewed.

There is a wealth of earlier work related to autonomous exploration, especially for 2D, but more recently also for 3D environments. Typical exploration approaches can be roughly classified into frontier-based, sampling-based, and hybrid strategies, even though there is a significant overlap between these categories.

A characteristic of frontier-based approaches is exploration by approaching a selected point on the frontier between the explored and unexplored portion of the environment. This idea was first introduced by Yamauchi in [3] and subsequently evaluated in more detail in [10]. In each iteration, the next best goal is a frontier point closest to the robot. The simplest approach to 3D exploration is to use 2D frontier-based exploration with 3D maps at different heights (oftentimes called 2.5D approaches) [11]. A complete frontier-based solution for 3D environments is developed in [12], where the next best goal is the frontier that minimizes the velocity change to maintain a consistently high flight speed. It is shown that this approach outperforms the closest frontier method [3]. Frontier-based exploration approaches for 3D environments are also researched in [9], [13], [14], [15], [16], [17], and [18].

Sampling-based approaches aim to determine a (minimal) sequence of robot (sensor) viewpoints to visit in the environment, until the entire space is explored. Potential viewpoints are typically sampled near the frontier or randomly. Then these viewpoints are evaluated for the potential information gain and the next best viewpoint is assigned. One of the first sampling-based methods is presented in [19] and then extended in [4], [20], and [21]. In [4], authors proposed the Receding Horizon Next-Best-View planning (RH-NBVP), which uses a Rapidly-exploring Random Tree (RRT) [22], [23] to guide a UAV into the unexplored area. While the method showed good scaling properties and performance in a local exploration, it is not resilient to dead ends, resulting in poor global scene coverage and, thus, a high overall exploration time, as shown in [12], [15], and [24], and in our previous work [9]. Improvements of the RH-NBVP are presented in [25], [26], [27].

Hybrid strategies combine the advantages of both frontier-based and sampling-based approaches. Selin et al. [28] successfully combine the RH-NBVP with conventional frontier reasoning to compensate for a poor performance in global exploration. In other words, [28] plans global paths towards frontiers and samples paths locally. Meng et al. [29] samples viewpoints around frontiers and finds the global shortest tour passing through them. Similarly, Respall et al. [30] samples viewpoints in the vicinity of a point of interest near a frontier, and additionally memorizes nodes that indicate regions of interest in a history graph to reduce the gain calculation time.

Apart from classification related to candidate extraction and evaluation, exploration algorithms differ in the map used for exploration policy. Besides the volumetric map, such as the OctoMap [2], the environment can also be represented by a topological map with semantic features [31], which can improve the efficiency of the robotic exploration by facilitating the next best goal selection. The nodes on the graph that contain the semantic features are used to guide the exploration. Gomez et al. [32] presented a hybrid mapping approach that combined topological mapping with 3D dense mapping for large indoor 3D environments.

Recently, more and more exploration systems use semantic features from volumetric maps to evaluate candidate locations and select the next best goal. The authors in [6] extend the sampling-based approach from [4] to include the semantic segmentation information in a harbor-like environment. Similarly, Ashour et al. [5] presents an exploration strategy for UAVs that integrates environmental semantics for the object mapping. The approach combines semantic information with autonomous exploration techniques to guide the exploration path and enhance object mapping efficiency using the approach from [8]. Instead of mapping objects during the exploration, objects can be extracted from 2D images and then converted to 3D point types using the point cloud library (PCL). Previously, Wang et al. [33] introduced the extraction of edges. Furthermore, most of the semantic-aware exploration strategies are goal-oriented (search for an object), such as [7], [34], and [35]. Authors in [7] introduced a frontier semantic exploration method for visual target navigation. Both frontier detection and semantic segmentation are performed using neural networks.

Regarding the navigation and operations in the warehouse environment using UAVs, Campos-Macias et al. [36] presented an autonomous navigation framework for capturing inventory and locating out-of-place items while focusing on the exploration in unknown 3D cluttered environments. They used an RGB-D camera for depth sensing and a tracking camera for the visual-inertial odometry. Kwon et al. [37] demonstrated autonomous navigation for inventory inspection tasks in long and narrow warehouse aisles using a low-cost sensing system. Their system consists of a relatively small number of sensors, including three cameras, a laser scanner and a range sensor.

Even though efforts to improve the efficiency, accuracy, and robustness of autonomous exploration have shown promising results, it is important to note that a semantically-enhanced exploration algorithm for onboard UAV applications, which focuses on fast object labeling, has not yet been developed in the literature. With this in mind, we present a novel autonomous exploration strategy specifically designed for UAVs with limited payload capabilities and computational resources. Our approach integrates real-time mapping, exploration, navigation, object detection, and object labeling capabilities directly onboard the UAV. The key component of our proposed strategy is a frontier exploration planner that incorporates semantic information about the environment into the exploration policy. By leveraging this semantic understanding, our planner enables the UAV to make informed decisions regarding which frontiers to explore and, thus, directs the exploration toward the quick labeling of all objects of interest on the map. By combining real-time mapping, exploration, navigation, object detection, and object labeling, our approach addresses the limitations of

existing algorithms and provides a comprehensive solution for autonomous exploration onboard UAVs.

III. PROBLEM DESCRIPTION

The main goal of the proposed approach is to explore a bounded and previously unknown 3D space $V \subset \mathbb{R}^3$ and to label objects of interest in a 3D map as soon as possible. As a basis for our approach, an OctoMap M is used, a hierarchical volumetric 3D representation of the environment [2]. Each cube of the OctoMap is denoted as a voxel (cell), which can be *free*, *occupied* or *unknown*. Free voxels form the free space $V_{free} \subset V$, occupied voxels form the occupied space $V_{occ} \subset V$ and unknown voxels form the unknown space $V_{un} \subset V$. Initially, the entire bounded space is unknown, $V \equiv V_{un}$, and the unknown space decreases as the exploration advances. The entire space is a union of the three subspaces $V \equiv V_{free} \cup V_{occ} \cup V_{un}$. The exploration problem is considered fully solved when $V_{occ} \cup V_{free} \equiv V \setminus V_{res}$, where V_{res} is residual space defined as an unexplored space, which remains inaccessible to the sensors. Namely, sensors have limitations in perceiving surfaces, leading to an inability to explore hollow spaces or narrow pockets within a given setup.

The object labeling in a map M is executed in parallel with the exploration. Let O be the set of objects of interest present in the map. The set O is defined as:

$$O = \{o_i | i = 1, 2, \ldots, N_{obj}\}, \tag{1}$$

where N_{obj} is the total number of objects in the 3D map. Each object o_i is represented by its 3D position and its semantic label s_i, selected from the set of semantic labels:

$$S = \{s_i | i = 1, 2, \ldots, N_{labels}\}, \tag{2}$$

where N_{labels} is the total number of different semantic labels. It is preset and depends on the elements expected in the environment. In this work, the focus is on static objects and the semantic labels are related to a warehouse scenario. Each object o_i is defined as $o_i = (\mathbf{p}_{obj_i}, s_i)$, where $\mathbf{p}_{obj_i} = \begin{bmatrix} x_i & y_i & z_i \end{bmatrix}^T \in \mathbb{R}^3$ is the object position. The semantic labels of objects are obtained by the semantic segmentation algorithm described in IV-C. Given the nature of this problem, it is crucial to employ an algorithm capable of detecting objects and estimating their positions in real-time, while exploring and mapping the unknown envronment. Additionally, a suitable and obstacle-free path should be computed online. The autonomy of the algorithm requires the planner to run onboard with limited computational resources.

IV. SYSTEM OVERVIEW

In the subsequent section, a detailed description of the proposed system is provided, including the sensors used and the methodologies employed for the simultaneous localization and mapping (SLAM), as well as semantic segmentation based on 2D images and object pose estimation from RGB-D data.

An overview of the proposed system is given in Fig. 1. It shows the proposed semantically-aware exploration system architecture, which consists of five main modules: 1) system input, 2) localization and mapping, 3) semantic data extraction, 4) exploration, and 5) path planning and navigation. A detailed description of all modules is given in the following sections.

A. UAV AND SENSOR MODELS

In this work, the exploration is performed with a UAV that has no prior knowledge of the environment. Although the concepts are explained with the UAV in mind, the same approach is applicable to other autonomous robots equipped with a camera or other sensors that can be used to utilize SLAM and build an OctoMap.

The UAV is represented with a state vector $\mathbf{x} = \begin{bmatrix} \mathbf{p}^T & \psi \end{bmatrix}^T \in \mathbb{R}^4$ that consists of the position $\mathbf{p} = \begin{bmatrix} x & y & z \end{bmatrix}^T \in \mathbb{R}^3$ and the yaw rotation angle around the body z axis $\psi \in [-\pi, \pi)$. Furthermore, the algorithm requires dynamical constraints in terms of velocity $\dot{\mathbf{x}}_{max} \in \mathbb{R}^4$ and acceleration $\ddot{\mathbf{x}}_{max} \in \mathbb{R}^4$ for each degree of freedom. For collision checking, it is considered that the UAV is inside a rectangular prism centered at $\mathbf{p}$, with adequate length, width and height l, w, h.

The algorithm relies on a maximum range of the sensor $R_{max} \in \mathbb{R}$ with horizontal and vertical Field of View (FOV) in range, α_h, $\alpha_v \in (0°, 360°]$, respectively. This allows our algorithm to work with point-cloud-producing sensors with various FOV, such as cameras with limited FOV, and LiDARs with limited α_v. In this paper, an RGB-D camera that provides rich visual and depth information is used, allowing the UAV to build a detailed and accurate map of the environment and localize itself using the visual SLAM method.

