Voxel Map to Occupancy Map Conversion Using Free Space Projection for Efficient Map Representation for Aerial and Ground Robots

Scott Fredriksson, *Graduate Student Member, IEEE*, Akshit Saradagi, and George Nikolakopoulos, *Member, IEEE*

***Abstract*—This article introduces a novel method for converting 3D voxel maps, commonly utilized by robots for localization and navigation, into 2D occupancy maps for both autonomous aerial vehicles (AAVs) and autonomous ground vehicles (AGVs). The generated 2D maps can be used for more efficient global navigation for both AAVs and AGVs, in enabling algorithms developed for 2D maps to be useful in 3D applications, and allowing for faster transfer of maps between multiple agents in bandwidth-limited scenarios. During the 3D to 2D map conversion, the method conducts safety checks with respect to the robot's safety margins. This ensures that an aerial or ground robot can navigate safely, relying primarily on the 2D map generated by the method. Additionally, the method extracts the height of navigable free space and a local estimate of the slope of the floor from the 3D voxel map. The height data is utilized in converting paths generated using the 2D map into paths in 3D space for both AAVs and AGVs. The slope data identifies areas too steep for a ground robot to traverse, marking them as occupied, thus enabling a more accurate representation of the terrain for ground robots. The proposed method is compared to the existing state-of-the-art fixed projection method in two different environments, over static maps and with progressively expanding maps. The methods proposed in this article have been implemented in the widely-used robotics frameworks ROS and ROS2, and are open-sourced.**

***Index Terms*—Mapping, motion and path planning, field robots.**

I. Introduction

A MAP is a fundamental component of robotics, serving key roles in localization, path planning, and obstacle avoidance. The most common form of map representation is the metric grid-based map in either 2D or 3D formats. Both 3D and 2D grid-based maps have their respective advantages: 3D maps are more accurate representations of the world and offer greater detail and precision, while 2D maps offer the advantage in terms of performance and minimality, particularly with respect to computational efficiency and memory usage. Therefore, utilizing both representations can be beneficial [1] for path planning in autonomous robots. A common approach involves creating and managing two separate 2D and 3D mapping solutions by either converting 3D scans used in 3D mapping to 2D scans for creating the 2D map [2], [3], [4] or using multiple sensors [1], [5]. However, this leads to the challenge of managing two different mapping frameworks on a single robot, potentially resulting in mismatched maps.

Another solution is to use a custom mapping solution that creates maps from raw sensor data. In the past, multiple studies have explored the conversion of point clouds generated by Kinect cameras [6], [7], [8] into 2D maps. However, such methods are unsuitable for handling point clouds containing ceilings or overhangs. The method in [9] generates a 2D map using keyframes from ORB-slam [10], while the work [11] utilized down projection to convert a 3D point cloud into a 2D map after filtering out the portion of the point cloud that represents the floor. Another solution is 2.5D mapping [12], [13], [14], which focuses on mapping only surfaces that a ground robot can traverse. A problem with such methods is that they are often application- or sensor-specific and are not compatible with methods, such as map segmentation and path planning, designed specifically for voxel maps.

In the robotics literature, there are very few solutions for effectively converting 3D voxel maps into 2D maps. The most prevalent technique involves projecting a part of the voxel map at a fixed height downward onto a 2D plane, a process implemented in the widely-used OctoMap [15] library within ROS (Robot Operating System). Another example is the hybrid height voxel mapper, HMAPs [12], which allows for the conversion from 3D to 2.5D and 2D maps using the same downward projection technique. Such approaches have several limitations. Specifically, the projected part of the map can not include any voxels from the ground or ceiling, as these would also be projected onto the 2D map. Similarly, any obstacle in the map not within the selected range will be omitted from the projection. Generally, this is not a problem in smaller, controlled environments, but in larger environments with varying heights of navigable space, with multiple height levels, or where there is drift in the robot's positioning along the vertical axis, this method becomes less effective.

Motivated by the lack of methods in existing literature for converting 3D voxel maps to 2D maps that can handle varying floor heights, this article proposes a more utility-focused

Received 19 July 2024; accepted 26 October 2024. Date of publication 11 November 2024; date of current version 19 November 2024. This article was recommended for publication by Associate Editor T. Do and Editor A. Bera upon evaluation of the reviewers' comments. *(Corresponding author: Scott Fredriksson.)*

The authors are with the Robotics and AI group, Department of Computer Science, Electrical and Space Engineering, Luleå University of Technology, SE-97187 Luleå, Sweden (e-mail: scofre@ltu.se).

The code is available at: https://github.com/LTU-RAI/Map-Conversion-3D-Voxel-Map-to-2D-Occupancy-Map.

