Real-Time Metric-Semantic Mapping for Autonomous Navigation in Outdoor Environments

Jianhao Jiao, *Member, IEEE*, Ruoyu Geng, Yuanhang Li, Ren Xin, *Graduate Student Member, IEEE*, Bowen Yang, *Member, IEEE*, Jin Wu, *Member, IEEE*, Lujia Wang, Ming Liu, *Senior Member, IEEE*, Rui Fan, *Senior Member, IEEE*, and Dimitrios Kanoulas, *Member, IEEE*

***Abstract*— The creation of a metric-semantic map, which encodes human-prior knowledge, represents a high-level abstraction of environments. However, constructing such a map poses challenges related to the fusion of multi-modal sensor data, the attainment of real-time mapping performance, and the preservation of structural and semantic information consistency. In this paper, we introduce an online metric-semantic mapping system that utilizes LiDAR-Visual-Inertial sensing to generate a global metric-semantic mesh map of large-scale outdoor environments. Leveraging GPU acceleration, our mapping process achieves exceptional speed, with frame processing taking less than 7*ms*, regardless of scenario scale. Furthermore, we seamlessly integrate the resultant map into a real-world navigation system, enabling metric-semantic-based terrain assessment and autonomous point-to-point navigation within a campus environment. Through extensive experiments conducted on both publicly available and self-collected datasets comprising 24 sequences, we demonstrate the effectiveness of our mapping and navigation methodologies.**

***Note to Practitioners*—This paper tackles the challenge of autonomous navigation for mobile robots in complex, unstructured environments with rich semantic elements. Traditional navigation relies on geometric analysis and manual annotations, struggling to differentiate similar structures like roads and sidewalks. We propose an online mapping system that creates a global metric-semantic mesh map for large-scale outdoor environments, utilizing GPU acceleration for speed and overcoming the limitations of existing real-time semantic mapping methods, which are generally confined to indoor settings. Our map integrates into a real-world navigation system, proven effective in localization and terrain assessment through experiments with both public and proprietary datasets. Future work will focus on integrating kernel-based methods to improve the map's semantic accuracy.**

***Index Terms*— Autonomous driving, mapping, navigation.**

Manuscript received 4 February 2024; revised 9 May 2024; accepted 2 June 2024. Date of publication 1 August 2024; date of current version 7 March 2025. This article was recommended for publication by Associate Editor Z. Liu and Editor J. Yi upon evaluation of the reviewers' comments. This work was supported by the National Natural Science Foundation of China (Grants No. 62303388 and 62233013), UK Research and Innovation Future Leaders Fellowship (RoboHike, Grant No. MR/V025333/1), Shanghai Municipal Science and Technology Major Project (Grant No. 2021SHZDZX0100), and Xiaomi Young Talents Program. *(Corresponding author: Jianhao Jiao.)*

Jianhao Jiao and Dimitrios Kanoulas are with the Department of Computer Science, University College London, WC1E 6BT London, U.K. (e-mail: ucacjji@ucl.ac.uk; d.kanoulas@ucl.ac.uk).

Ruoyu Geng, Ren Xin, and Ming Liu are with the Robotics and Autonomous Systems, The Hong Kong University of Science and Technology (Guangzhou), Nansha, Guangzhou, Guangdong 511400, China.

Yuanhang Li, Bowen Yang, and Jin Wu are with the Department of Electronic and Computer Engineering, The Hong Kong University of Science and Technology, Hong Kong, SAR, China.

Lujia Wang is with The Hong Kong University of Science and Technology (Guangzhou), Nansha, Guangzhou, Guangdong 511400, China.

Rui Fan is with the College of Electronics and Information Engineering, Shanghai Research Institute for Intelligent Autonomous Systems, the State Key Laboratory of Intelligent Autonomous Systems, and the Frontiers Science Center for Intelligent Autonomous Systems, Tongji University, Shanghai 201804, China (e-mail: rui.fan@ieee.org).

This article has supplementary downloadable material available at https://doi.org/10.1109/TASE.2024.3429280, provided by the authors.

Digital Object Identifier 10.1109/TASE.2024.3429280

I. Introduction

A. Motivation

As THE basis of localization and navigation, mapping is of growing importance in robotics. Mapping is the process of establishing an internal representation of environments which can be operated by algorithms [1]. As the widely used representation, metric maps (also referred to as "geometric maps") store geometry of a scene and are usually defined by positions of landmarks, distance to obstacles, or binary values to indicate free and occupied space, which are critical for robots to optimize a smooth and collision-free trajectory. However, metric maps have difficulty in maintaining the long-term consistency since geometric features are sensitive to illumination and structural changes. Also, metric maps have limitation in encoding human-readable information. It is inconvenient for robots to execuate abstract human instructions (*e.g.,* "navigate to the building" and "follow driving rules").

In contrast, metric-semantic mapping [2] is the capability to group semantic concepts into metric maps. The inclusion of human-labeled information facilitates many tasks such as scene abstraction [2] and exploration [3]. In this paper, we focus on the autonomous navigation task of ground robots in complicated environments with abundant semantic elements. A typical scenario is shown in Fig. 1, where many different objects such as trees and buildings appear. It is also composed of the sidewalk that is specifically designed for pedestrains. By incorporating human-prior knowledge, the semantic map enables the vehicle to navigate along the road, finding a path that is free of collisions and avoids intersecting with sidewalks and grasslands. However, geometry-based traversability extraction methods often face challenges in distinguishing between roads, sidewalks, and grass due to their similar structures. Hence, this paper aims to investigate

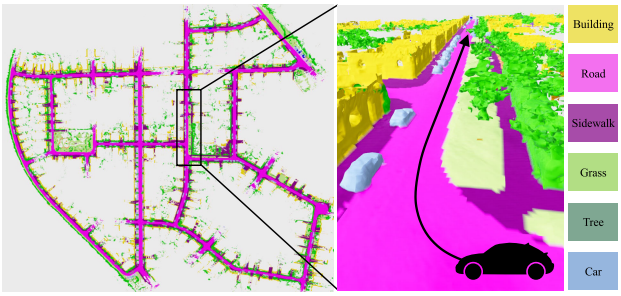


Fig. 1. To successfully navigate in the complicated environment or conduct high-level or interactive tasks for a robot (such as the vehicle shown in the figure), semantic information that categorizes surrounding objects at a human-readable format is required.

the online metric-semantic mapping method and its potential application in navigation systems.

B. Challenges

We consider that a desirable metric-semantic mapping approach should meet the following requirements:

1) **Accuracy:** The approach should aim to construct a map that closely represents real-world environments using onboard sensor data. However, factors such as measurement noise, different view angles, and limited observations affect the quality of the map construction.
2) **Efficiency:** Mapping is typically a time-consuming task, as numerous map elements need to be queried and updated based on new input. It is crucial to ensure real-time and consistent performance, especially for high-resolution or large-scale mapping applications.
3) **Versatility:** The resulting metric-semantic map should be capable of supporting a wide range of applications, including but not limited to localization, path planning, and environment understanding [2].

C. Contributions

As the **primary dcontribution**, we propose an online mapping system to address these challenges. This system leverages LiDAR-visual-inertial sensing to estimate the real-time state of the robot and construct a lighweight and global metric-semantic mesh map of the environment. To achieve this, we build upon the work of NvBlox [4] and thus utilize a signed distance field (SDF)-based representation. This representation offers the advantage of constructing surfaces with sub-voxel resolution, enhancing the accuracy of the map. While the focus of this paper is on mapping outdoor environments, the proposed solution is easily adaptable for various applications. The modular system consists of four primary components:

1) *State Estimator* (Section IV-A) is a LiDAR-visual-inertial odometry (LVIO) module implementing the Extended Kalman Filter (EKF) to estimate real-time sensors' poses with a local and sparse color point cloud.
2) *Semantic Segmentation* (Section IV-B) is a pre-trained convolutional neural network (CNN) that assigns a class label to every single pixel of each input image. A novel dataset that categorizes objects into diverse classes for the network training is also developed.
3) *Metric-Semantic Mapping* (Section IV-C) takes sensors' measurements and poses as input, and constructs a 3D global mesh of environments using the implicit SDF-based volumetric representation with semantic annotations from the 2D pixel-wise segmentation. The whole pipeline is implemented in parallel with the GPU and thus achieves the real-time performance. To approximate the surface geometry more accurate and complete (*e.g.,* less holes), the original distance calculation is improved.
4) *Traversability Analysis* (Section IV-D) identifies drivable areas by analyzing the geometric and semantic attributes of the resulting mesh map, thus narrowing the search space for subsequent motion planning.

The **second contribution** is an extensive experimental evaluation focusing on mapping. We evaluated the mapping system using both public datasets and our own collected data, including the *SemanticKITTI* [5], *SemanticUSL* [6], and *FusionPortable* dataset [7]. Additionally, we collected two test sequences on campus, covering outdoor scenes with buildings, roads, and grasslands. Our robot system utilizes maps constructed from these self-collected sequences, enabling the robot to complete point-goal navigation missions.

The **third contribution** encompasses real-world experiments on autonomous navigation employing the metric-semantic map created by our mapping method. This effort effectively bridges the previously unconnected realms of semantic mapping and navigation. The semantic data encoded in the map translates human instructions, thereby enabling robots to navigate safely within unstructured environments. We will publicly release the code of semantic mapping and self-collected datasets in the project website.[1]

II. Related Work

This section reviews the current literature on mapping and navigation techniques, focusing specifically on algorithms developed for mobile robots in unstructured environments.