B. LOCALIZATION AND MAPPING

The sensing system described above allows the robot to estimate its pose and capture point clouds of its environment. However, this sensed data is not sufficient by itself to create a consistent global 3D map. As the robot moves around the world, measurements from these sensors must be integrated, taking into account the motion of the robot, to create a coherent global representation of its surroundings, i.e., a map. In many cases, the pose of the robot in the map will be estimated at the same time as the map is built, which is often known as localization; in this case, the task is usually referred to as SLAM.

During autonomous exploration, robots need to navigate in unknown or partially known environments, gradually perceiving the environment through streaming data provided by onboard sensors. A majority of the 3D strategies use a metric map, an OctoMap [2], in order to navigate through 3D space and visualize the environment. The OctoMap is a hierarchical volumetric 3D representation of the environment. The OctoMap can be generated using the input from the SLAM algorithm [39], as shown in [9], or with raw data from a sensor system, such as a laser scanner or a camera, as demonstrated in [27]. In the proposed system, a camera point cloud is used to generate an OctoMap, as shown

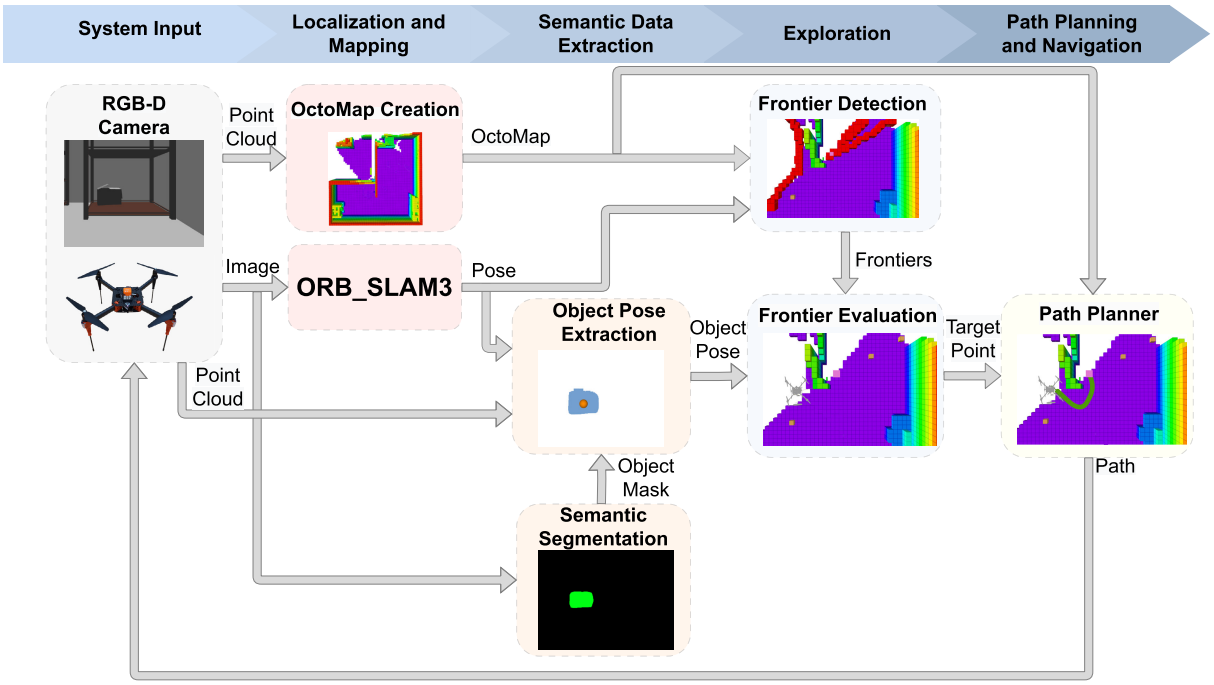

FIGURE 1. Overall schematic diagram of the 3D exploration. The system input module consists of a forward-facing camera that produces input data for both SLAM and OctoMap creation. The semantic segmentation module and object pose extraction module ensure object recognition and object pose estimation in a 3D map, respectively. The semantically-enhanced exploration method generates a target point towards which the robot plans its path and navigates. The navigation system relies on an off-the-shelf SLAM system with loop closing and relocalization capabilities [38], which is fed with RGB-D images.

in Fig. 1. Localization and mapping module results in an OctoMap of the environment and in the pose provided by a SLAM algorithm. An OctoMap is used for both exploration (frontier detection) and path planning and navigation module (collision-free navigation).

For visual SLAM ORB-SLAM3 [38] is utilized, which performs visual, visual-inertial and multimap SLAM with different camera types (monocular, stereo and RGB-D). Within this paper, RGB-D camera is used to perform ORB-SLAM3. It uses depth information to synthesize a stereo coordinate for extracted features on the image. This way SLAM system is agnostic of the input being stereo or RGB-D. As shown in [40] and [41], the RGB-D SLAM outperforms state-of-the-art methods in most sequences on TUM RGB-D dataset.

To avoid possible sudden jumps resulting from the ORB-SLAM3 algorithm loop closures, a multi-sensor fusion method introduced in [42] is used. In this case, the multi-sensor fusion method takes into account the inertial measurement unit (IMU) data obtained from the LPMS-CU2 unit, together with the pose measurements from the ORB-SLAM3.

C. 2D-IMAGE-BASED SEMANTIC SEGMENTATION

An input image from an RGB-D camera can be represented as a 2D array of pixel values. Semantic segmentation is a computer vision technique that involves labeling each pixel of the image with a specific class or category. The objective of the semantic segmentation is to predict the segmentation map for the input image, but instead of containing pixel values, it contains the predicted semantic labels for each pixel. To each object of interest, o_i corresponds a collection of pixels that form a distinct entity that can be visually identified and distinguished from the background or other elements in the image. Then, using a semantic segmentation algorithm, the semantic labels s_i for each object o_i can be determined. In other words, the goal is to find a function $f : O \rightarrow S$ that maps each object to its corresponding semantic label. Given a number of semantic labels N_{labels}, the determination of semantic labels for each object can be expressed as:

$$f(o_i) = s_j \quad \text{for } 1 \leq i \leq N_{obj}, 1 \leq j \leq N_{labels}, \tag{3}$$

where o_i is the object, and s_j is a semantic label, from the set S and for the given object.

By utilizing deep learning models, this approach can accurately segment objects and regions of interest in 2D images. In our work, for semantic segmentation, the HRNet [43] is used, which is a recently proposed model that retains high-resolution representations throughout the model, without the traditional bottleneck design. The model architecture is shown in Fig. 2.

The HRNet model for the 2D image semantic segmentation is used since it showed enviable performance results [45]. Furthermore, it is compact, fast, robust and easy to use, enabling the model adaptation to work on CPU only, making it suitable for applications running on UAVs with

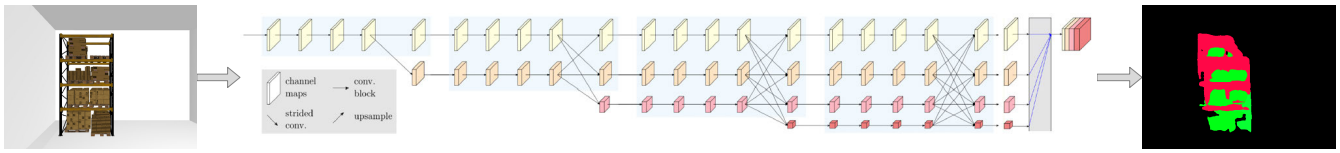

FIGURE 2. HRNet architecture applied to the task of semantic segmentation. The input image from the Gazebo simulator contains a shelf and boxes. The corresponding output image shows that the HRNet has successfully identified and segmented the objects. ©2021 IEEE. Reprinted with permission from [44].

limited computational resources. The model is trained on the ADE20K dataset with 150 objects and stuff classes included. ADE20K is the largest open-source dataset for semantic segmentation and scene parsing, released by the MIT Computer Vision team [46], [47]. The ADE20K dataset is selected since some datasets have a limited number of objects (e.g., COCO [48], Pascal [49]) and in many cases, those objects are not the most common objects one encounters in indoor environments, or the datasets only cover a limited set of scenes (e.g., Cityscapes [50]). Additionally, objects of interest are extracted from the semantically segmented image. In the case of warehouse exploration, the objects of interest are shelves, boxes, doors, etc. Fig. 2 illustrates an example of semantic segmentation performed on an image with a shelf and multiple boxes. As shown in Fig. 2, it is not important that each individual item (e.g., a box) from a set of items (e.g., boxes) is detected, but at least one that is then used in the exploration policy.

D. OBJECT POSITION EXTRACTION

Once objects are semantically segmented in a 2D image, the next step is to determine the 3D positions of those objects in the world to be used for 3D exploration. This process involves utilizing both an object mask and a camera point cloud (Fig. 1). For each object o_i with semantic label s_i from the set S, the 3D position of the object $\mathbf{p}_{obj_i} = \begin{bmatrix} x_{oi} & y_{oi} & z_{oi} \end{bmatrix}^T$ needs to be found. The image mask serves as a binary representation of the segmented objects in the image, where each pixel belonging to an object is assigned a value of 1, while pixels outside the objects are assigned a value of 0. This mask essentially acts as a filter, isolating the regions of interest from the background and other irrelevant elements in the image. Masks are then extracted for each object detected in the image. Alongside the image mask, a camera point cloud C is utilized. In general, a point cloud C is defined as:

$$C = \{\mathbf{c}_i | i = 1, 2, \ldots, N\}, \tag{4}$$

where $\mathbf{c}_i \in \mathbb{R}^3$ while N represents the number of points. In other words, a point cloud is a collection of 3D points that represent the surface geometry of objects in the scene.