Digital Object Identifier 10.1109/LRA.2024.3495575

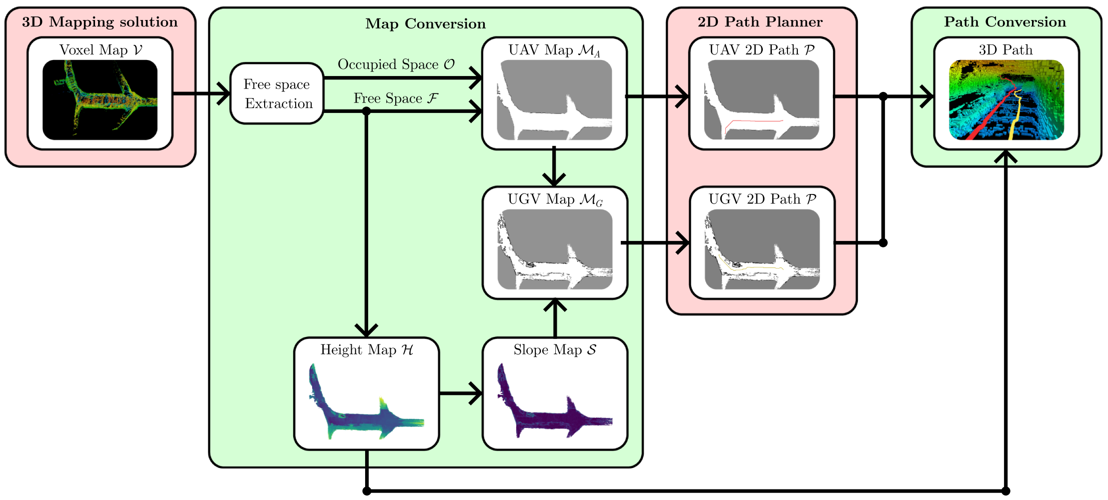


Fig. 1. An overview of the proposed methodology for 3D Voxel map to 2D map conversion, along with path planning for the AAVs and AGVs, and 2D to 3D path conversion.

method for converting 3D voxel maps into 2D maps, to enable efficient path planning for autonomous aerial vehicles (AAVs) and autonomous ground vehicles (AGVs), thus allowing robots to utilize the distinct mapping advantages of both 2D and 3D maps.

II. Contributions

The article's primary contribution is a novel method for converting a 3D voxel map into a 2D occupancy map. Instead of projecting a fixed section of the voxel map, as in existing methods, the proposed method projects free space, leading to a more robust and flexible approach in environments with varying elevations. The method also extracts the height values of the ceiling and floor of the environment and a local estimate of the slope of the floor. The proposed method generates two 2D maps: i) an obstacle-free map that is useful for computation-friendly AAV path planning, by taking into account the height of the floor and ceiling from the height map, and ii) a map for AGVs that incorporates walls and obstacles using slope estimation, which again enables more computation-friendly path planning for AGVs in comparison with 3D Voxel maps. The second contribution is the proposal of a method to convert 2D paths generated using 2D occupancy maps into 3D paths for AAVs and AGVs using the height map. This allows an aerial or ground robot to navigate safely, relying primarily on the 2D map generated by the method. These methods have been implemented in ROS and ROS2, and open sourced.[1] The proposed method is compatible with the UFOMap [16] and the OctoMap [15] mapping frameworks. However, the method can be implemented for any voxel-based mapping solution that represents space using occupied, free, and unknown voxel cells. The rest of the article is structured as follows. Section III describes the proposed methodology for 3D to 2D map conversion. Section IV demonstrates the implementation of the methodology for converting 2D paths into 3D paths using the height map generated by the map conversion process. Section V discusses the utility of the 3D to 2D map conversions in robotics applications. Section VI presents the validation of the overall methodology on maps from two different environments. Section VII contains a discussion of the results. Finally, Section VIII presents conclusions, with a discussion and summary of the findings.

[1]Source code is available at https://github.com/LTU-RAI/Map-Conversion-3D-Voxel-Map-to-2D-Occupancy-Map

III. 3D to 2D Map Conversion

An overview of the proposed methodology for 3D Voxel map to 2D map conversion is presented in Fig. 1. Using the 3D voxel map $\mathcal{V}$, four 2D grid-based maps are generated. Firstly, a height map $\mathcal{H}$ is created, where every cell of a 2D grid map contains height data of the navigable free space (ceiling and floor) relative to the origin of the Voxel map. Secondly, a slope map $\mathcal{S}$ is generated from the height map (floor height) by estimating the slope of the floor plane in the immediate neighborhood of every cell. Finally, two occupancy maps $\mathcal{M}_A$ and $\mathcal{M}_G$ are produced, one for the AAVs and the other for AGVs respectively, assigning values: -1 for unknown cells, and a value in the range [0,1] for the occupancy probability, with 0 being free, and 1 being occupied. All 2D maps are generated over a 2D grid $\mathcal{G}_{2D}$ with the same x and y size/extent and resolution $\mathcal{V}_{res}$ as the Voxel map being converted. Details about the individual steps in the methodology are presented next.

A. Extraction of Free Space and Height Map

Most robots have their LiDAR mounted vertically, resulting in an incomplete view of the floor. This configuration leads to numerous gaps in the mapped floor area, as can be seen in Figs. 3(e) and 4(e). To address this issue, instead of using occupied voxels, the free space in the map is utilized to identify the floor in $\mathcal{V}$.

1) Free and Occupied Space: To identify the floor, the map $\mathcal{V}$ is first represented as a union of free ($\mathcal{F}$) and occupied ($\mathcal{O}$) ranges on every cell $\{x_m, y_n\}$ in the 2D grid $\mathcal{G}_{2D}$ of size $M \times N$. The free space in the 3D map is represented

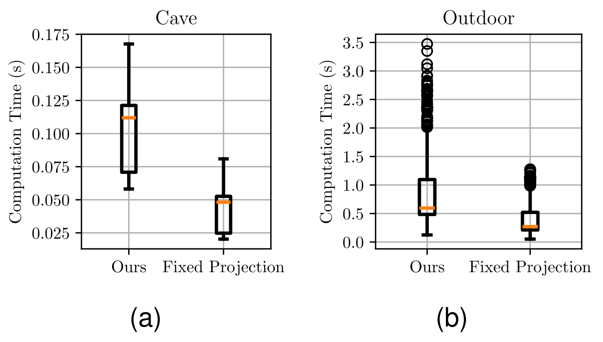


Fig. 2. The computation time to update the 2D map generated from the voxel map under two exploration missions.