A. Geometric Mapping

Existing map representations are categorized into explicit and implicit approaches. Explicit representations such as point clouds and surfels are widely studied in localization [19]. But points or surfels lack connectivity, where latent structural information is missing. Another type of explicit representation is the triangular mesh, where structural information through vertices and triangle facets are preserved. Meshes can depict manifold structures and topology of objects, which have been applied in scene reconstruction [20] and planning [21]. However, explicit representations have difficulty in maintaining the up-to-date map over a long period [22], where environments always are changing (*e.g.,* dynamic objects).

Implicit representations of environments are categorized into volumetric, elevation, and radiance field-based mappings. The 2.5D elevation map, efficient for legged robots' footstep planning, stores height as a Gaussian variable per grid

[1]https://gogojjh.github.io/projects/2024_semantic_mapping

TABLE I
DIFFERENCES IN EXISTING WORKS ON SEMANTIC MAPPING

Method	Metric Mapping	Processing Unit	Semantic Update	Map Representation	Scale	Application
SLAM++ [8]	KineticFusion [9]	GPU	Objects' pose optimizaiton	TSDF	Indoor	AR
SemanticFusion [10]	ElasticFusion [11]	GPU	Bayesian update	TSDF	Indoor	Not presented
Mask-Fusion [12]	ElasticFusion [11]	GPU	Geometry enhances segmentation	TSDF	Indoor	Grasping; AR
Voxbox++ [13]	VoxBlox [14]	CPU	Bayesian update	TSDF	Indoor	Not presented
BKISemMapping [15]	BGKOctoMap [16]	CPU	Bayesian Kernal Inference	3D occupancy grid	Outdoor	Not presented
Kimera [2]	VoxBlox [14]	CPU	Bayesian update	TSDF	Indoor	Scene Graph
Sni-SLAM [17]	NICE-SLAM [18]	GPU	Optimization	Radiance Field	Indoor	Not presented
Ours	NvBlox [4]	GPU	Bayesian update	TSDF	Outdoor	Navigation

but falls short in multi-layered scenarios and constraining 6-DoF motions [23]. Radiance field approaches [18] offering the ability to infer unseen areas but at the cost of high computational demands for large-scale mapping. Volumetric methods store 3D scene geometry using discretized volumes, facilitating parallel GPU implementation for real-time applications. Approaches include occupancy grid mapping, which assigns occupancy probabilities to voxels [24], and SDF-based mapping [14], capturing precise surface geometries with distance functions.

Our approach leverages the Truncated Signed Distance Function (TSDF) for environment representation, utilizing GPU parallelization to enhance mapping efficiency. We introduce a non-projective distance calculation to accurately estimate voxel distances, sidestepping the memory-heavy ESDF creation needed for Voxblox's collision detection. Instead, we utilize mesh-based traversability analysis and occupancy data for optimizing ground robot navigation.

B. Semantic Mapping

Semantic maps often build upon geometric representations by annotating map elements with labels. Semantic mapping is often coupled with segmentation algorithms, such as DeepLab [25], by classifying voxels into object categories. The pioneering works in real-time metric-semantic mapping is SLAM++ [8], where semantic objects are represented with CAD models and their poses are optimized independently. Recent studies such as SemanticFusion [10], Mask-Fusion [12], and Voxblox++ [13] have developed dense, voxel-based semantic maps, utilizing the map's geometry to enhance frontend segmentation. The Sni-SLAM [17] is proposed as the NeRF semantic mapping method. Table I summarizes some of them. As our closest work, Kimera [2] leverages a CPU-based framework (built upon VoxBlox) that uses RGB-D or stereo sensing to produce dense maps and employs visual-inertial odometry for motion estimation, mainly focusing on indoor environments. Conversely, our approach is specifically designed for the challenges of outdoor environments, introducing four major enhancements: *1)* LiDAR-visual-inertial sensing, which offers an extended measurement range, thereby substantially broadening the applicability of semantic mapping; *2)* enabling the construction of large-scale maps in real-time through the application of GPU parallelization techniques; *3)* introducing a comprehensive real-world and campus-scene semantic segmentation dataset; *4)* further leveraging the resulting metric-semantic map for localization and global planning purposes.

C. Terrain Traversability Recognition

The difficulty of navigation in unstructured environments mainly stems from variations of terrains. Several works [26] obtain traversability maps by extracting geometric attributes of the surface from LiDAR, including slope, height variation, and roughness, etc. But in many unstructured environments, path boundaries are commonly unclear and hardly inferred from geometry. Several following works [15], [24], [27] employ semantic segmentation to identify traversability by encoding prior human knowledge. The work most similar to ours is TNS [27], which generates a 2D traversability grid map by combining semantic and geometric information to develop an autonomous excavator application. In comparison, our work focuses on constructing a 3D global metric-semantic map, which offers a more comprehensive and versatile representation for environments. We use the map to benefit several tasks such as localization and motion planning. Real-world navigation experiment with a robot is demonstrated.

III. PRELIMINARIES

A. Sensor Configuration

This paper employs a LiDAR-Visual-IMU (LVI) configuration for data collection in mapping, leveraging the complementary strengths of each sensor in outdoor environments. The *IMU* offers high-rate linear acceleration and angular velocity measurements for accurate motion estimation. *LiDAR* provides 3D point clouds for direct measurement of environmental structures, unaffected by changes in illumination or viewpoint. *Cameras* capture dense RGB images, enabling fine-grained object classification, though they require external processing for 3D structure recovery. Despite the potential of RGB-D sensors for depth information, their performance is limited by distance and lighting variations.

Our mapping system leverages the LVI setup to simultaneously estimate the robot's real-time states and construct a global metric-semantic map. Utilizing the active sensing capabilities of LiDAR for consistent geometric measurements across frames, we develop a LiDAR-centric odometry for precise state estimation. Metric mapping employs LiDAR data to create SDF-based environmental representations, while semantic segmentation is achieved through camera data for pixel-wise object labeling, effectively differentiating between similar structures like roads and sidewalks. These labeled images feed into a semantic mapping module, linking map elements with 2D labels via projection. We precede the detailed mapping system exposition with an introduction to basic notations and the sensor calibration process.

B. Notions and Definitons

In this paper, we consider the minimal LVI setting shown in Fig. 4(a). Frames of the world, LiDAR, IMU, and camera are defined as $()^w$, $()^l$, $()^b$, $()^c$ respectively. The IMU frame is commonly treated as the base frame. We use $\mathbf{t} \in \mathbb{R}^3$ and $\mathbf{R} \in SO(3)$ to represent the 3-D translation and rotation. Especially, the rotation matrix is from the Lie group $SO(3)$ where $\mathbf{R}^\top\mathbf{R} = \mathbf{I}$, $\det\mathbf{R} = 1$. With these notions, we can reprense a sensor pose like the IMU in the world frame at time k as $(\mathbf{R}^w_{b_k}, \mathbf{t}^w_{b_k})$. The basic element in volumetric mapping is the voxel. Each voxel is represented by V_i, where i denotes the index. The size of each voxel is denoted by ν. The set of all defined semantic label is denoted by $\mathcal{L} = \{\text{road, sidewalk, vegetation, } \cdots\}$.

C. Synchronization and Calibration

We employ a Field Programmable Gate Array (FPGA) to synchronize sensor clocks via an external signal trigger, ensuring minimal latency in data collection across multiple sensors. The FPGA, receiving a pulse-per-second (PPS) signal from the GPS, adjusts the signal frequencies for each sensor. Spatial-temporal calibration (*i.e.,* intrinsics, extrinsics, and time offsets) is crucial for multi-sensor fusion. For spatial calibration, the Matlab calibration toolbox calibrates camera intrinsics, and we employ significant rotation and translation movements alongside Kalibr [28] to calibrate camera-IMU extrinsics using a checkerboard. The LiDAR-camera extrinsics are determined using a checkerboard-based method [29], optimizing extrinsics by minimizing point-to-plane and line-to-plane distances for precise LiDAR-camera data association. Given the challenge of unknown communication latency and processing time, we optimize the transformation between LiDAR and camera frames rather than estimating time offsets directly, as detailed in Section IV-A.

IV. Mapping

Fig. 2 shows the architecture of the proposed mapping system. The system starts with a **state estimator** (see Section IV-A), in which sensors' poses are estimated from sensor measurements. The **semantic segmenation** module (see Section IV-B) predicts pixel-wise labels for each image. It can be replaced by a point cloud-based segmentation network. The **metric-semantic mapping** module (see Section IV-C) utilizes multi-model data (e.g., point clouds, RGB images, labeled images) to constructs the TSDF-based representation of environments. The resulting mesh map consists of geometric and semantic information of environments. It is further analyzed by the **traversability extraction** module (see Section IV-D) to support navigation tasks, *i.e.,* localization and motion planning (see Section IV-E. All these modules use ROS's "Subscriber-Publisher" mechanism to transfer data.