To perform the object position extraction, the algorithm utilizes both the image mask and the camera point cloud. It associates the segmented objects in the image mask with their corresponding points in the point cloud. If H_{o_i} is the 2D image mask corresponding to the object o_i, then $C_{o_i} \in C$ is the set of 3D points from RGB-D camera that is aligned with the mask. It means that for the image mask of each object, there is a corresponding point cloud. The 3D position $\mathbf{p}_{obj_i}$ of the object o_i is then determined using point cloud C_{o_i}. Namely, the centroid technique is used to estimate the position from the points in C_{o_i}, expressed in the camera coordinate frame:

$$\mathbf{p}^{\mathrm{c}}_{obj_i} = \frac{\sum_{i=1}^{N_{pixels}} \mathbf{c}_{o_i}}{N_{pixels}}, \tag{5}$$

where N_{pixels} represents the total number of points aligned with the image mask of the object. Note that the number of points from the point cloud aligned with the mask is equal to the number of pixels from the image mask. By aligning the 3D points with the 2D image coordinates, the algorithm determines the position of each object in the coordinate system of the camera. Given the UAV state in the world frame, the position of the object in a global 3D map can be determined:

$$\mathbf{p}^{\mathrm{w}}_{obj_i} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \cdot \mathbf{T}^{\mathrm{b}}_{\mathrm{w}} \cdot \mathbf{T}^{\mathrm{c}}_{\mathrm{b}} \cdot \begin{bmatrix} \mathbf{p}^{\mathrm{c}}_{obj_i} & 1 \end{bmatrix}^T, \tag{6}$$

where $\mathbf{T}^{\mathrm{b}}_{\mathrm{w}} \in \mathbb{R}^{4\times4}$ is the homogeneous transform matrix defining the position and orientation of the UAV in the world coordinate frame, and $\mathbf{T}^{\mathrm{c}}_{\mathrm{b}} \in \mathbb{R}^{4\times4}$ defines the fixed transformation between the UAV body and camera. Note that the first matrix in the equation transforms the four-dimensional vector into a three-dimensional position vector. This combined approach effectively maps the 2D image objects to their corresponding 3D locations and allows for accurate and robust extraction of object positions, as shown in Section VI.

V. SEMANTIC-BASED EXPLORATION

In this section, the proposed semantic-based exploration method is described in detail. Our previously developed frontier-based method [9] is used to extract frontier voxels (frontiers) from the OctoMap. Those frontiers are candidates for the next waypoints of exploration. Each candidate is evaluated using a semantically-aware policy and, finally, the best candidate is selected as the next waypoint to which the UAV plans a path and navigates. The main contribution in

this part is the extension of the information gain function to include semantic data of the environment.

A. FRONTIER DETECTION

A frontier, F, can be defined as a set of voxels $\mathbf{v}_f$ with the following property [9]:

$$F = \{\mathbf{v}_f \in V_{free} : \exists neighbor(\mathbf{v}_f) \in V_{un}\}. \tag{7}$$

In other words, a frontier consists of free voxels with at least one unknown neighbor. The center of a frontier voxel is often called the frontier point. Since the space V is bounded, once the frontier set becomes empty, $F = \emptyset$, the exploration is considered done.

The OctoMap used for frontier detection is generated using camera point clouds C. The OctoMap is in the form of OcTrees, a format suitable for path planning. During the exploration, the OctoMap M is built iteratively using the method described in [2]. Within this work, the current OctoMap M^i is created from the current point cloud C^i added to the OctoMap explored so far:

$$M^i = f(M^{i-1}, C^i), M^0 = \emptyset. \tag{8}$$

With each new-coming point cloud, a new OctoMap is created according to Eq. 8. The OctoMap is updated as each point cloud is processed. At the same time, a frontier detection cycle is performed periodically to ensure that frontiers are constantly updated. Please note that the rate of an OctoMap update process is lower than the frontier detection process since the OctoMap update is a computationally demanding process, especially when using dense point clouds.

Let V^i_{free} and V^{i-1}_{free} correspond to the free voxels in two consecutive OctoMaps, M^i and M^{i-1}. The local frontier F_l, which contains only newly created frontier points can be calculated as follows [9]:

$$F_l = \{\mathbf{v}_f \in V^i_{free} \setminus V^{i-1}_{free} : \exists neighbor(\mathbf{v}_f) \in V^i_{un}\}. \tag{9}$$

The global frontier F_g is a union of all past local frontiers, updated in each iteration and filtered to exclude voxels that do not satisfy the property Eq. 7 anymore. F_g is calculated as follows:

$$\begin{aligned} F^i_g &= F^i_l \cup F^i_{gf} \\ F^i_{gf} &= \{\mathbf{v}_f \in F^{i-1}_g : \exists neighbor(\mathbf{v}_f) \in V^i_{un}\}, F^0_g = \emptyset. \end{aligned} \tag{10}$$

There is usually a large number of voxels in the global frontier (referred to only as frontier from now on) and their evaluation is expensive in view of the computing effort involved. In our previous work, we cluster the global frontier voxels F^i_g, as explained in [9], to get frontier voxels which are candidates, denoted as F_c, for becoming a next waypoint for the exploration. As stated in [9], the frontier is clustered using multi-resolution clustering and mean shift clustering algorithms. In the proposed approach, candidates F_c are frontier F_g, clustered using the mean shift clustering algorithm.

B. SEMANTICALLY-AWARE FRONTIER EVALUATION

The main goals of this approach are to explore the environment and label the objects of interest on the map. Labeled objects of interest are included in the exploration policy, assuming that this leads to faster object labeling. To evaluate each voxel in F_c, the *total gain* of every candidate $\mathbf{v}_c \in F_c$ is defined using the following function:

$$G(\mathbf{v}_c) = \alpha I_{gg}(\mathbf{v}_c) + \beta I_{sg}(\mathbf{v}_c), \tag{11}$$

where α and β are positive constants, while $I_{gg}(\mathbf{v}_c)$ and $I_{sg}(\mathbf{v}_c)$ represent *geometric information gain* and *semantic information gain* of each candidate $\mathbf{v}_c$, respectively. Therefore, α and β represent the trade-off between the geometric and semantic information gain. The values of α and β are experimentally determined and depend on the environment layout.

The geometric information gain $I_{gg}(\mathbf{v}_c)$ is defined using the function similar to the one proposed in [19]:

$$I_{gg}(\mathbf{v}_c) = \frac{I_{un}(\mathbf{v}_c)}{e^{\lambda L(\mathbf{p}_i, \mathbf{p}_{\mathbf{v}_c})}}, \tag{12}$$

where λ is positive constant, $L(\mathbf{p}_i, \mathbf{p}_{\mathbf{v}_c})$ is the distance between the robot's current position $\mathbf{p}_i$ and the position of the candidate $\mathbf{p}_{\mathbf{v}_c}$, while $I_{un}(\mathbf{v}_c)$ is a *information gain* i.e. a measure of the unexplored region of the environment that is potentially visible from $\mathbf{v}_c$. A high information gain indicates that a specific $\mathbf{v}_c$ provides significant information about the environment, while a low information gain suggests that the $\mathbf{v}_c$ contributes less to reducing unknown space. The information gain $I_{un}(\mathbf{v}_c)$ is defined as the share of unknown voxels in a cube placed around $\mathbf{v}_c$, as described in [9]. The cuboid width and height are defined by the parameter I_{range}, which depends on the used sensor range and the environment size. Often, the information gain is estimated using a ray tracing algorithm and a real sensor field of view instead of using a cube-based approximation. By using the proposed simplification, the high calculation effort required by ray tracing is avoided. The estimated distance is approximated using the Euclidean distance between the robot position $\mathbf{p}_i$ and the position of the candidate (voxel center) $\mathbf{p}_{\mathbf{v}_c}$, $L(\mathbf{p}_i, \mathbf{p}_{\mathbf{v}_c}) = \|\mathbf{p}_i - \mathbf{p}_{\mathbf{v}_c}\|$. The constant λ weights the importance of robot motion cost against the expected information gain. A small λ gives priority to the information gain, while $\lambda \to \infty$ means that the motion is so expensive that only $\mathbf{v}_c$ near the robot is selected. As described in [9], λ is set to satisfy the ratio between the desired information gain and the distance with respect to the desired behavior of the system. To include semantically segmented objects from the environment in the exploration policy, $I_{sg}(\mathbf{v}_c)$ is introduced, as shown in Eq. 11. $I_{sg}(\mathbf{v}_c)$ represents the semantic information gain of each candidate. Let n_{obj} be the number of currently semantically segmented objects in

the environment, then $I_{sg}(\mathbf{v}_c)$ is defined as:

$$I_{obj}(\mathbf{v}_c) = \begin{cases} \dfrac{1}{L(\mathbf{p}_{obj}, \mathbf{p}_{\mathbf{v}_c})} & \text{if } L(\mathbf{p}_{obj}, \mathbf{p}_{\mathbf{v}_c}) \leq I_{\text{range}}, \\ 0 & \text{otherwise}, \end{cases} \tag{13}$$

$$I_{sg}(\mathbf{v}_c) = \sum_{obj=1}^{n_{obj}} I_{obj}(\mathbf{v}_c), \tag{14}$$

where $L(\mathbf{p}_{obj}, \mathbf{p}_{\mathbf{v}_c})$ is the distance between the position of the object $\mathbf{p}_{obj}$ and the position of the candidate $\mathbf{p}_{\mathbf{v}_c}$. Position of the object $\mathbf{p}_{obj}$ is calculated as stated in Subsection IV-D. In other words, the semantic information gain of each candidate $\mathbf{v}_c$ is the sum of all visible objects from the candidate $\mathbf{v}_c$ inversely proportional to the distance of the object.