by the set $\mathcal{F} = \{F_{x_m,y_n}\}_{M\times N}$, with the element $F_{x_m,y_n} = \{f_1, f_2, \ldots, f_{k_{mn}}\}$, where each $f_i \in F_{x_m,y_n}$ is a range of free space along the height axis, at 2D grid location $\{x_m, y_n\}$ and k_{mn} is the number of such free space ranges. Each range of free space f_i contains two values, $f_i = \{\hat{f}_i, \check{f}_i\}$, where $\hat{f}_i$ and $\check{f}_i$ are the higher and lower bounds of the height range respectively. Similarly, the occupied space is represented as $\mathcal{O} = \{O_{x_m,y_n}\}_{M\times N}$ where $O_{x_m,y_n} = \{o_1, o_2, \ldots, o_{l_{ij}}\}$ is the set of occupied ranges $o_i = \{\hat{o}_i, \check{o}_i\}$. The ranges in the set F_{x_m,y_n} and in the set O_{x_m,y_n} cannot overlap or be connected with another range in the same set. Since a 2D path planning algorithm utilizing the 2D maps generated in this work cannot perform collision checks on the height axis (the Z axis), the proposed method disregards any free range $f_i \in \mathcal{F}$ with a height less than the safety margin R_{maxZ}, which is dependent on the dimensions of the robot. In other words, an f_i is removed if $|f_i| = (\hat{f}_i - \check{f}_i) < R_{maxZ}$, where $|f_i|$ denotes the height of the range.

2) Height Map: Given the free space ranges in $\mathcal{F}$, the height map $\mathcal{H} = \{h_{x_m,y_n}\}_{M\times N}$ is built as a collection of height ranges h_{x_m,y_n}, one for each cell $\{x_m, y_n\}$. An element of the height map $h_{x_m,y_n} = \{\hat{h}_{x_m,y_n}, \check{h}_{x_m,y_n}\}$, where the bottom of the height range $\check{h}_{x_m,y_n}$ is the height of the floor and $\hat{h}_{x_m,y_n}$ is the height of the ceiling with respect to the origin of the voxel map. The floor $\check{h}_{x_m,y_n}$ of a cell h_{x_m,y_n} is the bottom of $f_1 \in F_{x_m,y_n}$, and the ceiling $\hat{h}_{x_m,y_n}$ is the top of $f_{k_{mn}} \in F_{x_m,y_n}$. The above formulation assumes only one level of navigable space in the map in the height axis, as in a building with just one floor or a subterranean environment with no overlapping tunnels.

B. Derivation of the Slope Map Through Least Squares

The slope map, denoted as $\mathcal{S} = \{s_{x_m,y_n}\}_{M\times N}$ is a 2D grid where each cell s_{x_m,y_n} contains an estimate of the slope of the floor in a neighborhood of the cell s_{x_m,y_n}, which is the slope of the plane fitted using the floor height $\check{h}_{x_i,y_j}$ and position $\{x_i, y_j\}$ of the cells neighboring the position $\{x_m, y_n\}$ in the grid. The cells h_{x_i,y_j} that satisfy the condition $\max(|x_m - x_i|, |y_n - y_j|) \leq S_A$ are considered when calculating the slope of the cell s_{x_m,y_n}. $S_A \in \mathbb{N}_{\geq 1}$ is a tunable parameter for the size of the neighborhood considered when calculating the slope. To fit a plane at the cell s_{x_m,y_n}, the linear equation of the form

$$ax + by + c = z \tag{1}$$

is considered, where the parameters a, b and c are real numbers. The interest here is in finding the slope of a plane that is a best fit for the height and position of the cells neighboring the cell s_{x_m,y_n}, that is

$$\underbrace{\begin{bmatrix} x_{i_1} & y_{j_1} & 1 \\ x_{i_2} & y_{j_2} & 1 \\ \vdots & \vdots & \vdots \end{bmatrix}}_{A} \begin{bmatrix} a \\ b \\ c \end{bmatrix} = \underbrace{\begin{bmatrix} \check{h}_{x_{i_1},y_{j_1}} \\ \check{h}_{x_{i_2},y_{j_2}} \\ \vdots \end{bmatrix}}_{B}. \tag{2}$$

The parameters of the best-fit plane are found using the least squares method, given by

$$\begin{bmatrix} a_{ls} \\ b_{ls} \\ c_{ls} \end{bmatrix} = (A^T A)^{-1} A^T B. \tag{3}$$

Given (1), the slope at cell s_{x_m,y_n} is given by

$$s_{x_m,y_n} = \sqrt{a_{ls}^2 + b_{ls}^2}, \tag{4}$$

where a_{ls} is the slope along the x axis and b_{ls} is the slope along the y axis.

C. Generation of 2D Occupancy Maps for AAVs and AGVs

In this subsection, two occupancy maps $\mathcal{M} = \{m_{x_i,y_j}\}_{M\times N}$ are constructed: one for AAVs ($\mathcal{M}_A$) and the other for AGVs ($\mathcal{M}_G$).

1) AAV Map $\mathcal{M}_A$: The map generation begins by representing all cells of the 2D occupancy map with free space identified in Section III-A as free, i.e., $m_{x_i,y_j} = 0$, when $F_{x_i,y_j} \neq \{\varnothing\}$ i.e, when the set F_{x_i,y_j} is non-empty. The remaining cells m_{x_i,y_j} that are not free, but have at least one neighboring cell that is free are examined using the following criteria to decide their occupancy probabilities:

$$m_{x_i,y_j} = \begin{cases} O_{cc}(x_i,y_j) & \text{if } O_{cc}(x_i,y_j) \geq O_{min} \\ -1 & \text{otherwise} \end{cases} \tag{5}$$