A. State Estimator

The state estimator utilizes LVI odometry for real-time pose estimation. It is adapted from the R3LIVE system [30], which integrates LIO and VIO subsystems for sensor pose and local map estimation in a coarse-to-fine approach. The LIO subsystem uses IMU measurements for high-rate motion propagation and LiDAR scans to construct a 3D map, focusing on a local region to minimize memory usage. It employs an error-state iterated Kalman filter (ESIKF) to refine LiDAR state estimates by minimizing point-to-plane residuals. The residual is formulated as

$$\mathbf{0} = \mathbf{n}_j^{w\top}[\mathbf{R}^w_{l_k}\mathbf{p}^{l_k}_j + \mathbf{t}^w_{l_k} - \mathbf{q}^w_j], \tag{1}$$

where j is the index of a point in the LiDAR scan, $\mathbf{n}^w_j$ is the normal vector of the corresponding plane, and $\mathbf{q}^w_j$ is a point lying on the plane. The subsequent VIO subsystem renders a 3D map with RGB color with input images, *i.e.,* each map point is represented as $\{\mathbf{p}, \mathbf{c} = [R, G, B]^\top\}$. It computes camera's pose by minimize photometric errors between frame points and corresponding map points taking $\mathbf{R}^c_l\mathbf{R}^{l_k}_w$ as an initial guess. We do not directly setting $\mathbf{R}^c_l\mathbf{R}^{l_k}_w$ as the camera's pose due to the existing of non-zero time offset between the camera and LiDAR. The photometric error is defined as

$$\mathbf{0} = \mathbf{c}^{l_k}_j - \mathbf{I}_k[\kappa(\mathbf{R}^c_w\mathbf{p}^w + \mathbf{t}^c_w)], \tag{2}$$

where $\kappa(\cdot)$ projects a 3D point onto the image plane and $\mathbf{I}(u, v)$ returns the linearly interpolated RGB color at the pixel. Unlike the original R3LIVE approach, we exclude RGB points older than 3 seconds from the map for alignment, significantly reducing the memory footprint. After each frame, we relay updated sensor poses and undistorted point clouds to the following mapping modules.

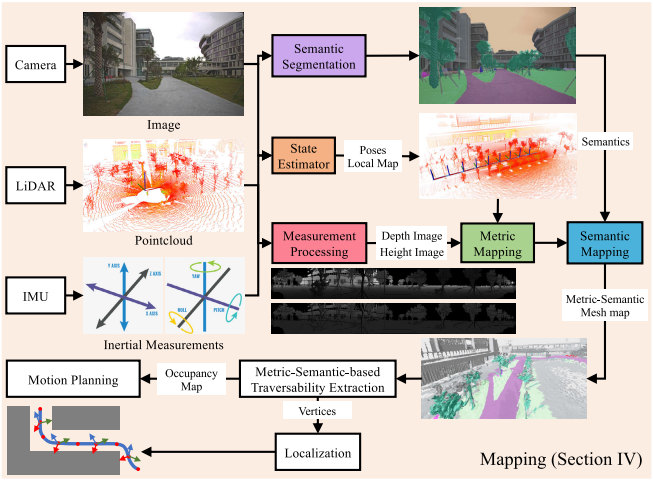


Fig. 2. Block diagram illustrating the full pipeline of the proposed mapping system. The system starts with the state estimation (see Section IV-A). The segmentation module (see Section IV-B) annotates each image pixel with a label. The measurement proecssing module converts point clouds into range and depth images. The mapping (see Section IV-C) constructs a global metric-semantic mesh map. The resulting map is extracted with traversable regions (see Section IV-D), and then used for localization and generating a collision-free path by a motion planning algorithm (see Section IV-E).

B. Semantic Segmentation

We design a network that is coupled with prototype learning for segmentation. It is composed of an off-the-shelf segmentation backbone [31], a customized segmentation head, and a confidence head. To guide the network to pay more attention to

the areas where the predictions are uncertain, the confidence head is used to predict pixel-wise aleatoric uncertainty [32] from images. The detail of the network is explained in [33].

C. TSDF-Based Volumetric Mapping

After obtaining the undistorted 3D scans (*e.g.,* LiDAR/RGB-D scans) and labeled images with associated poses, our metric-semantic mapping approach incrementally builds a dense 3D map. In traditional CPU-based serial pipeline, the time for updating voxels' values is linear to the number of data. This weakness limits the usage of sensors with large field of view (FOV) and dense measurements such as LiDARs. The state-of-the-art method (*i.e.,* Voxblox [14]) achieves the nearly real-time performance with point average. But this manner may unavoidably cause information loss. In contrast, the mapping pipeline including the retrieval and operation of all visible voxels is done in parallel within a GPU. Ths pipeline consists of three key modules: **measurement preprocessing** on point clouds, **metric mapping**, and **semantic mapping**. Here we introduce the details.

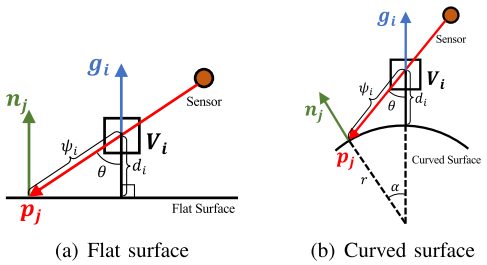
1) Measurement Preprocessing: As commonly done in learning-based approaches [34], point clouds are often converted into images and then processed in GPUs. With the known specifications of a LiDAR (*i.e.,* horizontal and vertical angular resolution $\Delta\phi$ and $\Delta\theta$ as well as the starting vertical angle θ_0), we project an undistorted point cloud onto a depth image D and height image H. Such images are very lightweight (100KB *v.s.* 10MB). Each pixel from D and H is calculated by the corresponding point as follows:

$$D(u,v) = Fr, \qquad H(u,v) = F(z+O),$$
$$u = \frac{\pi - \arctan(y,x)}{\Delta\phi}, \qquad v = \frac{\arccos(z/r) - \theta_0}{\Delta\theta}, \tag{3}$$

where (x, y, z) is the point's coordinate, z is its Euclidean distance to the origin, and F and O are two scalars.

2) Metric Mapping: The volumetric mapping divides the space into a set of voxels V_i. Each voxel has a unique global coordinate $\mathbf{v}_i \in \mathbb{Z}^3$, from which the raw coordinate of its center is $\mathbf{x}_i = v\mathbf{v}_i = [x, y, z]^\top$ (v is the voxel size). Voxels are stored using a two-level hierarchy approach [35]. The first level implements a hash table that maps 3D grid indices to *VoxelBlocks*. This hash table can be queried in GPU kernels using an interface based on stdgpu [36]. Each VoxelBlock contains a small group of densely allocated $8 \times 8 \times 8$ voxels which are stored contiguously in GPU memory. In the second layer, each voxel insided in the block can be accessed.

In TSDF-based mapping, each voxel stores a truncated signed distance D_i, a weight W_i to indicate the confidence, and a normalized gradient vector $\mathbf{g}_i \in \mathbb{R}^3$ of the signed distance. Both VoxBlox and NvBlox define D_i as the projective distance that is equal to the distance along the sensor ray to the measured surface of each voxel. $D_i < 0$ means that the voxel is behind the surface. Instead, we utilize non-projective distance, as described in [37], by leveraging normal and gradient vectors to characterize the local planarity of surfaces and provide an approximation of the true distance. Fig. 3 visualizes the non-projective distance d_i. For each incoming depth and height image, we first compute the normal image $\mathbf{N}$. The normal of each pixel is computed as $\mathbf{N}(u,v) = \frac{(\mathbf{p}_{j,1}-\mathbf{p}_j)\times(\mathbf{p}_{j,2}-\mathbf{p}_j)}{\|(\mathbf{p}_{j,1}-\mathbf{p}_j)\times(\mathbf{p}_{j,2}-\mathbf{p}_j)\|}$, Where $\mathbf{p}_{j,1}$ and $\mathbf{p}_{j,2}$ correspond to points back-projected from $D(u, v-1)$ and $D(u-1, v)$, respectively. We traverse and ray-cast each pixel to retrieve visible voxels. The non-projective signed distance is then defined as



(a) Flat surface (b) Curved surface

Fig. 3. The non-projective distance uses the local planarity of surfaces to approximate the true distance d_i. ψ_i is the projective distance of the voxel. The gradient vector is computed as the weighted average of normal vectors. d_i is calculated according to equ.(4). The radius of the curved surface (approximate to a circle) in (b) is marked as r.

$$d_i = \begin{cases} |\cos\theta|\psi_i, & \text{if } \alpha \approx 0 \\ |\dfrac{(\cos\alpha - 1)\sin\theta}{\sin\alpha}| + |\cos\theta|\psi_i & \text{otherwise,} \end{cases} \tag{4}$$

where θ is the angle between the ray and the gradient $\mathbf{g}_i$ and α is the angle between the gradient and the surface normal corresponding to $\mathbf{p}_j$. We define τ as the truncated distance and $W_i = \frac{\psi+\tau}{2\tau}$ as the linear weight. The distance $D_{i,k}$ and weight $W_{i,k}$ of V_i are updated at the k^{th} input data as follows

$$D_{i,k+1} = \frac{W_k D_k + W_i \Gamma(d_i,\tau)}{W_{i,k} + W_i}, \quad W_{i,k+1} = W_{i,k} + W_i,$$
$$\Gamma(d,\tau) = \begin{cases} \min(d,\tau) & \text{if } d \geq 0 \\ \max(d,-\tau) & \text{if } d < 0. \end{cases} \tag{5}$$