Finally, the best frontier voxel is one that maximizes the total information gain $G(\mathbf{v}_c)$:

$$\mathbf{v}_{bf} = \arg\max_{\mathbf{v}_c \in F_C} G(\mathbf{v}_c). \tag{15}$$

The best frontier voxel $\mathbf{v}_{bf}$ is forwarded as a target point to path planner module.

C. PATH PLANNING AND NAVIGATION

As soon as the best frontier point is selected, it is forwarded to a path planner as a waypoint. The robot starts to follow the planned path and navigates to the best frontier point $\mathbf{v}_{bf}$.

The path planning module includes the Rapidly-exploring Random Tree Star (RRT*) algorithm, an extension of the original RRT algorithm. Unlike the RRT, RRT* improves the convergence properties of generated paths by performing rewiring operations during the expansion phase, unlike the RRT algorithm that terminates upon finding a first feasible path. This adaptive rewiring step allows the tree to continuously refine the path as more iterations are performed, eventually converging to near-optimal solutions. The approach utilized within this paper has been developed in our previous work [51], work [51], [52], and is available online [53]. In each iteration, the planner avoids occupied voxels in the OctoMap and generates a path through the free voxels up to the best frontier point. The crucial part of the planner is the state validity checker, which evaluates the validity of configurations based on system constraints such as collision avoidance and environment-specific criteria. As new configurations are sampled or interpolated between the existing configurations during the RRT* expansion step, the state validity checker is invoked to evaluate their validity. In the practical implementation, the path planner takes a binary representation of the OctoMap as an input, which provides an efficient and compact description of the environment. The UAV is represented as a rectangular prism of appropriate dimensions within the state validity checker.

Once a target point is specified and the state validity checker is defined, the path planner creates a collision-free path from the current UAV position to the target point. However, during the path execution, the OctoMap is updated and newly discovered obstacles may appear in or near the path, as noticed in the experimental analysis within our previous work [9]. To overcome this issue, the path planner is extended to check the validity of the path during motion, as the OctoMap is updated. For each point along the path, it checks whether the UAV can execute it without interfering with obstacles, as shown in Fig. 3.

If the validity checker detects an obstacle on the path up to the current goal point, the UAV is stopped and the current goal point is classified as unreachable. The exploration planner starts a new iteration (frontiers update, the best frontier selection and path planning). This ensures that the UAV does not attempt to traverse newly discovered obstacles, increasing the safety and efficiency of the exploration. Once the goal point is set as unreachable, it is no longer considered a candidate for the best frontier during the exploration process. Re-planning to the same goal point is left for future work.

A new cycle of the procedure to determine the best frontier point is started either when the previous path is discarded or after the previous frontier point is reached by the UAV. During the exploration, the number of frontiers is changing and once the entire environment is explored and a complete map of the environment is created, the exploration process is considered done.

The path execution and UAV control are achieved using an MPC-based tracking method. The original implementation is presented in [54] while an adapted version of their work is presented in [55] and used in this paper. The main motivation for using this tracking method is that it allows the UAV to smoothly follow and quickly change the UAV trajectory based on the current system state and model dynamics. Furthermore, the tracker enables safe and stable flight, regardless of the target point resulting from the exploration planner.

VI. SIMULATION ANALYSIS

Simulations are performed in the Gazebo environment using the Robot Operating System (ROS) and a quadcopter. The quadcopter is equipped with a camera with specifications from Intel(R)RealSense(TM) Depth Camera D455. It has a horizontal and vertical FOV $\alpha_h = 90°$, $\alpha_v = 65°$, respectively, and maximum depth defined in Table 1. For collision checking, the dimensions of a rectangular prism around the UAV are set to $l = 0.6$ m, $w = 0.6$ m, $h = 0.5$ m.

The proposed ASEP algorithm is compared with the closest frontier method (CF) introduced by [3] and adapted to our planner. Additionally, it is compared to our more recent multi-resolution frontier planner (MRF) [9]. The parameters used in the MRF are set to their default values explained in [9], with velocities as given in Table 1. Both the CF and the MRF are adapted to our quadcopter, equipped with a camera, and to our control system to allow the fairest possible comparison. The approaches are compared in two scenarios

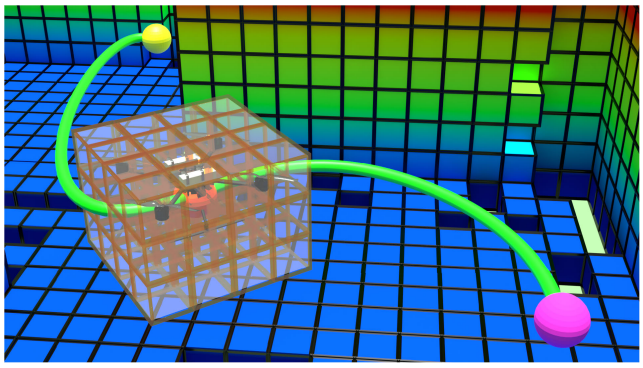

FIGURE 3. A UAV executes a planned path (green) from a start point (pink) to a target point (yellow) in an OctoMap. A rectangular prism-shaped state validity checker (transparent white and orange) simplifies the representation of the UAV. The validity checking is performed for each point (orange point) of the planned path.

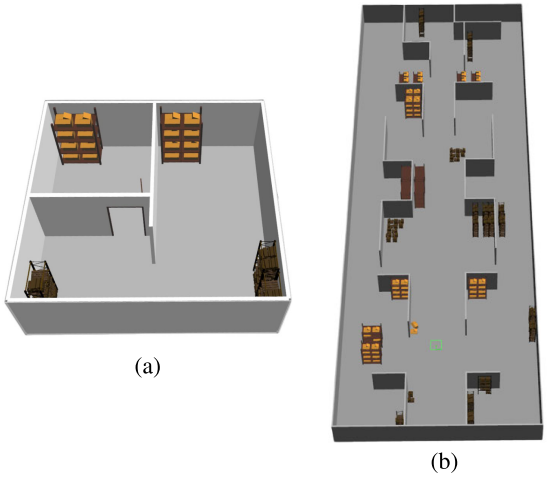

FIGURE 4. Gazebo warehouse scenarios. (a) Simple warehouse scenario. (b) Complex warehouse scenario.

(Fig. 4) with different environment sizes and OctoMap resolutions r. All simulations have been run 10 times on Intel(R)Core(TM) i7-10750H CPU @ 2.60GHz × 12.

The first scenario refers to a 10 m × 10 m × 3 m relatively simple warehouse (Fig. 4 (a)). The second scenario refers to a 20 m × 60 m × 3 m complex warehouse environment (Fig. 4 (b)). Please note that in the simulation analysis the odometry is provided by the simulator, while in real experiments the odometry is provided by the SLAM. This is mainly due to the fact that the camera simulation is not realistic enough to be suitable for the ORB-SLAM3 algorithm.

A. OBJECT LABELING

Within this work, the objective is to label objects in warehouse scenarios (shelves, boxes and doors). It means that N_{labels} is set to three. Note that the total number of labels in a general case is adapted to the environment and the elements expected in it. The total number of objects is initially unknown, but after exploration, it is

TABLE 1. Exploration parameters for simulation scenarios. r is OctoMap resolution, R_{max} is the maximum range of the camera, $\dot{\mathbf{p}}_{max}^{\{x,y,z\}}$ and $\dot{\psi}_{max}$ are max UAV velocities in the x, y, z axes and yaw direction, respectively, while $\ddot{\mathbf{p}}_{max}^{\{x,y,z\}}$ is the acceleration for the same degrees of freedom. I_{range} is the cuboid width and height used for information gain calculation, while λ, α and β are constants used in equations for exploration policy.

Parameter	Simple warehouse	Complex warehouse	Real world
r [m]	0.1, 0.2	0.1, 0.2	0.1, 0.2
R_{max} [m]	6.0	6.0	6.0
$\dot{\mathbf{p}}_{max}^{\{x,y,z\}}$ [m/s]	1.0	1.0	0.25
$\ddot{\mathbf{p}}_{max}^{\{x,y,z\}}$ [m/s^2]	0.5	0.5	0.2
$\dot{\psi}_{max}$ [rad/s]	0.75	0.75	0.25
I_{range} [m]	3.0	5.0	5.0
λ	0.1	0.1	0.1
α	0.35	0.35	0.35
β	0.7	0.7	0.7

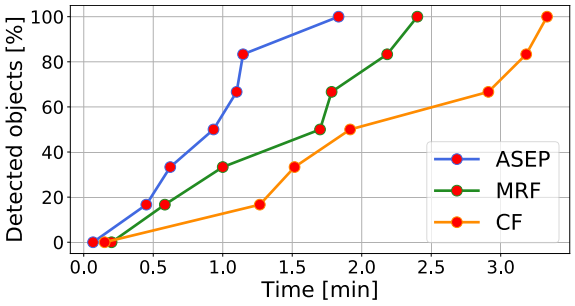

FIGURE 5. The detected objects in time for the simple warehouse environment at $r = 0.2$ m.