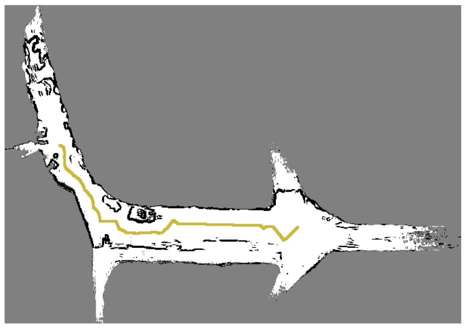
where $O_{cc}(x_i, y_j)$ is the occupancy value in the range 0 and 1 for the examined map cell m_{x_i,y_j} computed using (6) and O_{min} is the minimum allowed occupancy value. If $O_{cc}(x_i, y_j) < O_{min}$, the cell's occupancy is deemed unknown and the value -1 is assigned. The function $O_{cc}(x_i, y_j)$, which calculates the overlapping percentage between the occupied ranges of O_{x_i,y_j} and the neighboring free space in the height map $\mathcal{H}$, is defined as

$$O_{cc}(x_i,y_j) = \max_{x_m,y_n \in N(x_i,y_j)} \left(\frac{\sum_{o_i \in O_{x_i,y_j}} k(h_{x_m,y_n}, o_i)}{||h_{x_m,y_n}||} \right), \tag{6}$$

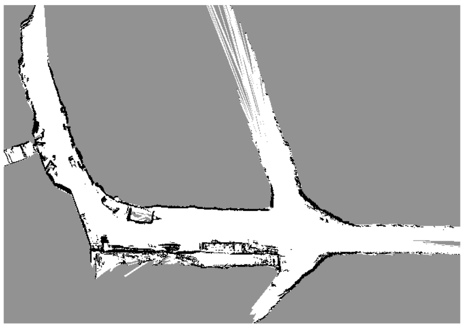
where the function $k(h_{x_m,y_n}, o_i)$ defined as:

$$k(h_{x_m,y_n}, o_i) = \max(0, \min(\hat{h}_{x_m,y_n}, \hat{o}_i) - \max(\check{h}_{x_m,y_n}, \check{o}_i)) \tag{7}$$

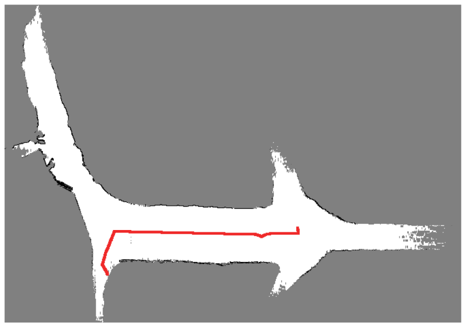
returns the length of the overlap between the two ranges, and when there is no overlap, the function returns 0. If the examined cell has multiple neighboring free cells, only the free cell that results in the maximum occupancy is used. In (6), $N(x_i, y_j)$ is a set of neighboring cells $\{x_r, y_s\}$ with $m_{x_r,y_s} = 0$, i.e., the cells



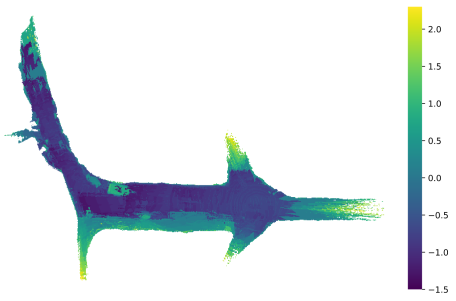
(a) UGV map and path plan for a UGV



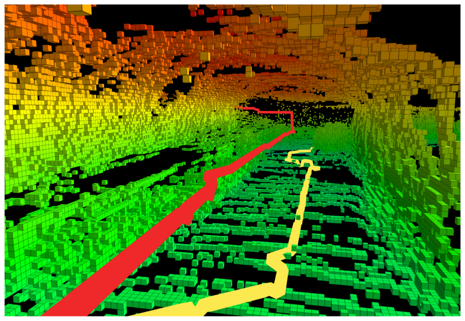
(b) Map generated by fixed projection method



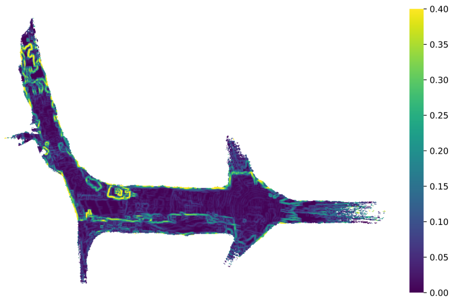
(c) UAV map and path plan for a UAV



(d) Height map



(e) Voxel map with 3D paths for UAV (red) and UGV (yellow)



(f) Slope map

Fig. 3. 2D maps and paths generated by the proposed methods for AAV and AGV as well as the map generated by the fixed projection method in the cave environment. The results from the 2D path to 3D path conversion are shown in subfigures (e) and (f).

containing free navigable range that satisfy the conditions in the following equation:

$$N(x_i, y_j) = \{\{x_r, y_s\} | \max(|x_r - x_i|, |y_s - y_j|) = 1, m_{x_r,y_s} = 0\}. \quad (8)$$

Note that low walls and scattered objects are not relevant for aerial robot navigation, as there is navigable space for aerial robots above the walls and scattered objects. The approach proposed for the generation of $\mathcal{M}_A$ is concerned with the detection of free space for aerial navigation and ignores such entities for the aerial robot maps.

2) AGV Map $\mathcal{M}_G$: The 2D map generation for ground-based robots utilizes the same free space and occupancy detection as the map generated for the AAV, but in addition, the slope map generated in Section III-B is used as well. Cells near low walls and scattered objects exhibit increased slope values as the height increases suddenly. In the AGV map $\mathcal{M}_G$, if the slope associated with a cell that is free is greater than R_{MS}, the cell is instead considered occupied, where R_{MS} is the maximum slope that a ground robot can traverse. Thus, the low walls and scattered objects that are relevant for AGVs are detected and represented in the AGV map $\mathcal{M}_G$.