3) Semantic Mapping: Given the k^{th} label image $\mathbf{I}_k$ and the associate pose, our semantic mapping only retrieve and update visible and valid voxels ($W > 0$) within the camera frustum by raycasting. Similar to the metric mapping, each voxel is projected onto the image plane to obtain the corresponding semantic label. Different from the metric mapping, we propose that each voxel stores a discrete probability distribution, $P(l_s)$ over the set of class labels, $l_s \in \mathcal{L}$. Each new voxel is initialized with a uniform distribution over the semantic classes, as we begin with no a priori evidence as to its latent classification. Besides labels, the segmentation network in Section IV-B also outputs per-pixel probability image $\mathbf{O}_{(u,v)}$ over the class labels $P(\mathbf{O}_{(u,v)} = l_s|\mathbf{I}_k)$. We can update the probability distribution of the i^{th} voxel by means of a recursive Bayesian update [38]:

$$P(l_s|\mathbf{I}_{1,\ldots,k}) = \frac{1}{Z} P(l_s|\mathbf{I}_{1,\ldots,k-1}) P(\mathbf{O}_{(u,v)} = l_s|\mathbf{I}_k), \tag{6}$$

where Z is a constant. The segmentation network may produce incorrect labels, while (6) associates label hypotheses from multiple images and combine evidence iteratively. After that, the global metric-semantic mesh is extracted using the marching cubes algorithm [39], where the label of each vertex is extracted from the one with the highest probability.

D. Traversability Extraction

Traditional methods are either based on visual features [40] or geometric structures [41], having limitations in complex unstructured environments with many road variations. The detection of traversability should consider both robots' mobility and human instructions. Robots' mobility is typically formulated according to their kinomatic properties. Regarding the latter factor, the introduced semantic information that encodes human knowledge benefits two aspects: identifying untraversable terrains and guiding robots to follow basic driving rules. Therefore, this section proposes a traversability extraction method that jointly considers geometric and semantic information from the resulting map $\mathcal{M}^w$.

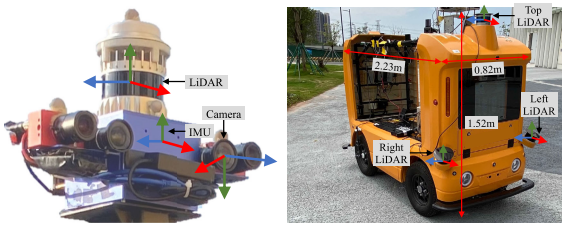
1) Analysis of Geometric Properties: The 3D mesh map, represented as the polygon mesh is a collection of vertices, edges, faces, and labels [20]. Each face provides normal information that is suitable for terrain assessment. We analyze the below geometric properties to determine whether the road is traversable or not from the geometric perspective: height difference v_{hd}, steepness v_s, and roughness v_r. Fig. 8 visualizes some examples. The "height difference" and "steepness" are used to indicate the risk of collision. And the "steepness" indicates the changing height of terrain. A vertex is selected if its v_{hd}, v_s, and v_r are all larger than thresholds t_{hd}, t_v, t_r. After that, we get the filtered mesh map $\mathcal{M}^{w'}$.

- **Height Difference** refers to the maximum difference in elevation between two points within a local region (*i.e.,* a ball $\mathcal{B}$ with radius r): $v_{hd} = \arg\max_{\boldsymbol{v}_i, \boldsymbol{v}_j \in \mathcal{B}} \|\boldsymbol{v}_i - \boldsymbol{v}_j\|$.
- **Steepness** refers to the degree of incline of a surface: $v_s = \arccos(\boldsymbol{n}_{\boldsymbol{v}_i})$.
- **Roughness** refers to the irregularities and unevenness of a ground: $v_r = \frac{1}{|\mathcal{B}|} \sum_{\boldsymbol{v} \in \mathcal{B}} \boldsymbol{n}_{\boldsymbol{v}}$.

2) Analysis of Semantic Properties: Due to the limited FoV of cameras, several vertices in the resulting map $\mathcal{M}^{w'}$ may not be labeled and are removed. The labeled vertices indicate the categories or classes of objects in the environment. In our approach, we can establish a strict definition of traversability based on prior knowledge and specific requirements for a particular robot. For example, we can classify "road" regions as drivable for vehicles (as shown in Fig. 4(b)), while "sidewalk" or "grass" regions are not.

E. Localization and Motion Planning

The resulting map plays a critical role in subsequent navigation tasks, serving as the global map for localization and planning. We extract vertices from $\mathcal{M}^w$ to form a global point cloud map. We use the prior map-based localization method [42] to obtain the real-time global pose of the vehicle by registering the map of the current scan. For motion planning, we project the vertices of the above traversable map onto a 2D occupancy grid map. Each grid cell is drivable if its associated "occupancy probability" is zero (see Fig. 8). To compute a collision-free and optimal global trajectory from an initial point to a specified goal, we utilize the search-based hybrid A* algorithm. This algorithm incorporates heuristics while accommodating the vehicle's nonholonomic constraints, enabling the generation of viable and smooth trajectories. A series of equidistant waypoints is discretized from the resultant path and then taken by the vehicle's speed controller.



(a) Mapping device (b) Autonomous vehicle

Fig. 4. (a) The mapping device that consists of a high-resolution LiDAR and camera is used to collect data for the environmental mapping. (b) The real-world vehicle provides a platform for testing the navigation system.

V. Experiment

We perform the mapping experiments on both public and self-collected datasets. First, for benchmarking, we perform experiments on three public datasets: the *SemanticKITTI*, the *SemanticUSL*, and the *FusionPortable* dataset. Second, we validate the proposed traversability extraction and motion planning method, with the demonstration of real-world navigation tests on an autonomous vehicle.

A. Implementation Details

The mapping system is mainly implemented with C++ with CUDA, while the semantic segmentation is implemented in Python with the Pytorch library. The mapping algorithms are tested on two computing platforms: a desktop PC equipped with an Intel i9-12900KF CPU, 64GB of RAM, and an Nvidia GeForce RTX 3080Ti GPU, as well as an embedded Nvidia Jetson ORIN 32GB computer. Besides public datasets, we also collect real-world data to test our mapping method. We use a handheld multi-sensor device (see Fig. 4(a)) to collect data. The device is installed a OS1 LiDAR (resolution: 128×1024), two FILR BFS-U3-31S4C global-shutter color cameras, and one STIM300IMU. During the data collection, the average movement speed is around $5m/s$. In real-world navigation experiments, we use the autonomous vehicle [43] (see Fig. 4(b)) for tests. The vehicle is mounted with four 16-beam LiDARs and one Livox mid-70 LiDAR. Table II shows parameters that are empirically set in experiments. The voxel size is different in *SemanticKITTI* 00, 02, 08 since the scope is very large and GPU memory is limited to store all voxels.

B. Dataset

This section presents the segmentation dataset that is collected at the campus and used to train our semantic segmentation network. We collect 15 data sequences covering most outdoor places of the campus and annotate 1092 images of size 2048×1536. We split 95% and 5% images into the train and validation set, respectively. Unlike existing datasets [44] that focus on urban areas, our dataset consists of many types of terrains (see Fig. 5) and anomaly objects, which is beneficial to

TABLE II
PARAMETERS IN EXPERIMENTS

Name	Mapping		Traversability Extraction	
	Voxel size ν	Truncated dis. τ	Radius r	Thresholds t_{hd}, t_v, t_r
Value	$0.25m$ or $0.3m$	5ν	$0.25m$	$0.6m,\ 20^\circ,\ 30^\circ$

SemanticKITTI $00, 02, 08$: $\nu = 0.3m$. Others: $\nu = 0.25m$.

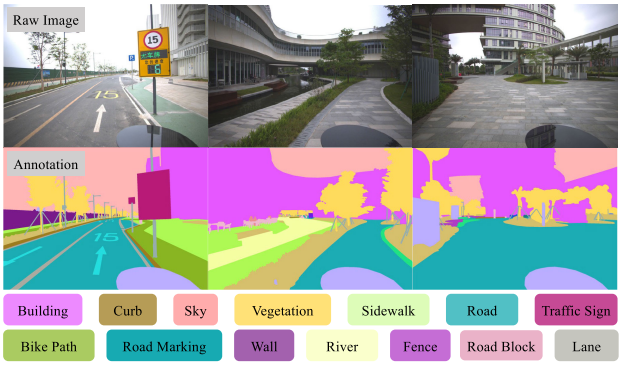


Fig. 5. We show a few samples from our dataset (top) and corresponding annotations (bottom). All images are collected at the campus.

downstream tasks including planning and navigation of ground robots. The network is pretrained on both the Cityscales [44] and our dataset, obtaining the 54.53% *Mean Intersection Over Union (mIoU)* on the validation set.

C. Metric-Semantic Mapping Experiments

1) Evaluation Metrics: We extract vertices from the reconstructed metric-semantic mesh map produced by our method for evaluation. The set of vertices form a point cloud $\mathcal{M}$ that is compared with respect to the ground-truth point cloud $\mathcal{G}$ based on five metrics: *Reconstruction Error (RE)* in terms of the RMSE, *Chamfer Distance (CD)*, *Reconstruction Coverage (RC)*, *mIoU*, and *Accuracy of Correctly Labeled Points (Acc)*. The latter two metrics evaluate the quality semantic segmentation of the resulting map [5].