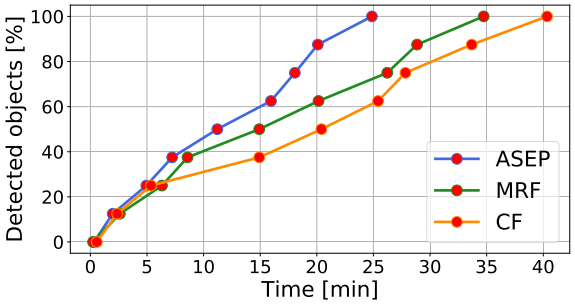

FIGURE 6. The detected objects in time for the complex warehouse environment at $r = 0.1$ m.

$N_{obj} = 30$ and $N_{obj} = 160$ for the simple and complex scenarios, respectively. All three approaches are compared in both simulation scenarios, with different resolutions. The results are shown in Fig. 5 and Fig. 6 as a percentage of detected objects in time. It can be observed that the ASEP needs less time to detect all given objects since it is directed by the semantic information from the environment. For instance, in a complex environment, all objects are detected and labeled on the map in 25 minutes when using ASEP, while MRF and CF need 35 and 40 minutes, respectively.

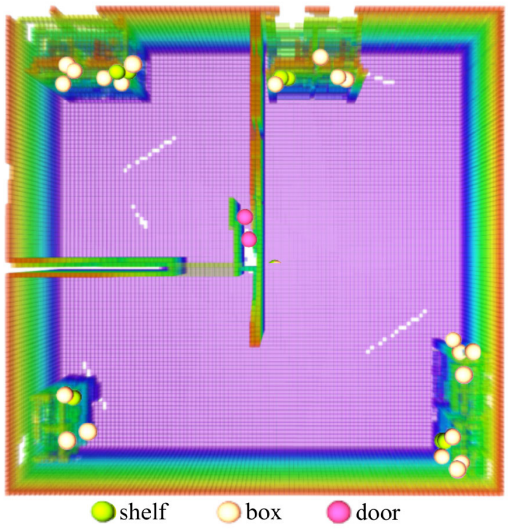

FIGURE 7. The OctoMap of the simple warehouse environment with transparency effect applied on and with detected objects during the exploration. Shelves, boxes and doors are shown with different colors on the map.

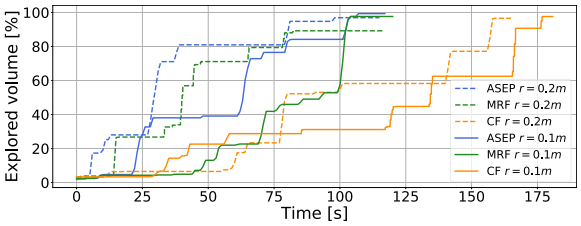

FIGURE 8. The explored volume in total exploration time for the simple warehouse scenario.

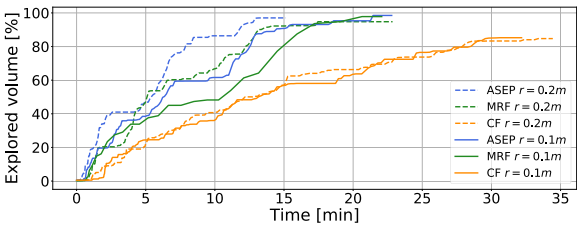

FIGURE 9. The explored volume in total exploration time for the complex warehouse scenario.

Fig. 7 shows the labeled objects on the map of the simple warehouse scenario. However, due to factors such as UAV tilting and limitations of the detection algorithm, the object positions in the OctoMap are not entirely accurate. To address this, any new object that is within 0.3 m of an existing object with the same semantic label is averaged with the centroid of the existing object, and the object position in 3D (described in IV-D) is then updated. In this way, multiple labeling of the same object is avoided. Consequently, labeled objects deviate from their real positions in the environment, given by the simulator. The mean deviation of each labeled object from its real position in the simulator is calculated by comparing the position of the semantic label to the real position of the closest object of that label obtained from the simulator. The calculated values are as follows: 0.41 m, 0.48 m, and 0.54 m for boxes, shelves, and doors, respectively. These values indicate that, considering the overall scale and dimensions of the scenarios, the position of the labeled objects matches very well with their actual position in the environment.

B. COMPUTATION PERFORMANCE

Computation times t_c and total exploration times t_{exp} for all 10 runs are shown in Table 2. The performance of all three approaches is compared in both simulation scenarios and at different resolutions. The computation time is equal to the time required to detect frontiers, update global frontiers and find the best frontier $\mathbf{v}_{bf}$. For the MRF, t_c includes time to cluster frontiers using multi-resolution frontier clustering (with exploration depth d_{exp} set to 15) and mean shift clustering algorithm, as described in [9]. The clustering methods in the MRF affect the t_c and thus the t_{exp}. It can be observed that the computation times for the MRF approach are higher than in our approach, especially when using a high-resolution map. Such results are expected since the MRF multi-resolution frontier clustering is computationally expensive, especially with a high number of frontiers. Furthermore, the use of multi-resolution frontier clustering in the MRF causes the computation time to increase as the complexity of the environment increases. On the other hand, the computation times in the CF depend mainly on the size of the environment and the number of frontiers, rather than on the map resolution (which is evident in both scenarios). For instance, in the simple scenario at the resolution $r = 0.1$ m, our planner runs more than ten times faster than the MRF. The ASEP and the CF have similar computation times since the difference is only in information gain calculation. The computation time in the proposed planner increases with the finer resolution but is still low enough not to affect the exploration. The results have confirmed that the multi-resolution frontier clustering may cause a bottleneck during exploration in larger and more complex scenarios with a large number of frontiers and at fine resolutions.

C. GLOBAL EXPLORATION USING PROPOSED PLANNER

Simulations were also performed to compare the total exploration time of our exploration planner with the MRF and the CF. The algorithms were tested using a voxel resolution of $r = 0.2$ m and $r = 0.1$ m. Note that the goal of the ASEP is to label objects of interest as soon as possible. Therefore, the placement of objects of interest in the environments may affect the total exploration time. However, a comparison of total exploration time shows that it is possible to perform exploration in a comparable time to the state-of-the-art algorithms, while in the same time doing the object labeling.

Fig. 8 shows the explored volume over time for algorithms in simple warehouse scenario. It can be observed that the

TABLE 2. The tuples of mean and standard deviation for the total exploration time t_{exp} and the computational time per iteration t_c.

		ASEP		MRF [9]		CF [3]	
Scenario	**r** [m]	$\mathbf{t_c}$ [ms]	$\mathbf{t_{exp}}$[min]	$\mathbf{t_c}$ [ms]	$\mathbf{t_{exp}}$[min]	$\mathbf{t_c}$ [ms]	$\mathbf{t_{exp}}$[min]
Simple warehouse	0.2	(5.00, 3.97)	(1.94, 0.48)	(7.61, 8.31)	(2.02, 0.55)	(8.73, 2.69)	(2.67, 0.67)
	0.1	(10.83, 7.78)	(2.01, 0.52)	(129.92, 47.21)	(2.08, 0.61)	(15.64, 5.65)	(3.01, 0.75)
Complex warehouse	0.2	(5.03, 5.41)	(18.88, 3.78)	(4.39, 25.00)	(20.85, 4.89)	(6.32, 2.51)	(33.78, 5.95)
	0.1	(8.24, 5.24)	(21.74, 3.85)	(1043.44, 163.79)	(22.67, 4.85)	(13.51, 4.75)	(35.14, 6.17)

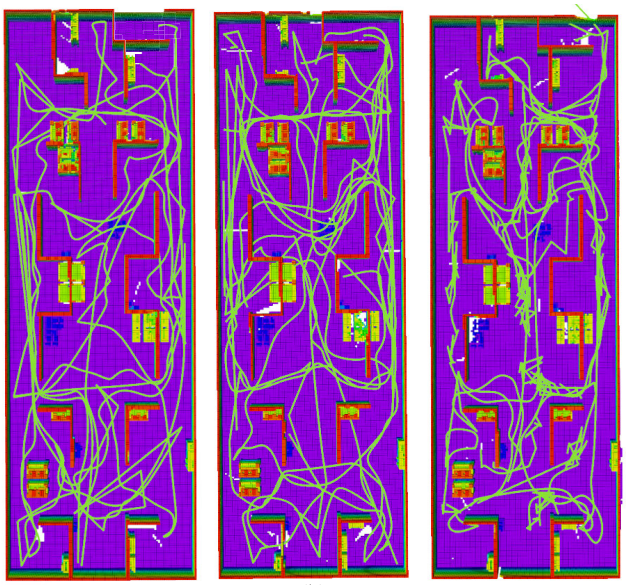

FIGURE 10. The OctoMap of the complex scenario created by the ASEP, MRF and CF, respectively, with planned paths, at the resolution $r = 0.2$ m.

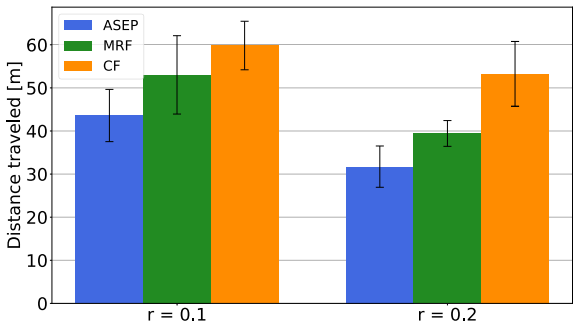

FIGURE 11. Total distance traveled in the simple environment for the ASEP, MRF and CF. Data are given as means of 10 runs with standard deviations.