IV. 2D to 3D Path Conversion

As discussed in Section I, one of the primary advantages of the proposed method is its ability to plan paths using a 2D map instead of relying on a full 3D voxel map. Let us assume that a classical or state-of-the-art 2D path planner is used to find paths

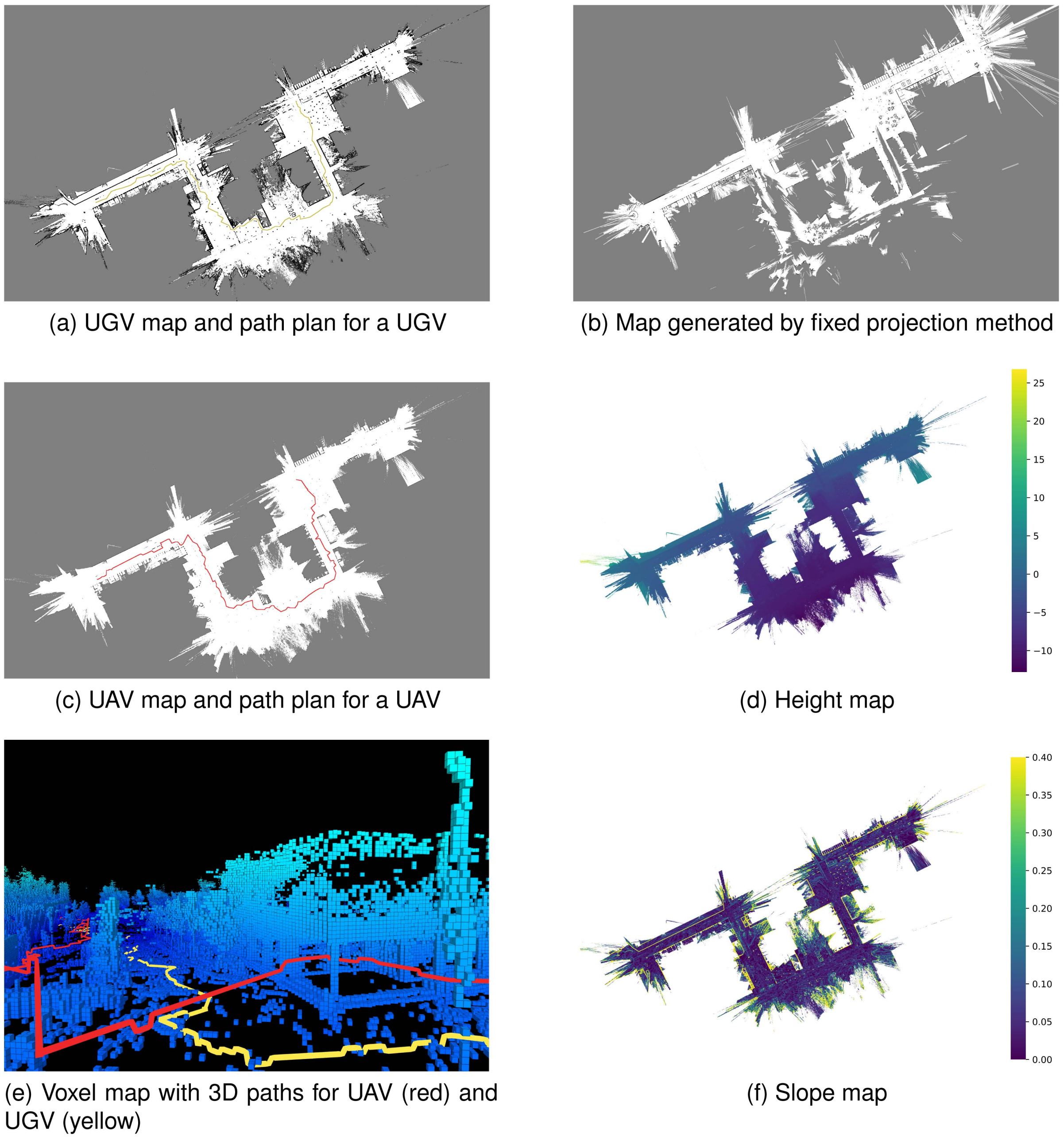

(a) UGV map and path plan for a UGV
(b) Map generated by fixed projection method
(c) UAV map and path plan for a UAV
(d) Height map
25
20
15
10
5
0
−5
−10
(e) Voxel map with 3D paths for UAV (red) and UGV (yellow)
(f) Slope map
0.40
0.35
0.30
0.25
0.20
0.15
0.10
0.05
0.00

Fig. 4. 2D maps and paths generated by the proposed methods for AAV and AGV as well as the map generated by the fixed projection method in the cave environment. The results from the 2D path to 3D path conversion are shown in subfigures (e) and (f).

in the AAV and AGV maps. To make the 2D paths practicable for robots, especially AAVs, it is essential to determine the height, the Z position, of each point along the 2D path. To address this challenge, for the path $\mathcal{P}$ generated by the 2D path planner, which is a sequence of positions $\mathcal{P}_i$, the following process is employed to determine the Z position of each point $\mathcal{P}_i$ in the path.