- *Reconstruction Error* computes the average point-to-point distance between $\mathcal{M}$ and $\mathcal{G}$ [37]:

$$RE = \sqrt{\frac{1}{|\mathcal{M}|}\sum_{\boldsymbol{p}\in\mathcal{M}} \underbrace{\min(2\nu,\ \|\boldsymbol{p}-\boldsymbol{q}\|)^2}_{d(\boldsymbol{p},\mathcal{G})}}, \tag{7}$$

  where ν is the size of a voxel and $\boldsymbol{q} \in \mathcal{G}$ is the nearest point to $\boldsymbol{p}$.
- *Chamfer Distance* computes the Chamfer-L1 Distance [45] as:

$$CD = \frac{1}{2|\mathcal{M}|}\sum_{\boldsymbol{p}\in\mathcal{M}} d(\boldsymbol{p},\mathcal{G}) + \frac{1}{2|\mathcal{G}|}\sum_{\boldsymbol{q}\in\mathcal{G}} d(\boldsymbol{q},\mathcal{M}). \tag{8}$$

- *Reconstruction Coverage* is defined as the ratio between the number of GT points that do have a nearby point from $\mathcal{M}$ ($\leq 2\nu$) and the point number of $\mathcal{G}$ [37].
- *Semantic Mapping Score* is calculated in terms of the *Mean Intersection Over Union (mIoU) and Accuracy of Correctly Labeled Points (Acc)* [5].

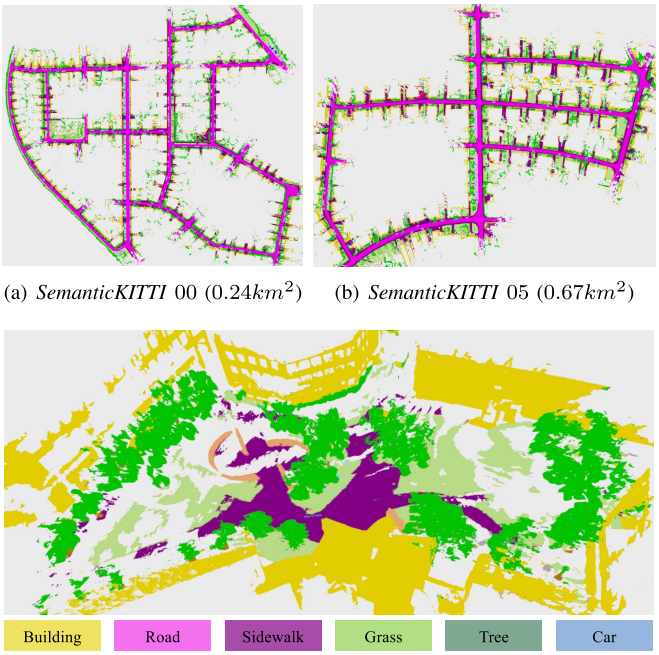
2) Baseline Methods: We compare our proposed mapping method with two state-of-the-art TSDF-based mapping methods: **VoxBlox** and **VoxField**, which are proposed in [14] and [37], respectively. Both of them are CPU-based mapping methods, but they do not support semantic mapping and traversability extraction. Our approach improves the non-projective distance calculation of VoxField by redesigning the weighting strategy. It also has much difference from VoxField in implementation, including measurement preprocessing, retrieval of visible voxels, and mesh generation using the marching cube algorithm. The other baselines should be variants of our method that use the original projective distance calculation (**Ours-Proj**) and does not use the recursive Bayesian Filter in semantic mapping (**Ours-wo-Bay**), respectively.

3) Results on Public Datasets: Both *SemanticKITTI* and *SemanticUSL* are two datasets that provide dense annotations for each LiDAR scan. Sequences 00–10 from *SemanticKITTI* and sequences 03, 12, 21, 32 from *SemanticUSL* are taken for evaluation since ground-truth labels and maps are provided. We utilize pretrained Cylinder3D [46] that is a state-of-the-art LiDAR-only semantic segmentation approach to generate semantic measurements in experiments. For experiments on *FusionPortable*, we only compute RE, CD, and RC socres since this dataset does not provide semantic annotations. Sequences *Garden_Night* (GN), *Canteen_Night* (CN), *Garden_Day* (GD), *Canteen_Day* (CD), *Escalator_Day* (ED), *Building_Day* (BD), and *Campus_Road_Day* (CRD) are taken in tests. Fig. 6 visualizes the resulting mesh map of several sequences. Since CRD does not provide the ground-truth map, only qualitative results are shown.

Quantitative results on all sequences are given in Table III. The average computation time reported in the table consists of processing time of these modules: normal image estimation, metric mapping, and semantic mapping. The non-projective distance calculation is validated to be effective since it improves the construction accuracy of VoxField and ours in terms of RE and CD, as compared with VoxBlox and Ours-Proj. Due to the advanced implementation of our methods, scores of RE, CD, and time are the highest for the most sequences. Ours has the lower coverage on *SemanticKITTI* and *SemanticUSL* datasets than VoxBlox does since our method removes unreliable LiDAR points that do not have normal or stay at a large incline angle (especially for ground points), making some voxels empty. These empty voxels do not have valid distance values, and thus cannot generate mesh. Data were collected in indoor buildings of the *FusionPortable* dataset. Most of LiDAR points are kept, and thus the RC of ours is large. Furthermore, both scores of mIoU and Acc of ours are larger than Ours-wo-Bay's, indicating the utility of recursive Bayesian update in maintaining the consistency of semantic information.

We conduct supplementary experiments to assess the influence of factors such as measurement noise, varying view angles, and limited observations on map construction. Due to the page limit, we show these results in the website.

4) Qualitative Results on Self-Collected Datasets: We use the mapping device to collect data in the campus. We collected two typical sequences that contain objects including roads,



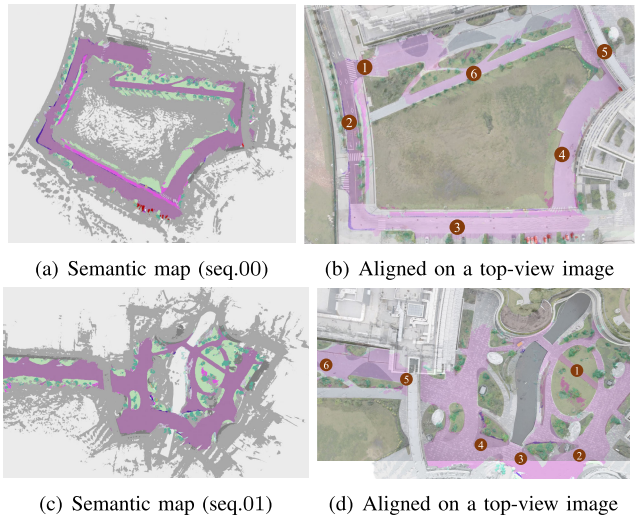
(a) *SemanticKITTI* 00 ($0.24km^2$) (b) *SemanticKITTI* 05 ($0.67km^2$)

(c) *SemanticUSL* 12 ($0.014km^2$)

(d) *FusionPortable Building_Day* ($0.019km^2$)

(e) *FusionPortable Campus_Road_Day* ($0.225km^2$)

Fig. 6. Results of the global map on four public sequences. Semantic labels are shown as colors on maps except for the *FusionPortable* dataset. Both (a), (b), and (c) use the same color scheme to (e) does.



(a) Semantic map (seq.00) (b) Aligned on a top-view image

(c) Semantic map (seq.01) (d) Aligned on a top-view image

Fig. 7. Semantic maps are created from the self-collected datasets: sequence 00 (top) and sequence 01 (bottom). Maps are manually aligned with images to show the specific meaning of labels. Traversable regions are then extracted from these maps. In navigation experiments, we command a vehicle to drive through goals that are marked in (b) and (d). Third-view pictures that show how the vehicle drives are presented in Fig. 9.

sidewalks, terrain, vegetations, vehicles, and buildings, which are appropriate to test our metric-semantic mapping method. Fig. 7 visualizes the resulting metric-semantic map that is aligned with the statllite image. Colors of each point of the map indicate the label. Due to the limited field of view of the camera, plenty of points are not annotated.

5) Timing: Table IV reports the detailed computation time regarding each step, with comparison of VoxBlox and VoxField. We take the typical sequence 00 of *SemanticKITTI* as an example which has over 0.24 km^2. Most of computations of mapping are done in GPUs and very fast, even on the Jetson ORIN. Computing normals on a range image requires around $0.2ms$. The metric mapping module that processes each new frame takes an average of $1.0ms$, including gathering visible voxels by ray tracing as well as updating their distance and weight. The semantic mapping module needs to find visible voxels and update their class probabilities via. the Bayesian filter, costing around $1.0ms$. Our method with the 3080Ti GPU only takes an average of $32.3ms$ to update the global metric-semantic mesh at a fixed frequency.