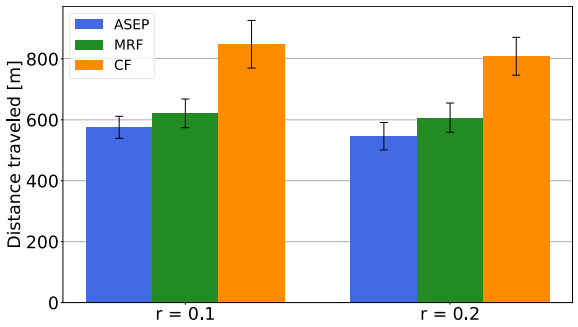

FIGURE 12. Total distance traveled in the complex environment for the ASEP, MRF and CF. Data are given as means of 10 runs with standard deviations.

ASEP and the MRF complete the exploration of the entire area at almost the same time and remarkably faster than the CF. The graph shows that our method and the MRF need around 100 s to explore the simple scenario at different map resolutions, while the CF needs more than 160 s. However, the MRF results in less explored volume, especially at lower resolutions. Namely, at the $r = 0.2$ m it explores around 92% of the environment, while the ASEP and CF explore almost 98%. This occurs because MRF uses multi-resolution clustering at lower OcTree depth $d_{exp} = 15$, leading to the next best frontier selected at d_{exp}, which is shifted from the detected frontier at the deepest level of OcTree, which is 16.

In the complex scenario, the explored volume in time is shown in Fig. 9. Our method explores the entire environment more than twice as fast as the CF and about 5 minutes faster than the MRF. ASEP and MRF behave similarly at the beginning, but over time the MRF and especially the CF show their drawbacks, which affect the total exploration time. The ASEP explores the complex environment in 18.88 min with a standard deviation of 3.78 min and in 21.74 min with a standard deviation of 3.85 min for a resolution of 0.2 m and 0.1 m, respectively. A thorough comparison of the experimental results with sampling-based approaches, such as [4] and [28] is omitted since they use different approaches for frontier generation and selection. Taking these results into consideration, it is shown that combining the semantic information from the environment in the frontier evaluation can result in a faster total exploration, but it definitely results in faster labeling of all given objects in the environment. Please note that the object arrangement influences the total exploration time. For instance, if objects of interest are tightly grouped at a single point in the environment, we expect the UAV to first circle the objects and then move to other parts of the environment. This configuration could increase the overall exploration time.

The OctoMap of the complex scenario generated by all three planners at $r = 0.2$ m is shown in Fig. 10 along with the corresponding UAV paths. Note that some of the paths shown are not fully executed because the collision

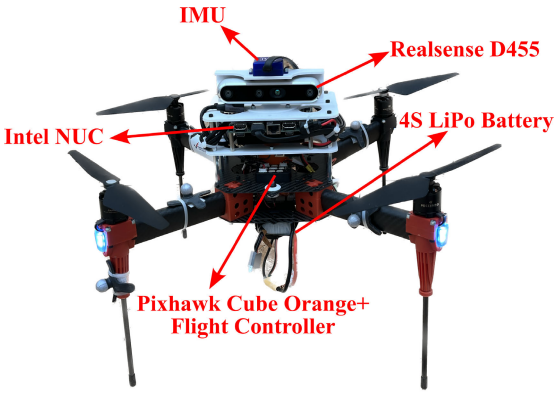

FIGURE 13. A *Hexsoon EDU-450* quadcopter equipped with a *Intel NUC*, a Realsense *D*455 camera, an IMU, a battery and a flight controller.

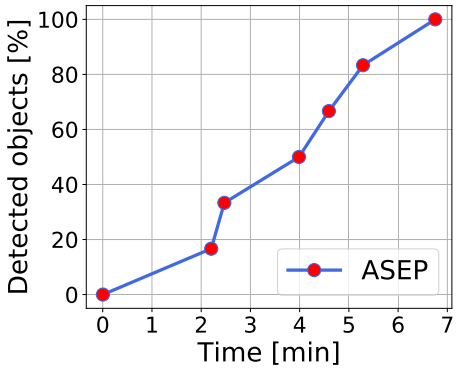

FIGURE 14. The detected objects in time for the real-world environment at $r = 0.1$ m.

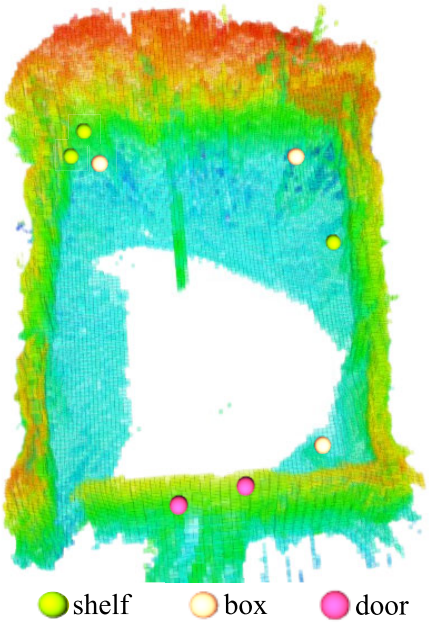

FIGURE 15. The OctoMap of the real-world environment with transparency effect applied on and with detected objects during the exploration of a real-world scenario. Shelves, boxes and doors are shown with different colors on the map.

checker was triggered. In Fig. 11 and Fig. 12, the distance traveled by the UAV is visualized in both simple and complex scenarios at different resolutions. The metric for the distance traveled is derived from the UAV odometry obtained from the simulator. The ASEP achieves the shortest path planned, as well as traveled among the three algorithms, while CF shows a tendency towards larger traveled distances. This was particularly evident in the complex environment at $r = 0.1$ m where the CF recorded an average distance of 847.98 m and the ASEP 575.50 m. The nature of the CF algorithm often results in significant back-and-forth movement, which can lead to a less efficient exploration trajectory (Fig. 10(c)). The distance traveled for the MRF is similar to the ASEP. Namely, for the simple warehouse scenario at $r = 0.1$ m the UAV traveled on average 43.59 m and 53.02 m by ASEP and MRF respectively, while at $r = 0.2$ m, 31.73 m and 39.43 m.

VII. EXPERIMENTAL ANALYSIS

A. SETUP

For our indoor experimental analysis, a *Hexsoon EDU-450* quadcopter is used (Fig. 13) which features four *T-motors* HS2216 920KV motors attached to a carbon fiber frame. The dimensions of the UAV are 0.36 m × 0.36 m × 0.3 m, which makes it a relatively small UAV suitable for indoor environments. The total flight time of the UAV is around 8 min with a mass of m = 2.5 kg, including batteries, electronics and sensory apparatus. The *Cube Orange+* flight controller unit is attached to the center of the UAV body, and it is responsible for the low-level attitude control of the vehicle. Furthermore, the UAV is equipped with an *Intel NUC*, i7-8650U CPU @ 1.90GHz × 8, onboard computer for collecting and processing sensory data. The onboard computer runs *Linux Ubuntu 18.04* with *ROS Melodic* framework that communicates with the autopilot through a serial interface. The UAV is equipped with a Realsense *D*455 camera with a maximum range of 6 m. The parameters used in the real world are stated in Table 1. The experiments are performed in an environment of 10 m × 8 m × 3 m. Experimental evaluations were tested using a voxel resolution of $r = 0.2$ m and $r = 0.1$ m, the same as in the simulation analysis.

B. RESULTS AND DISCUSSION

Running the planner with limited onboard resources and in real-time, the exploration and object labeling are demonstrated. As in the simulation scenario, N_{labels} is set to three, while $N_{obj} = 8$ is detected during exploration. The percentage of detected objects in time is shown in Fig. 14. Fig. 15 shows the labeled objects on the map of the real-world scenario at $r = 0.1$ m.

Computation times t_c and total exploration times t_{exp} for both runs are shown in Table 3. The average computation time is comparable to the time achieved in the simulation

TABLE 3. The tuples of mean and standard deviation for the total exploration time t_{exp} and the computational time per iteration t_c for the real-world scenario.

Scenario	r [m]	t_c [ms]	t_{exp}[min]
Real world	0.2	(3.95, 5.68)	(6.96, 1.08)
	0.1	(24.60, 36.60)	(7.95, 1.52)

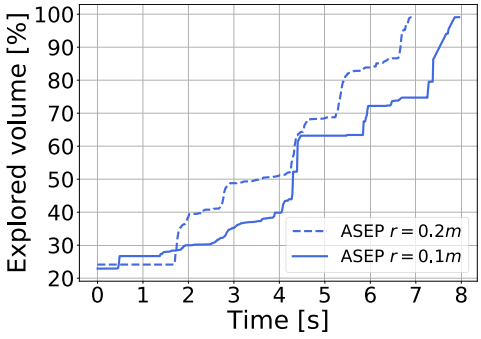

FIGURE 16. The explored volume in time for the real-world scenario.