To determine the height $\mathcal{P}_{iZ}$ of each point $\mathcal{P}_i$ on the path $\mathcal{P}$, the following equation is used:

$$\mathcal{P}_{iZ} = \max\left(\check{\mathcal{H}}(\mathcal{P}_{i-p_f}), \ldots, \check{\mathcal{H}}(\mathcal{P}_{i+p_f})\right) + R_{off} \tag{9}$$

where $\check{\mathcal{H}}(\mathcal{P}_i)$ represents the floor height in the height map $\mathcal{H}$ at the position of $\mathcal{P}_i$. The parameter p_f is user-defined, specifying the number of steps that are considered along the path in front of and behind the current position of the robot. The parameter R_{off} is another user-defined parameter that sets a desired safe height above the ground for the path. The use of the $\max$ function ensures that the path proactively adjusts its height p_f steps before encountering an obstacle. For AAVs, considering just the points along the path for height computation may prove insufficient to ensure safe distances from the objects in the neighborhood of the path. Therefore, when generating 3D paths for the AAVs, an

TABLE I
PARAMETERS USED IN THE VALIDATION CASES

	R_{maxZ}	O_{min}	S_A	R_{MS}	R_r	R_{off}	p_f
UAV	1 m	0.5	NA	2	0.5 m	1 m	2 m / $\mathcal{V}_{res}$
UGV	1 m	0.5	2	2	NA	0.1 m	0.5 m / $\mathcal{V}_{res}$

TABLE II
COMPUTATION TIME FOR MAP CONVERSION FOR THE DIFFERENT METHODS

	Full map time		Partial map average time	
	Cave	Outdoor	Cave	Outdoor
Ours	0.23 s	6.07 s	0.10 s	0.83 s
Fixed Projection	0.094 s	2.84 s	0.042 s	0.38 s

additional check is performed to avoid collisions with a sphere-shaped safety region around the AAVs. The safety region has a radius of $0 < R_r \leq R_{maxZ/}2$. If the height calculated in the previous step results in a collision with either the floor or the ceiling in the height map $\mathcal{H}$, the point in the path that results in a collision is moved to a Z position that does not cause a collision, ignoring R_{off} at that point in the path.

V. USE CASES AND UTILITY OF THE PROPOSED METHOD

The authors identify several use cases for the 3D to 2D map conversion method proposed in this work. To begin with, a 2D map on a computer screen can be more intuitive for a human compared to a 3D map, to both visualize and interact with, for example, in selecting goal points or monitoring the navigation of a robot. The second use case arises in using a 2D map for global navigation for AAVs. Path planning using the 2D map is more efficient with respect to memory and CPU usage, as the navigation problems are quicker to solve since the Z-axis need not be considered. The third utility of the proposed methods is in enabling the use of methods (such as map segmentation or several 2D navigation solutions) that are designed specifically for 2D maps or scale poorly for a 3D map, to be applicable in 3D scenarios.

Map regions, objects, and paths found using the 2D maps can then be moved to 3D space using the height information of the environment extracted by the proposed method. This can be done using the method presented in Section IV, which converts 2D paths planned on a 2D map to 3D paths for both AGVs and AAVs.

The final utility arises in scenarios where multiple robotic agents need to communicate and exchange large maps with each other. Since the 2D map has a smaller file size than the full 3D Voxel map (as shown in Table III in Section VI), it can be communicated between robots or to a remote location more rapidly within the allocated bandwidth. Although the 2D map and height map do not perfectly represent the environment, they still offer a good practical sense of the layout, enabling the robots to perform general path planning.

VI. VALIDATION

The proposed method was validated using two 3D voxel maps of real environments gathered from robotic experiments: a small section of a cave and a large outdoor environment. A voxel map using the UFOMaps [16] was generated with a resolution of $\mathcal{V}_{res} = 0.1$ m in the cave environment and with a resolution of $\mathcal{V}_{res} = 0.2$ in the outdoor environment. Using the map conversion method outlined in Section III, 2D occupancy maps $\mathcal{M}_A$ and $\mathcal{M}_G$ were generated from voxel maps of the two environments.

The proposed method was compared with the state-of-the-art fixed projection technique for voxel-to-2D map conversion, that employs regular down-projection within a fixed height range, as used in [12], [15]. The height range selected for fixed projection was from -0.3m to 0.7m, below and above the origin point in the map. Note that the origin point corresponds to the starting position of the LiDAR, which is mounted on top of a robot, in the process of mapping the cave and the outdoor environments. Consequently, the projected slice is selected to begin slightly lower.

All scenarios were executed, and computation times were measured on the same computer. The tests were run on a single thread of an AMD 5850U CPU using Linux kernel version 6.9.7. The parameters used by the method are listed in Table I.

A. Conversion of Static 3D Voxel Maps

The 2D maps for the autonomous aerial vehicle (AAV) are shown in Figs. 3(c) and 4(c), and for the autonomous ground vehicle (AGV) are shown in Figs. 3(a) and 4(a). Note that low walls and scattered objects are not relevant for aerial robots, as there is navigable space for aerial robots above the walls and scattered objects. For the ground robots, however, the walls and scattered objects are highly relevant and are retained by the proposed method, using the information contained in the slope map (the procedure is described in Section III-C) and using the slope threshold parameter R_{MS}. The slope maps are depicted in Figs. 3(f) and 4(f).

The slope map is generated from the height map $\mathcal{H}$. The height of the floor, $\check{\mathcal{H}}$, with respect to the origin of the voxel map, is depicted in Figs. 3(d) and 4(d).

The maps generated by the fixed projection method are shown in Figs. 3(b) and 4(b). In the cave scenario, the fixed projection method produces a 2D map without any issues (see 3(b)). However, in the outdoor scenario, the fixed projection method struggles to generate a reliable 2D map of the environment. This is because, the fixed portion that is projected is positioned too high to capture a large part of the outdoor map (which has an elevation range of up to 8 m). Obstacles are excluded from converted 2D map as they are located below the projected area. The computation time to perform the full map conversion can be seen in the Table II, under the title "full map time".