D. Point-Goal Navigation Experiments

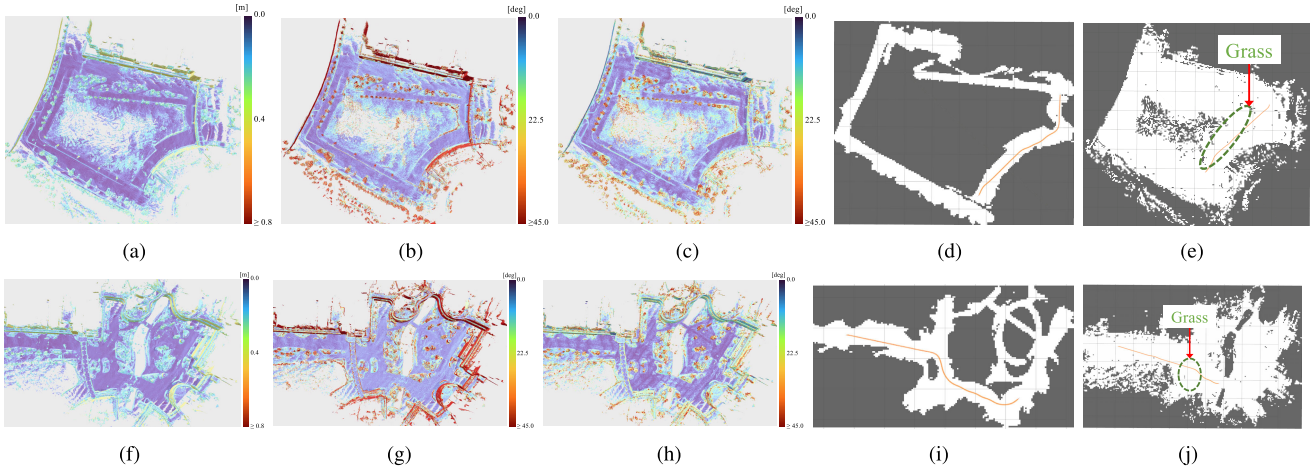
1) Results of Traversability Extraction: After computing geometric properties (*i.e.,* "height difference", "steepness", and "roughness") of the resulting metric-semantic mesh map for the two self-collected sequences, we visualize these values in Fig. 8. Considering the vehicle's mobility, objects that are not traversable such as cars, buildings, and trees are easily distinguished and filtered out by setting thresholds. But for the sidewalk (designed for pedestrians) and grassland which are not traversable for vehicles have to be distinguished by semantic information. By combining all geometric and semantic information, we obtain the 2D occupancy map, as shown in Fig. 8(d) and Fig. 8(i), respectively.

2) Results of Real-World Navigation: To demonstrate the practical application of our occupancy maps in motion planning and validate their effectiveness, we conducted a preliminary experiment. We designated start and end points on the map, with the resulting paths visualized in Fig. 8(d) and Fig. 8(j). This test reveals a critical insight: paths generated without integrating semantic information inadvertently cross through grassland areas. Such terrains, characterized by their uneven nature, pose significant navigational hazards, underscoring the vehicle's risk of becoming ensnared. This observation starkly highlights the necessity of semantic insights to distinguish between traversable and non-traversable regions, thereby ensuring the safety and reliability of the navigation paths chosen.

TABLE III

METRIC-SEMANTIC MAPPING RESULTS IN TERMS OF RECONSTRUCTION ERROR (↓), CHAMFER DISTANCE (↓), RECONSTRUCTION COVERAGE (↑), SEMANTIC MAPPING SCORE (↑), AND COMPUTATION TIME (↓)

Metrics	Methods	*SemanticKITTI*											*SemanticUSL*				*FusionPortable*					
		00	01	02	03	04	05	06	07	08	09	10	03	12	21	32	GN	CN	GD	CD	ED	BD
RE[cm]	VoxBlox	9.7	9.4	8.8	8.9	10.4	9.1	9.1	10.0	9.5	8.8	8.9	10.0	9.8	10.1	10.0	9.8	14.8	10.5	14.0	16.0	15.8
	VoxField	**7.2**	8.0	**6.1**	6.9	7.7	7.6	7.6	7.8	**7.5**	6.8	7.0	8.4	8.1	8.7	8.5	9.9	14.1	10.3	13.5	**15.1**	14.8
	Ours-Proj	7.8	**7.2**	6.8	6.3	7.0	6.9	7.0	6.8	8.4	6.2	6.1	7.4	7.6	7.3	7.3	9.0	15.1	9.5	13.4	15.5	14.3
	Ours	7.5	**7.2**	6.4	**5.9**	**6.8**	**6.6**	**6.7**	**6.5**	8.1	**5.8**	**5.8**	**7.1**	**7.1**	**6.7**	**6.9**	**8.8**	**13.5**	**8.9**	**12.6**	**15.1**	**13.8**
CD[cm]	VoxBlox	6.1	7.9	6.9	5.4	7.2	6.5	5.5	5.8	8.8	5.9	5.2	6.1	8.3	9.8	**10.2**	11.2	15.7	10.5	13.9	12.5	12.1
	VoxField	6.4	**7.7**	7.3	5.3	**6.9**	6.2	4.9	5.4	7.7	5.2	4.8	**5.8**	7.8	9.4	**10.2**	11.5	15.0	12.1	13.8	11.8	12.0
	Ours-Proj	6.3	8.2	7.6	5.1	7.5	6.0	4.9	5.4	8.0	5.1	4.8	6.0	7.9	9.6	10.5	9.1	16.0	9.9	13.2	12.2	11.2
	Ours	**5.9**	8.0	**6.5**	**5.0**	7.3	**5.7**	**4.5**	**5.1**	**7.5**	**4.7**	**4.5**	5.9	**7.7**	**9.3**	10.4	**8.2**	**14.0**	**9.2**	**12.7**	**11.1**	**10.6**
RC[%]	VoxBlox	**95.3**	**88.7**	**94.2**	**96.6**	**92.9**	**91.6**	95.2	**96.6**	86.1	95.3	**97.1**	**94.9**	**87.1**	82.9	**82.1**	70.8	58.2	68.0	63.6	75.2	81.9
	VoxField	91.4	87.3	89.6	95.3	90.2	91.2	95.6	95.5	87.9	**95.7**	96.5	94.4	87.0	**83.1**	80.5	69.6	59.0	67.7	63.3	75.4	80.0
	Ours-Proj	93.2	83.5	90.8	94.8	86.7	91.0	95.0	94.1	88.6	94.8	95.1	92.0	85.8	80.1	77.5	78.3	59.4	75.6	64.7	75.4	83.2
	Ours	94.0	83.7	93.3	94.7	87.0	91.3	**95.9**	94.3	**89.6**	95.4	95.4	92.1	85.9	80.9	77.3	**80.7**	**61.4**	**77.1**	**65.2**	**76.7**	**83.8**
mIoU[%]	Ours-wo-Bay	65.7	39.0	61.9	63.0	61.2	66.3	62.9	68.3	53.4	64.4	63.3	61.7	43.2	55.9	26.3	–	–	–	–	–	–
	Ours	**76.0**	**41.0**	**75.4**	**71.9**	**68.8**	**74.9**	**72.7**	**74.6**	**62.8**	**76.2**	**76.8**	**66.2**	**45.6**	**62.3**	**30.8**	–	–	–	–	–	–
Acc[%]	Ours-wo-Bay	88.7	87.5	86.7	88.5	89.4	85.8	85.8	89.0	83.1	86.8	84.2	94.6	93.5	90.2	68.0	–	–	–	–	–	–
	Ours	**92.9**	**90.5**	**92.1**	**93.0**	**92.3**	**90.0**	**90.7**	**92.2**	**87.4**	**91.2**	**89.4**	**96.2**	**95.6**	**94.0**	**74.2**	–	–	–	–	–	–
Time[ms]	VoxBlox	200.4	288.0	218.5	242.9	270.6	208.8	286.9	181.9	231.8	235.5	185.5	324.2	368.8	293.6	267.6	327.7	282.3	331.3	288.2	158.4	271.4
	VoxField	130.5	157.0	141.2	188.0	154.2	131.9	157.2	115.9	144.4	140.8	115.1	188.0	214.6	164.7	157.7	329.6	281.4	318.0	286.5	162.3	281.9
	Ours (3080Ti)	**2.4**	**6.5**	**2.5**	**3.7**	**4.3**	**3.1**	**4.1**	**2.9**	**2.8**	**3.6**	**2.8**	**5.4**	**6.8**	**6.3**	**6.0**	**1.6**	**1.6**	**1.6**	**1.5**	**1.4**	**1.7**
	Ours (ORIN)	17.2	31.9	17.9	24.2	26.0	23.0	25.6	21.0	20.3	24.7	19.9	29.0	34.4	24.9	22.6	16.7	16.2	16.2	16.4	11.5	16.8



(a) (b) (c) (d) (e)

(f) (g) (h) (i) (j)

Fig. 8. Visualization of geometric properties of the resulting metric-semantic mesh map and projected 2D occupancy maps for navigation on sequence00 (top) and sequence 01 (bottom): (a)(f) height difference, (b)(g) steepness, (c)(h) roughness, (d)(i) occupancy map using semantic information, and (e)(j) occupancy map without using semantic information. The yellow lines in (d)(i) and (e)(j) indicate the found navigation paths, where the path in (d)(i) does not intersect with untraversable regions.