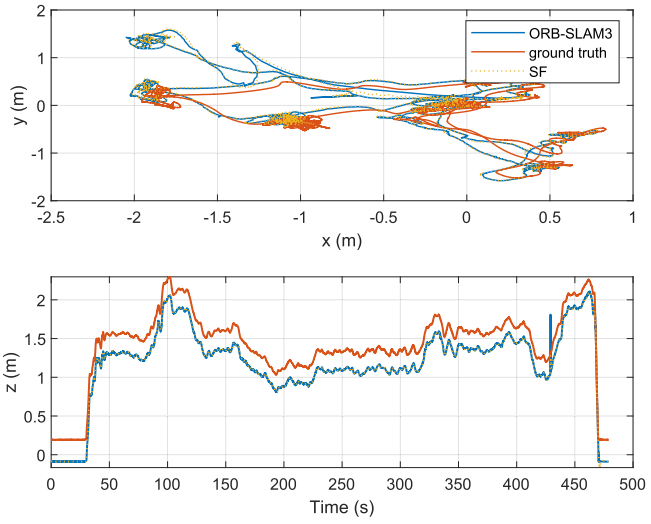

FIGURE 17. Comparison of ORB-SLAM3, multi-sensor fusion and ground truth positions for the x, y, and z coordinates in a single exploration run.

setup. Fig. 16 shows that the total exploration time is about 7 minutes for $r = 0.2$ m and about 8 minutes for $r = 0.1$ m. The result of the exploration is the OctoMap of the environment shown in Fig. 15, in which the objects labeled during the exploration are also shown. Running the planner in the real world and in real time, the successful exploration and object labeling is demonstrated while running the SLAM, semantic segmentation and exploration on the UAV with limited onboard resources.

ORB-SLAM3 algorithm is used for localization during real-world exploration. The positions estimated by the ORB-SLAM3, multi-sensor fusion (SF), and ground truth are shown in Fig. 17. It is noteworthy that the ORB-SLAM3 and SF are very similar and both show a deviation from the ground truth position over time, emphasizing the importance of ongoing calibration. The results highlight the potential of the ORB-SLAM3 algorithm and the multi-sensor fusion approach for reliable localization suitable for exploration.

VIII. CONCLUSION

This paper deals with a novel semantically-enhanced frontier-based exploration planner called ASEP. The ASEP is capable of autonomously exploring a previously unknown GPS-denied area, creating an occupancy grid map OctoMap and labeling objects of interest in the OctoMap. Results show improved behavior in terms of time needed to label all objects in the environment compared to state-of-the-art strategies. An exploration strategy that combines both geometric and semantic information from the environment speeds up the exploration of all objects, while a novel object labeling algorithm ensures real-time object detection and evaluation. This 3D exploration planner has been successfully tested in both simulation scenarios and a real-world experiment using a quadcopter equipped with a camera. Video recordings of the semantically augmented 3D exploration can be found on YouTube [56].

REFERENCES

[1] UNIZG-FER and LARICS. *AeroSTREAM Project*. Accessed: Jun. 6, 2023. [Online]. Available: https://aerostream.fer.hr/aerostream

[2] A. Hornung, K. M. Wurm, M. Bennewitz, C. Stachniss, and W. Burgard, "OctoMap: An efficient probabilistic 3D mapping framework based on octrees," *Auto. Robots*, vol. 34, no. 3, pp. 189–206, Apr. 2013.

[3] B. Yamauchi, "A frontier-based approach for autonomous exploration," in *Proc. Int. Symp. Comput. Intell. Robot. Automat. (CIRA)*, Jul. 1997, pp. 146–151.

[4] A. Bircher, M. Kamel, K. Alexis, H. Oleynikova, and R. Siegwart, "Receding horizon 'next-best-view' planner for 3D exploration," in *Proc. IEEE Int. Conf. Robot. Automat. (ICRA)*, May 2016, pp. 1462–1468.

[5] R. Ashour, T. Taha, J. M. M. Dias, L. Seneviratne, and N. Almoosa, "Exploration for object mapping guided by environmental semantics using UAVs," *Remote Sens.*, vol. 12, no. 5, p. 891, Mar. 2020.

[6] R. P. de Figueiredo, J. le Fevre Sejersen, J. G. Hansen, M. Brandão, and E. Kayacan, "Real-time volumetric-semantic exploration and mapping: An uncertainty-aware approach," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS)*, Sep. 2021, pp. 9064–9070.

[7] B. Yu, H. Kasaei, and M. Cao, "Frontier semantic exploration for visual target navigation," in *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, May 2023, pp. 4099–4105.

[8] T. Dang, C. Papachristos, and K. Alexis, "Visual saliency-aware receding horizon autonomous exploration with application to aerial robotics," in *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, May 2018, pp. 2526–2533.

[9] A. Batinovic, T. Petrovic, A. Ivanovic, F. Petric, and S. Bogdan, "A multi-resolution frontier-based planner for autonomous 3D exploration," *IEEE Robot. Autom. Lett.*, vol. 6, no. 3, pp. 4528–4535, Jul. 2021.

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

[11] A. Bachrach, R. He, and N. Roy, "Autonomous flight in unknown indoor environments," *Int. J. Micro Air Vehicles*, vol. 1, no. 4, pp. 217–228, Dec. 2009.

[12] T. Cieslewski, E. Kaufmann, and D. Scaramuzza, "Rapid exploration with multi-rotors: A frontier selection method for high speed flight," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS)*, Sep. 2017, pp. 2135–2142.

[13] C. Zhu, R. Ding, M. Lin, and Y. Wu, "A 3D frontier-based exploration tool for MAVs," in *Proc. IEEE 27th Int. Conf. Tools Artif. Intell. (ICTAI)*, Nov. 2015, pp. 348–352.

[14] A. Mannucci, S. Nardi, and L. Pallottino, "Autonomous 3D exploration of large areas: A cooperative frontier-based approach," in *Proc. Int. Workshop Modelling Simulation Auton. Syst.*, 2017, pp. 18–39.

[15] A. Dai, S. Papatheodorou, N. Funk, D. Tzoumanikas, and S. Leutenegger, "Fast frontier-based information-driven autonomous exploration with an MAV," in *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, May 2020, pp. 9570–9576.

[16] M. Faria, R. Marín, M. Popović, I. Maza, and A. Viguria, "Efficient lazy Theta* path planning over a sparse grid to explore large 3D volumes with a multirotor UAV," *Sensors*, vol. 19, no. 1, p. 174, Jan. 2019.

[17] B. Zhou, Y. Zhang, X. Chen, and S. Shen, "FUEL: Fast UAV exploration using incremental frontier structure and hierarchical planning," *IEEE Robot. Autom. Lett.*, vol. 6, no. 2, pp. 779–786, Apr. 2021.

[18] B. Fang, J. Ding, and Z. Wang, "Autonomous robotic exploration based on frontier point optimization and multistep path planning," *IEEE Access*, vol. 7, pp. 46104–46113, 2019.

[19] H. H. González-Baños and J.-C. Latombe, "Navigation strategies for exploring indoor environments," *Int. J. Robot. Res.*, vol. 21, nos. 10–11, pp. 829–848, Oct. 2002.

[20] T. Baiming, S. Jicheng, D. Chaofan, and L. Qingbao, "A target point based MAV 3D exploration method," in *Proc. IEEE Int. Conf. Mechatronics Autom. (ICMA)*, Aug. 2018, pp. 2406–2413.

[21] D. Joho, C. Stachniss, P. Pfaff, and W. Burgard, "Autonomous exploration for 3D map learning," in *Autonome Mobile Systeme*. Cham, Switzerland: Springer, 2007, pp. 22–28.

[22] S. M. Lavalle and J. J. Kuffner, "Rapidly-exploring random trees: Progress and prospects," in *Algorithmic and Computational Robotics: New Directions*. Boca Raton, FL, USA: CRC Press, 2000, pp. 293–308.

[23] J. J. Kuffner and S. M. LaValle, "RRT-connect: An efficient approach to single-query path planning," in *Proc. Millennium Conf. IEEE Int. Conf. Robot. Automat. Symposia (ICRA)*, Apr. 2000, pp. 995–1001.

[24] D. Deng, Z. Xu, W. Zhao, and K. Shimada, "Frontier-based automatic-differentiable information gain measure for robotic exploration of unknown 3D environments," 2020, *arXiv:2011.05288*.

[25] C. Witting, M. Fehr, R. Bähnemann, H. Oleynikova, and R. Siegwart, "History-aware autonomous exploration in confined environments using MAVs," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS)*, Oct. 2018, pp. 1–9.

[26] L. Schmid, M. Pantic, R. Khanna, L. Ott, R. Siegwart, and J. Nieto, "An efficient sampling-based method for online informative path planning in unknown environments," *IEEE Robot. Autom. Lett.*, vol. 5, no. 2, pp. 1500–1507, Apr. 2020.

[27] A. Batinovic, A. Ivanovic, T. Petrovic, and S. Bogdan, "A shadowcasting-based Next-Best-View planner for autonomous 3D exploration," *IEEE Robot. Autom. Lett.*, vol. 7, no. 2, pp. 2969–2976, Apr. 2022.

[28] M. Selin, M. Tiger, D. Duberg, F. Heintz, and P. Jensfelt, "Efficient autonomous exploration planning of large-scale 3-D environments," *IEEE Robot. Autom. Lett.*, vol. 4, no. 2, pp. 1699–1706, Apr. 2019.

[29] Z. Meng, H. Qin, Z. Chen, X. Chen, H. Sun, F. Lin, and M. H. Ang, "A two-stage optimized next-view planning framework for 3-D unknown environment exploration, and structural reconstruction," *IEEE Robot. Autom. Lett.*, vol. 2, no. 3, pp. 1680–1687, Jul. 2017.

[30] V. M. Respall, D. Devitt, R. Fedorenko, and A. Klimchik, "Fast sampling-based Next-Best-View exploration algorithm for a MAV," in *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, May 2021, pp. 89–95.

[31] C. Wang, D. Zhu, T. Li, M. Q.-H. Meng, and C. W. de Silva, "Efficient autonomous robotic exploration with semantic road map in indoor environments," *IEEE Robot. Autom. Lett.*, vol. 4, no. 3, pp. 2989–2996, Jul. 2019.