Table III shows the reduction in the raw size of one of the 2D maps with height data compared to the original voxel map. The voxel map is stored as an octree, while the 2D map and the height data are stored in a 2D matrix.

To validate the path conversion from 2D to 3D using the 2D map $\mathcal{M}$ and the height map $\mathcal{H}$, two 2D paths were generated per environment: one for the autonomous aerial vehicle (AAV), as shown in Figs. 3(c) and 4(c), and another for the autonomous ground vehicle (AGV), as seen in Figs. 3(a) and 4(a). The paths

TABLE III
COMPARISON BETWEEN THE RAW SIZES OF DIFFERENT MAPS, EXPRESSED IN MEGABYTES (MB) AND AS A PERCENTAGE COMPARED TO THE ORIGINAL UFOMAP USED TO GENERATE THE 2D MAP

Scenario	UFOMap (MB)	2D Map (MB)	2D Map + Height Data (Floor) (MB)	2D Map + Height data (Floor & Ceiling) (MB)
Cave	6.7 [100%]	0.3 [4.5%]	1.6 [23.9%]	2.9 [43.3%]
Outdoor	169.7 [100%]	6.3 [3.7%]	31.7 [18.7%]	57.1 [33.6%]

The UFOMap is stored as an octree, while the 2D map and height data are stored in a 2D matrix.

where generated using a path-planer developed for our earlier work on exploration [17] that is designed only for a 2D map. The 2D paths are converted into 3D paths for both the robots using the method in IV. The resulting 3D paths are shown in Figs. 3(e) and 4(e).

B. Map Conversion With Continuously Expanding Voxel Map

In each scenario, a dataset recorded during a robotic platform's exploration of the environment was used, to generate a continuously growing voxel map. In the cave scenario, the voxel map was updated at 2.37 Hz, and in the outdoor scenario, the voxel map was updated at 0.95 Hz. The goal of this validation step is to benchmark the performance of the methods in an online exploration scenario. The 2D map is updated alongside the Voxel map. Only the updated section of the 3D map was used for map conversion for both the proposed method and the fixed projection method. The computation times for both methods in the two scenarios are presented in the box plots in Fig. 2, and the average computation times are presented in Table II.

VII. DISCUSSION

As demonstrated in Section VI, the proposed method effectively performs voxel map to 2D map conversion in environments featuring non-uniform height variations and in the presence of overhangs or ceilings. The AGV map includes obstacles and walls impassable by the AGV, thus ensuring comprehensive environmental representation.

1) Comparison With Fixed Projection Method: The proposed method was compared to the state-of-the-art fixed projection method used by [15], [12] for voxel-based maps. As seen in Table II, the proposed method is computationally slower than the fixed projection method. This is expected, as the fixed projection method is significantly simpler, as it only needs to iterate through the map in the fixed region, projecting each voxel onto the 2D map. In contrast, the proposed method must check the occupancy of the bounding wall of the free space areas and perform a traversability estimation to detect obstacles in the map.

However, compared to fixed projection, the proposed method can handle situations where the height of the ground is not constant. A good example of this is the outdoor environment (Fig. 4), where the height difference from the starting position to the lowest ground point is more than 8 meters. The fixed projection method cannot generate a complete map of the environment, as can be seen in Fig. 4(b), whereas the proposed method generates a complete map with all obstacles included, as seen in Fig. 4(a). This is because, the height of the ground changes in outdoor scenarios, and thus, low obstacles that are below the height of the fixed slice being projected are lost in resulting 2D map. In the lower parts of the map, even walls of buildings and free space mapped by the voxel map are below the projected slice.

The height of the projected region could be adjusted for a better result, but for the outdoor scenario (Fig. 4(b)), there does not exist a height range for projection that results in a complete map of the area. Moreover, selecting the correct slice for projection is highly dependent on the robot and the environment, and it can require multiple experimental runs to figure out the best settings for the fixed projection. This is another strength of the proposed method, as there is no need for such tuning to achieve functional 2D maps. Another advantage of the proposed method is that it generates 2D maps for both AAVs and AGVs simultaneously, whereas fixed projections need to be run twice at different settings. Also, using a fixed height range for projection is not very practical for AAVs, as they need to operate at different heights in an environment.

The raw file sizes of the 2D maps generated by the proposed method and the original voxel map are presented in Table III. In cases where only AAV or AGV maps need to be wirelessly communicated, the required size is less than 5% of the original voxel map, which is significant in multi-agent scenarios where bandwidth is limited.

2) Online Implementation Over Expanding Voxel Maps: In the validation scenario with a continuously expanding voxel map (Section VI-B), in smaller enclosed environments such as subterranean areas (caves), the average computation time is only 0.10 s, enabling near real-time map updates, as shown in Fig. 2(a). In contrast, in more open environments like outdoor scenarios, the conversion time increases to an average of 0.83 s. Despite this increase, it remains faster than the average map update time, which is 0.42 s for the cave scenario and 1.05 s for the outdoor scenario. There are some outliers in the conversion time in the outdoor scenario, as shown in the boxplot in Fig. 2(b), where the conversion time exceeds that of the voxel map updates. In such cases, the missed voxel map updates will be included with the next 2D map update.

It is clear from the results that for smaller environments, the proposed method is fast enough for real-time re-planning. However, in larger environments, the map update might not be fast enough for reliable dynamic local re-planning. In such cases, the 2D map must instead be used only for global planning, and another solution must be employed for reactive navigation.

3) Limitations of the Proposed Method: An issue with the AGV map generated by the proposed method is the inaccurate representation at some frontiers, which are the boundaries between free and unknown space. This inaccuracy arises from the greedy method for floor detection presented in Section III-A, which inaccurately classifies the floor height around the frontiers, resulting in a steep slope in these areas.