TABLE IV

COMPUTATION TIME [ms] ON THE *SemanticKITTI* 00 AND ACCELERATION RATIO COMPARED WITH VOXFIELD

Methods	Normal Image	Metric Map.	Semantic Map.	Mesh Generation
VoxBlox	–	200.4 ± 30.6	–	111.2 ± 41.1
VoxField	6.3 ± 2.0	124.2 ± 17.6	–	99.7 ± 25.0
Ours (3080Ti)	$\mathbf{0.2} \pm 0.1$ ($\times 31.4$)	$\mathbf{1.0} \pm 0.2$ ($\times 124.2$)	$\mathbf{1.0} \pm 0.2$	$\mathbf{32.3} \pm 7.7$ ($\times 3.1$)
Ours (ORIN)	0.7 ± 0.3 ($\times 9.0$)	9.3 ± 0.7 ($\times 13.4$)	8.0 ± 0.9	232.8 ± 83.0 ($\times 0.4$)

We extended our research to include practical applications by deploying the map on a real-world vehicle. Demonstrated in Fig. 7(b) and Fig. 7(d), the vehicle was tasked with completing two navigation tests based on a series of predefined goal points. The motion planner successfully identified collision-free trajectories, enabling the vehicle to navigate the prescribed paths effectively. Visual evidence of these navigation tests is captured in third-person photographs, as showcased in Fig. 9, with the vehicle achieving an average speed of approximately $3m/s$. For a comprehensive view of these tests, we invite readers to view the accompanying demonstration video.

3) Discussion: Our proposed system represents a robust and efficient mapping solution, characterized by its high computational performance and versatile design. Crafted with modularity at its core and seamlessly integrated with the Robot Operating System (ROS), it offers unparalleled flexibility for customization to meet diverse application needs. Empirical evidence from our experiments underscores the metric-semantic map's superiority in facilitating enhanced visualization, precise localization, and effective traversability analysis for navigation. By embedding semantic information that reflects human knowledge, the system adeptly distinguishes between drivable and non-drivable areas, such as sidewalks and grasslands. This feature not only elevates

(a) Navigation in the region that is covered by the sequence 00.

(b) Navigation in the region that is covered by the sequence 01.

Fig. 9. Without driving into grassland and sidewalks, the vehicle successfully navigate via. regions that are covered by the self-collected dataset after being given a set of goal points. These pictures are capture in places which are indicated on maps shown in Fig. 7.

the safety of autonomous navigation but also significantly diminishes the human effort required for robotic system implementation, marking a notable advancement over previous efforts [43].

VI. Conclusion

In this paper, we introduce an online metric-semantic mapping system tailored for autonomous navigation, featuring LiDAR-IMU-Visual odometry, image-based semantic segmentation, TSDF-based mapping, and extraction of traversable areas. We further integrate this mapping with a navigation system, enhancing map-based localization and motion planning. Our evaluation includes extensive mapping and point-to-point navigation tests across 24 sequences from both public and proprietary datasets in a campus setting.

Despite its strengths, our system faces limitations, notably in GPU memory reliance, which challenges city-scale mapping scalability (*e.g.*, AutoMerge [47]). A potential remedy is a submap approach, balancing voxel storage between GPU for immediate access and CPU for less active data. Additionally, the absence of loop correction introduces drift over time, an issue that could be alleviated by integrating submap techniques and mesh deformation optimizations for map correction (*e.g.*, Kimera [48]). Lastly, maintaining semantic features' spatio-temporal consistency poses difficulties, with potential solutions hinted at in kernel-based methods (*e.g.*, [24]).

Acknowledgment

The authors thank Qingwen Zhang, Yingbing Chen, Mingkai Tang, Xiangcheng Hu, Hexiang Wei, and Tianshuai Hu for their suggestions on the mapping system and real-world navigation experiments. They also thank the Robotics and Autonomous System (ROAS) at The Hong Kong University of Science and Technology (GZ) for providing the experimental site, and gratefully acknowledge ChatGPT for polishing the manuscript.

References

[1] S. Thrun, "Robotic mapping: A survey," in *Exploring Artificial Intelligence in the New Millennium*, G. Lakemeyer and B. Nebel, Eds., San Mateo, CA, USA: Morgan Kaufmann, 2002.

[2] A. Rosinol et al., "Kimera: From SLAM to spatial perception with 3D dynamic scene graphs," *Int. J. Robot. Res.*, vol. 40, nos. 12–14, pp. 1510–1546, Dec. 2021.

[3] J. Wang, W. Chi, C. Li, C. Wang, and M. Q.-H. Meng, "Neural RRT*: Learning-based optimal path planning," *IEEE Trans. Autom. Sci. Eng.*, vol. 17, no. 4, pp. 1748–1758, Oct. 2020, doi: 10.1109/TASE.2020.2976560.

[4] A. Millane et al., "Nvblox: GPU-accelerated incremental signed distance field mapping," 2023, *arXiv:2311.00626*.

[5] J. Behley, A. Milioto, and C. Stachniss, "A benchmark for LiDAR-based panoptic segmentation based on KITTI," in *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, May 2021, pp. 13596–13603.

[6] P. Jiang and S. Saripalli, "LiDARNet: A boundary-aware domain adaptation model for point cloud semantic segmentation," in *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, May 2021, pp. 2457–2464.

[7] J. Jiao et al., "FusionPortable: A multi-sensor campus-scene dataset for evaluation of localization and mapping accuracy on diverse platforms," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS)*, Oct. 2022, pp. 3851–3856.

[8] R. F. Salas-Moreno, R. A. Newcombe, H. Strasdat, P. H. J. Kelly, and A. J. Davison, "SLAM++: Simultaneous localisation and mapping at the level of objects," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2013, pp. 1352–1359.

[9] R. A. Newcombe et al., "KinectFusion: Real-time dense surface mapping and tracking," in *Proc. 10th IEEE Int. Symp. Mixed Augmented Reality*, Oct. 2011, pp. 127–136.

[10] J. McCormac, A. Handa, A. Davison, and S. Leutenegger, "Semanticfusion: Dense 3D semantic mapping with convolutional neural networks," in *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, May/Jun. 2017, pp. 4628–4635.

[11] T. Whelan, S. Leutenegger, R. Salas Moreno, B. Glocker, and A. Davison, "ElasticFusion: Dense SLAM without a pose graph," in *Proc. Robot., Sci. Syst.*, Jul. 2015, pp. 1–9.

[12] M. Runz, M. Buffier, and L. Agapit, "MaskFusion: Real-time recognition, tracking and reconstruction of multiple moving objects," in *Proc. IEEE Int. Symp. Mix. Augment. Real.*, Oct. 2018, pp. 10–20.

[13] M. Grinvald et al., "Volumetric instance-aware semantic mapping and 3D object discovery," *IEEE Robot. Autom. Lett.*, vol. 4, no. 3, pp. 3037–3044, Jul. 2019.

[14] H. Oleynikova, Z. Taylor, M. Fehr, R. Siegwart, and J. Nieto, "Voxblox: Incremental 3D Euclidean signed distance fields for on-board MAV planning," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS)*, Sep. 2017, pp. 1366–1373.

[15] L. Gan, R. Zhang, J. W. Grizzle, R. M. Eustice, and M. Ghaffari, "Bayesian spatial kernel smoothing for scalable dense semantic mapping," *IEEE Robot. Autom. Lett.*, vol. 5, no. 2, pp. 790–797, Apr. 2020.

[16] K. Doherty, T. Shan, J. Wang, and B. Englot, "Learning-aided 3-D occupancy mapping with Bayesian generalized kernel inference," *IEEE Trans. Robot.*, vol. 35, no. 4, pp. 953–966, Aug. 2019.

[17] S. Zhu et al., "SNI-SLAM: Semantic neural implicit SLAM," 2023, *arXiv:2311.11016*.

[18] Z. Zhu et al., "NICE-SLAM: Neural implicit scalable encoding for SLAM," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2022, pp. 12786–12796.

[19] J. Jiao, H. Ye, Y. Zhu, and M. Liu, "Robust odometry and mapping for multi-LiDAR systems with online extrinsic calibration," *IEEE Trans. Robot.*, vol. 38, no. 1, pp. 351–371, Feb. 2022.