[32] C. Gomez, M. Fehr, A. Millane, A. C. Hernandez, J. Nieto, R. Barber, and R. Siegwart, "Hybrid topological and 3D dense mapping through autonomous exploration for large indoor environments," in *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, May 2020, pp. 9673–9679.

[33] Y. Wang, D. Ewert, D. Schilberg, and S. Jeschke, "Edge extraction by merging the 3D point cloud and 2D image data," in *Automation, Communication and Cybernetics in Science and Engineering 2013/2014*. Cham, Switzerland: Springer, 2014, pp. 773–785.

[34] E. P. H. Alarcón, D. B. Ghavifekr, G. Baris, M. Mugnai, M. Satler, and C. A. Avizzano, "An efficient object-oriented exploration algorithm for unmanned aerial vehicles," in *Proc. Int. Conf. Unmanned Aircr. Syst. (ICUAS)*, Jun. 2021, pp. 330–337.

[35] K. Ota, Y. Sasaki, D. K. Jha, Y. Yoshiyasu, and A. Kanezaki, "Efficient exploration in constrained environments with goal-oriented reference path," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS)*, Oct. 2020, pp. 6061–6068.

[36] L. Campos-Macías, R. Aldana-López, R. de la Guardia, J. I. Parra-Vilchis, and D. Gómez-Gutiérrez, "Autonomous navigation of MAVs in unknown cluttered environments," *J. Field Robot.*, vol. 38, no. 2, pp. 307–326, May 2020.

[37] W. Kwon, J. H. Park, M. Lee, J. Her, S.-H. Kim, and J.-W. Seo, "Robust autonomous navigation of unmanned aerial vehicles (UAVs) for warehouses' inventory application," *IEEE Robot. Autom. Lett.*, vol. 5, no. 1, pp. 243–249, Jan. 2020.

[38] C. Campos, R. Elvira, J. J. G. Rodríguez, J. M. M. Montiel, and J. D. Tardós, "ORB-SLAM3: An accurate open-source library for visual, visual-inertial, and multimap SLAM," *IEEE Trans. Robot.*, vol. 37, no. 6, pp. 1874–1890, Dec. 2021.

[39] W. Hess, D. Kohler, H. Rapp, and D. Andor, "Real-time loop closure in 2D LiDAR SLAM," in *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, May 2016, pp. 1271–1278.

[40] R. Mur-Artal and J. D. Tardós, "ORB-SLAM2: An open-source SLAM system for monocular, stereo, and RGB-D cameras," *IEEE Trans. Robot.*, vol. 33, no. 5, pp. 1255–1262, Oct. 2017.

[41] J. Sturm, N. Engelhard, F. Endres, W. Burgard, and D. Cremers, "A benchmark for the evaluation of RGB-D SLAM systems," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, Oct. 2012, pp. 573–580.

[42] L. Markovic, M. Kovac, R. Milijas, M. Car, and S. Bogdan, "Error state extended Kalman filter multi-sensor fusion for unmanned aerial vehicle localization in GPS and magnetometer denied indoor environments," in *Proc. Int. Conf. Unmanned Aircr. Syst. (ICUAS)*, Jun. 2022, pp. 184–190.

[43] K. Sun, B. Xiao, D. Liu, and J. Wang, "Deep high-resolution representation learning for human pose estimation," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2019, pp. 5686–5696.

[44] J. Wang, K. Sun, T. Cheng, B. Jiang, C. Deng, Y. Zhao, D. Liu, Y. Mu, M. Tan, X. Wang, W. Liu, and B. Xiao, "Deep high-resolution representation learning for visual recognition," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 43, no. 10, pp. 3349–3364, Oct. 2021.

[45] *Semantic Segmentation on MIT ADE20K Dataset in PyTorch*. Accessed: Feb. 21, 2023. [Online]. Available: https://github.com/CSAILVision/semantic-segmentation-pytorch

[46] B. Zhou, H. Zhao, X. Puig, T. Xiao, S. Fidler, A. Barriuso, and A. Torralba, "Semantic understanding of scenes through the ADE20K dataset," *Int. J. Comput. Vis.*, vol. 127, no. 3, pp. 302–321, Mar. 2019.

[47] B. Zhou, H. Zhao, X. Puig, S. Fidler, A. Barriuso, and A. Torralba, "Scene parsing through ADE20K dataset," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jul. 2017, pp. 5122–5130.

[48] T.-Y. Lin, M. Maire, S. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dollár, and C. L. Zitnick, "Microsoft COCO: Common objects in context," in *Proc. Eur. Conf. Comput. Vis.*, 2014, pp. 740–755.

[49] M. Everingham, L. Van Gool, C. K. I. Williams, J. Winn, and A. Zisserman, "The Pascal visual object classes (VOC) challenge," *Int. J. Comput. Vis.*, vol. 88, no. 2, pp. 303–338, Jun. 2010.

[50] M. Cordts et al., "The cityscapes dataset for semantic urban scene understanding," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Las Vegas, NV, USA, 2016, pp. 3213–3223, doi: 10.1109/CVPR.2016.350.

[51] B. Arbanas, A. Ivanovic, M. Car, M. Orsag, T. Petrovic, and S. Bogdan, "Decentralized planning and control for UAV-UGV cooperative teams," *Auto. Robots*, vol. 42, no. 8, pp. 1601–1618, Dec. 2018.

[52] A. Ivanovic and M. Orsag, "Parabolic airdrop trajectory planning for multirotor unmanned aerial vehicles," *IEEE Access*, vol. 10, pp. 36907–36923, 2022.

[53] UNIZG-FER and LARICS. *Path and Trajectory Planning*. Accessed: Jun. 6, 2023. [Online]. Available: https://github.com/larics/larics_motion_planning

[54] T. Baca, D. Hert, G. Loianno, M. Saska, and V. Kumar, "Model predictive trajectory tracking and collision avoidance for reliable outdoor deployment of unmanned aerial vehicles," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS)*, Oct. 2018, pp. 6753–6760.

[55] A. Batinovic, J. Goricanec, L. Markovic, and S. Bogdan, "Path planning with potential field-based obstacle avoidance in a 3D environment by an unmanned aerial vehicle," in *Proc. Int. Conf. Unmanned Aircr. Syst. (ICUAS)*, Jun. 2022, pp. 394–401.

[56] *ASEP: An Autonomous Semantic Exploration Planner with Object Labeling*. Accessed: Aug. 10, 2023. [Online]. Available: https://www.youtube.com/playlist?list=PLC0C6uwoEQ8ZHLRJLCw2YEaQE4aR-Y65f

ANA MILAS (Member, IEEE) received the B.S.E.E. and M.S.E.E. degrees from the University of Zagreb, Croatia, in 2016 and 2018, respectively, where she is currently pursuing the Ph.D. degree. She is also a Research Associate with the Faculty of Electrical Engineering and Computing, University of Zagreb. In 2019, she did an internship with the University of Seville, Spain. In 2020, she attended the IEEE RAS Summer School on Multi-Robot Systems, Prague, Czech Republic. As a Ph.D. student, she participated in a Mohamed Bin Zayed International Robotics Challenge (MBZIRC2020) and MBZIRC2023 Project. Her research interests include aerial robotics, autonomous mapping, and exploration. During the bachelor's degree, she received the University of Zagreb Rector Award for the work titled ''Coordinated Multi-Robot Exploration Based on Graph SLAM Method and Rapidly Exploring Random Trees,'' in 2018.

ANTUN IVANOVIC (Student Member, IEEE) received the M.S.E.E. and Ph.D. degrees from the University of Zagreb, Croatia, in 2015 and 2023, respectively.

He is currently a Postdoctoral Researcher with the Laboratory for Robotics and Intelligent Control Systems, University of Zagreb Faculty of Electrical Engineering and Computing (UNIZG-FER), where he joined, in 2015. During the bachelor's degree, he received the Rector's Award for the work titled "Augmented Human Machine Interface for Aerial Manipulators." His research interests include robotics, unmanned aerial vehicles, aerial manipulation, and motion planning. As a Ph.D. student, he participated in a European Robotics Challenge (EuRoC) and MBZIRC2020 (Mohamed Bin Zayed International Robotics Challenge) competitions, and multiple international projects. He is also working on several EU or nationally-funded projects, including Specularia, VirtualUAV, and ASAP. In 2018, he was a Visiting Researcher with the United States Military Academy West Point, USA, where he collaborated on work related to aerial-ground cooperative manipulation.

TAMARA PETROVIC (Member, IEEE) received the master's degree from the University of Zagreb Faculty of Electrical Engineering and Computing (UNIZG-FER), in 2007. Her Ph.D. dissertation was titled ''Centralized Control of Variable Structure Multi-Vehicle Systems Based on Resource Allocation,'' in 2014.

Since 2008, she has been with UNIZG-FER, LARICS, as a Research Fellow, and since 2014, she has been a postgraduate. Her research interests include multi-robot systems and discrete event systems. She has participated in the work of several international and domestic scientific research projects. During her studies, she was a recipient of the bronze plaque ''Josip Lonar,'' in 2007, and three ''Josip Lonar'' awards, in 2003, 2004, and 2005, for outstanding academic success. She published two book chapters, six journal articles, and 17 conference papers. She participates as a reviewer in several international scientific journals and conferences. She participates in teaching activities with the Department of Control and Computer Engineering, which include exercises and lectures, co-mentoring of students, and improvement of teaching content and materials. In 2016 and 2017, she was carried out the duties of the Faculty Erasmus+ Coordinator for traineeships and the mobility of control engineering and automation students.