Despite this limitation, the mapped area remains accurate for AGV navigation. In scenarios such as autonomous exploration, reliable frontier detection is an absolute requirement, necessitating reliance on the voxel map for frontier detection.

A limitation of using a 2D planner for AAV path planning instead of a full 3D planner is that, in scenarios with multiple 3D paths leading to a target point, the 2D planner may choose suboptimal paths, especially in environments with significant height differences. For example, it may select a path that flies over a tall obstacle instead of circumventing it. Another issue with using a 2D planner for AAVs and subsequent 2D-3D path conversion is that, a planner that generates a trajectory (comprising a path with a velocity target at each point) cannot be converted to 3D trajectories using the proposed method, as the velocities will not be accurate in environments with significant slopes. A potential solution is to integrate the height map generated by the proposed method into the 2D planner's calculations, to account for height when planning and optimizing the trajectory.

VIII. Conclusion

Motivated by the utility of converting 3D maps to 2D for global navigation, which enables the use of methods designed for 2D maps in 3D environments, and faster information sharing between robots, this work presented a novel method for converting 3D voxel maps to 2D occupancy maps for AAVs and AGVs, along with accompanying height and slope maps. A method was also proposed for converting the paths generated by the 2D planer back into 3D space using the height information extracted by the method. The method proposed in this paper is capable of generating 3D paths for both ground and aerial robots and was validated successfully in two different environments (a cave and an outdoor scenario).

References

[1] R. K. Megalingam, S. Tantravahi, H. S. S. K. Tammana, and H. S. R. Puram, "2D-3D hybrid mapping for path planning in autonomous robots," *Int. J. Intell. Robot. Appl.*, vol. 7, no. 2, pp. 291–303, Jun. 2023.

[2] O. Wulf, K. O. Arras, H. I. Christensen, and B. Wagner, "2D mapping of cluttered indoor environments by means of 3D perception," in *Proc. IEEE Int. Conf. Robot. Automat.*, Apr. 2004, pp. 4204–4209.

[3] S. Sandfuchs, M. P. Heimbach, J. Weber, and M. Schmidt, "Conversion of depth images into planar laserscans considering obstacle height for collision free 2D robot navigation," in *Proc. 2021 IEEE Eur. Conf. Mobile Robots*, Aug. 2021, pp. 1–6.

[4] A. Mora, R. Barber, and L. Moreno, "Leveraging 3D data for whole object shape and reflection aware 2D map building," *IEEE Sensors J.*, vol. 24, no. 14, pp. 21941–21948, Jul. 2024.

[5] T. H. Nam, J. H. Shim, and Y. I. Cho, "A 2.5D map-based mobile robot localization via cooperation of aerial and ground robots," *Sensors*, vol. 17, no. 12, Dec. 2017, Art. no. 2730.

[6] K. Kamarudin et al., "Method to convert Kinect's 3D depth data to a 2D map for indoor SLAM," in *Proc. IEEE 9th Int. Colloq. Signal Process. Appl.*, Mar. 2013, pp. 247–251.

[7] L. Garrote, J. Rosa, J. Paulo, C. Premebida, P. Peixoto, and U. J. Nunes, "3D point cloud downsampling for 2D indoor scene modelling in mobile robotics," in *Proc. 2017 IEEE Int. Conf. Auton. Robot Syst. Competitions*, Apr. 2017, pp. 228–233.

[8] G. Brahmanage and H. Leung, "Building 2D maps with integrated 3D and visual information using kinect sensor," in *Proc. 2019 IEEE Int. Conf. Ind. Cyber Phys. Syst.*, May 2019, pp. 218–223.

[9] A. Yusefi, A. Durdu, and C. Sungur, "ORB-SLAM-based 2D reconstruction of environment for indoor autonomous navigation of AAVs," *Avrupa Bilim ve Teknoloji Dergisi*, pp. 466–472, Oct. 2020 .

[10] R. Mur-Artal, J. M. M. Montiel, and J. D. Tardós, "ORB-SLAM: A versatile and accurate monocular SLAM system," *IEEE Trans. Robot.*, vol. 31, no. 5, pp. 1147–1163, Oct. 2015.

[11] Y. Li, D. Wang, Q. Li, G. Cheng, Z. Li, and P. Li, "Advanced 3D navigation system for AGV in complex smart factory environments," *Electronics*, vol. 13, no. 1, Jan. 2024, Art. no. 130.

[12] S. Yang, S. Yang, and X. Yi, "An efficient spatial representation for path planning of ground robots in 3D environments," *IEEE Access*, vol. 6, pp. 41539–41550, 2018.

[13] H. Gim, M. Jeong, and S. Han, "Autonomous navigation system with obstacle avoidance using 2.5D map generated by point cloud," in *Proc. IEEE 21st Int. Conf. Control, Automat. Syst.*, Oct. 2021, pp. 749–752.

[14] J. Liu, X. Chen, J. Xiao, S. Lin, Z. Zheng, and H. Lu, "Hybrid map-based path planning for robot navigation in unstructured environments," in *Proc. 2023 IEEE/RSJ Int. Conf. Intell. Robots Syst.*, Oct. 2023, pp. 2216–2223.

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

[16] D. Duberg and P. Jensfelt, "UFOMap: An efficient probabilistic 3D mapping framework that embraces the unknown," *IEEE Robot. Automat. Lett.*, vol. 5, no. 4, pp. 6411–6418, Oct. 2020.

[17] S. Fredriksson, A. Saradagi, and G. Nikolakopoulos, "Robotic exploration through semantic topometric mapping," in *Proc. 2024 IEEE Int. Conf. Robot. Automat.*, 2024, pp. 9404–9410.