[20] J. Lin et al., "ImMesh: An immediate LiDAR localization and meshing framework," *IEEE Trans. Robot.*, vol. 39, no. 6, pp. 4312–4331, Dec. 2023.
[21] S. Pütz, T. Wiemann, M. K. Piening, and J. Hertzberg, "Continuous shortest path vector field navigation on 3D triangular meshes for mobile robots," in *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, May 2021, pp. 2256–2263.
[22] L. Schmid et al., "Panoptic multi-TSDFs: A flexible representation for online multi-resolution volumetric mapping and long-term dynamic scene consistency," in *Proc. Int. Conf. Robot. Autom. (ICRA)*, May 2022, pp. 8018–8024.
[23] B. Yang, Q. Zhang, R. Geng, L. Wang, and M. Liu, "Real-time neural dense elevation mapping for urban terrain with uncertainty estimations," *IEEE Robot. Autom. Lett.*, vol. 8, no. 2, pp. 696–703, Feb. 2023.
[24] L. Gan et al., "Multitask learning for scalable and dense multilayer Bayesian map inference," *IEEE Trans. Robot.*, vol. 39, no. 1, pp. 699–717, Feb. 2023.
[25] L. C. Chen, G. Papandreou, and I. Kokkinos, "DeepLab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected CRFs," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 40, no. 4, pp. 834–848, Jun. 2017.
[26] T. Miki, L. Wellhausen, R. Grandia, F. Jenelten, T. Homberger, and M. Hutter, "Elevation mapping for locomotion and navigation using GPU," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS)*, Oct. 2022, pp. 2273–2280.
[27] T. Guan, Z. He, R. Song, D. Manocha, and L. Zhang, "TNS: Terrain traversability mapping and navigation system for autonomous excavators," 2021, *arXiv:2109.06250*.
[28] P. Furgale, J. Rehder, and R. Siegwart, "Unified temporal and spatial calibration for multi-sensor systems," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, Nov. 2013, pp. 1280–1286.
[29] J. Jiao, F. Chen, H. Wei, J. Wu, and M. Liu, "LCE-Calib: Automatic LiDAR-frame/event camera extrinsic calibration with a globally optimal solution," *IEEE/ASME Trans. Mechatronics*, vol. 28, no. 5, pp. 2988–2999, Oct. 2023, doi: 10.1109/TMECH.2023.3259444.
[30] J. Lin and F. Zhang, "R3LIVE: A robust, real-time, RGB-colored, LiDAR-inertial-visual tightly-coupled state estimation and mapping package," in *Proc. Int. Conf. Robot. Autom. (ICRA)*, May 2022, pp. 10672–10678.
[31] J. Wang et al., "Deep high-resolution representation learning for visual recognition," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 43, no. 10, pp. 3349–3364, Oct. 2020.
[32] A. Kendall and Y. Gal, "What uncertainties do we need in Bayesian deep learning for computer vision?" in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 30, 2017, pp. 1–11.
[33] J. Liu, J. Zhang, and N. Barnes, "Modeling aleatoric uncertainty for camouflaged object detection," in *Proc. IEEE/CVF Winter Conf. Appl. Comput. Vis. (WACV)*, Jan. 2022, pp. 1445–1454.
[34] X. Chen et al., "Moving object segmentation in 3D LiDAR data: A learning-based approach exploiting sequential data," *IEEE Robot. Autom. Lett.*, vol. 6, no. 4, pp. 6529–6536, Oct. 2021.
[35] M. Nießner, M. Zollhöfer, S. Izadi, and M. Stamminger, "Real-time 3D reconstruction at scale using voxel hashing," *ACM Trans. Graph.*, vol. 32, no. 6, pp. 1–11, Nov. 2013.
[36] P. Stotko, "stdgpu: Efficient STL-like data structures on the GPU," 2019, *arXiv:1908.05936*.
[37] Y. Pan, Y. Kompis, L. Bartolomei, R. Mascaro, C. Stachniss, and M. Chli, "Voxfield: Non-projective signed distance fields for online planning and 3D reconstruction," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS)*, Oct. 2022, pp. 5331–5338.
[38] A. Rosinol, M. Abate, Y. Chang, and L. Carlone, "Kimera: An open-source library for real-time metric-semantic localization and mapping," in *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, May 2020, pp. 1689–1696.
[39] T. S. Newman and H. Yi, "A survey of the marching cubes algorithm," *Comput. Graph.*, vol. 30, no. 5, pp. 854–879, Oct. 2006.
[40] P. Furgale and T. D. Barfoot, "Visual teach and repeat for long-range rover autonomy," *J. Field Robot.*, vol. 27, no. 5, pp. 534–560, Sep. 2010.
[41] F. Yang, C. Cao, H. Zhu, J. Oh, and J. Zhang, "FAR planner: Fast, attemptable route planner using dynamic visibility update," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS)*, Oct. 2022, pp. 9–16.
[42] X. Hu et al., "PALoc: Advancing SLAM benchmarking with prior-assisted 6-DoF trajectory generation and uncertainty estimation," *IEEE/ASME Trans. Mechatronics*, early access, Mar. 26, 2024, doi: 10.1109/TMECH.2024.3362902.
[43] T. Liu et al., "The role of the hercules autonomous vehicle during the COVID-19 pandemic: An autonomous logistic vehicle for contactless goods transportation," *IEEE Robot. Autom. Mag.*, vol. 28, no. 1, pp. 48–58, Mar. 2021.
[44] M. Cordts et al., "The cityscapes dataset for semantic urban scene understanding," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2016, pp. 3213–3223.
[45] L. Mescheder, M. Oechsle, M. Niemeyer, S. Nowozin, and A. Geiger, "Occupancy networks: Learning 3D reconstruction in function space," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, Jun. 2019, pp. 4460–4470.
[46] H. Zhou et al., "Cylinder3D: An effective 3D framework for driving-scene LiDAR semantic segmentation," 2020, *arXiv:2008.01550*.
[47] P. Yin et al., "AutoMerge: A framework for map assembling and smoothing in city-scale environments," *IEEE Trans. Robot.*, vol. 39, no. 5, pp. 3686–3704, Oct. 2023.
[48] Y. Tian, Y. Chang, F. H. Arias, C. Nieto-Granda, J. P. How, and L. Carlone, "Kimera-multi: Robust, distributed, dense metric-semantic SLAM for multi-robot systems," *IEEE Trans. Robot.*, vol. 38, no. 4, pp. 2022–2038, Aug. 2022.

Jianhao Jiao (Member, IEEE) received the B.Eng. degree in instrument science from Zhejiang University, Hangzhou, China, in 2017, and the Ph.D. degree from the Department of Electronic and Computer Engineering, The Hong Kong University of Science and Technology, Hong Kong, China. He is currently a Senior Research Fellow with the Department of Computer Science, University College London. His research interests include navigation and embodied intelligence.

Ruoyu Geng received the B.S. degree from Jilin University, Changchun, China. She is currently pursuing the Ph.D. degree with the Department of Robotics and Autonomous Systems, The Hong Kong University of Science and Technology (Guangzhou). Her research interest mainly focuses on learning-aided SLAM and semantic segmentation.

Yuanhang Li received the B.S. degree from Xiamen University, Xiamen, China, in 2021. He is currently pursuing the Ph.D. degree with the Department of Electronic and Computer Engineering, The Hong Kong University of Science and Technology. His areas of research include multi-object tracking and trajectory planning algorithms.

Ren Xin (Graduate Student Member, IEEE) received the B.Eng. degree in mechanical design and its automation from the Ocean University of China, Qingdao, China. He is currently pursuing the Ph.D. degree with the Department of Robotics and Autonomous Systems, The Hong Kong University of Science and Technology (Guangzhou). His research interest mainly focuses on learning based methods of path planning, motion prediction, and map representations.

Bowen Yang (Member, IEEE) received the B.Eng. degree from The Hong Kong Polytechnic University, China, in 2019. He is currently pursuing the Ph.D. degree with the Department of Electronic and Computer Engineering, The Hong Kong University of Science and Technology, China. His research mainly focuses on scene understanding and path planning for autonomous robots.

Jin Wu (Member, IEEE) received the B.S. degree from the University of Electronic Science and Technology of China, Chengdu, China. He is currently pursuing the Ph.D. degree with the Department of Electronic and Computer Engineering, The Hong Kong University of Science and Technology. He has been a Research Assistant with the Department of Electronic and Computer Engineering, The Hong Kong University of Science and Technology, since 2018. He was with Tencent Robotics X, Shenzhen, China, from 2019 to 2020. From 2012 to 2018, he was with UAV industry and has brought up two companies.

Lujia Wang received the Ph.D. degree from the Department of Electronic Engineering, The Chinese University of Hong Kong, Sha Tin, Hong Kong, in 2015. From 2016 to 2021, she was an Associate Professor with Shenzhen Institutes of Advanced Technology, Chinese Academy of Sciences, Shenzhen, Guangdong. She is currently a Senior Lecturer of the Red Bird M.Phil. Program, The Hong Kong University of Science and Technology (Guangzhou).

Ming Liu (Senior Member, IEEE) received the Ph.D. degree from the Department of Mechanical and Process Engineering, ETH Zürich, Zürich, Switzerland, in 2013, supervised by Prof. Roland Siegwart. He is currently an Associate Professor of Robotics and Autonomous Systems with The Hong Kong University of Science and Technology (Guangzhou).

Rui Fan (Senior Member, IEEE) received the B.Eng. degree in automation from Harbin Institute of Technology in 2015, and the Ph.D. degree in electrical and electronic engineering from the University of Bristol in 2018. He began his faculty career as a Full Research Professor with the College of Electronics and Information Engineering, Tongji University, in 2021, and then promoted to a Full Professor; and Shanghai Research Institute for Intelligent Autonomous Systems in 2022. His research interests include computer vision, deep learning, and robotics, with a specific focus on humanoid visual perception under the two-streams hypothesis.

Dimitrios Kanoulas (Member, IEEE) received the Ph.D. degree from Northeastern University, Boston. He was a Post-Doctoral Researcher with the Italian Institute of Technology, for five years. He is currently a Professor of Robotics and AI with the Department of Computer Science, University College London (UCL); and a UKRI Future Leaders Fellow (FLF). He has published more than 50 research papers in high-impact robotic journals and conferences.