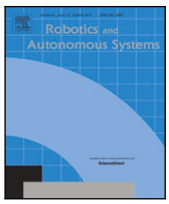
Robotics and Autonomous Systems 172 (2024) 104513

Contents lists available at ScienceDirect

Robotics and Autonomous Systems

journal homepage: www.elsevier.com/locate/robot



Semantic interpretation of raw survey vehicle sensory data for lane-level HD map generation

Matteo Bellusci *, Paolo Cudrano, Simone Mentasti, Riccardo Erminio Filippo Cortelazzo, Matteo Matteucci

Politecnico di Milano, Milano, 20133, Italy

ARTICLE INFO

Keywords:
HD mapping
Line markings recognition
Environment perception
Deep learning
Intelligent vehicles
Digital twins

ABSTRACT

High-definition (HD) maps provide a complementary source of information for Advanced Driver Assistance Systems (ADAS), allowing them to better understand the vehicle's surroundings and make more informed decisions. HD maps are also largely employed in virtual testing phases to evaluate the behavior of ADAS components under simulated conditions. With the advent of autonomous sensorized vehicles, raw machine-oriented data will be increasingly available. The proposed pipeline aims to provide a high-level semantic interpretation of raw vehicle sensory data to derive, in an automated fashion, lane-oriented HD maps of the environment. We first present RoadStarNet, a deep learning architecture designed to extract and classify road line markings from imagery data. We show how to obtain a semantic Bird's-Eye View (BEV) mapping of the extracted road line markings by exploiting frame-by-frame localization information. Then, we present how to progress to a graph-based representation that allows modeling complex road line markings' structures practically, as this representation can be leveraged to produce a Lanelet2 format HD map. Lastly, we experimentally evaluate the proposed approach in real-world scenarios in terms of accuracy and coverage performance.

1. Introduction

Although they require a non-negligible manual effort to be built, realistic high-definition (HD) maps have become a fundamental component in numerous autonomous driving projects [1,2]. Indeed, it is well-established that self-driving cars often rely on HD maps for accurate navigation, trajectory planning, and localization. In recent years, HD maps have also gained popularity as a foundation for the creation of digital twins [3,4]; by using HD maps as a basis, digital twins can be created to simulate and replicate real environments with a high level of realism [5]. Digital twins can be employed for various tasks including dataset generation, testing, and validation of algorithms before their deployment on real vehicles. Therefore, it is essential that these digital representations accurately capture all the relevant components of the real world, and the HD map on which the digital twin is built must be highly precise and a faithful representation of the environment.

The traditional method of manually generating these types of maps is very time-consuming [6], repetitive, and error-prone. In particular, human operators generally rely on imagery data to design the HD map, placing all road elements and core components by hand. While this task is relatively straightforward for traffic signs and lights, it is more challenging for road line markings. Operators often use aerial views of the area of interest to manually draw the road lines and assign a class to them; however, this process, besides being very time-consuming, requires the operators a careful placement of the lines.

To address these challenges, we propose a pipeline based on Artificial Intelligence (AI) for automatic lane-level HD mapping from data collected by surveying vehicles. The proposed pipeline requires just limited human intervention in the final stage of the process to validate the produced results. The pipeline is designed to take, as input, data from at least a monocular camera and precise Global Navigation Satellite System (GNSS) measurements in order to create an HD map in the standardized format Lanelet2 [7], which can be directly used in several autonomous driving systems. The generated map adheres to all Autoware [8] requirements for visualization. The proposed method leverages a custom network for line segmentation on the input images. Then, a semantic line map is reconstructed through a Bird's-Eye View (BEV) projection. This map is then processed using a graph-based approach to extract the horizontal road markings' structure, which is used to generate the final Lanelet2 HD map. A graphical representation

* Corresponding author.
E-mail addresses: matteo.bellusci@polimi.it (M. Bellusci), paolo.cudrano@polimi.it (P. Cudrano), simone.mentasti@polimi.it (S. Mentasti), riccardoerminio.cortelazzo@mail.polimi.it (R.E.F. Cortelazzo), matteo.matteucci@polimi.it (M. Matteucci).

https://doi.org/10.1016/j.robot.2023.104513

Available online 18 September 2023

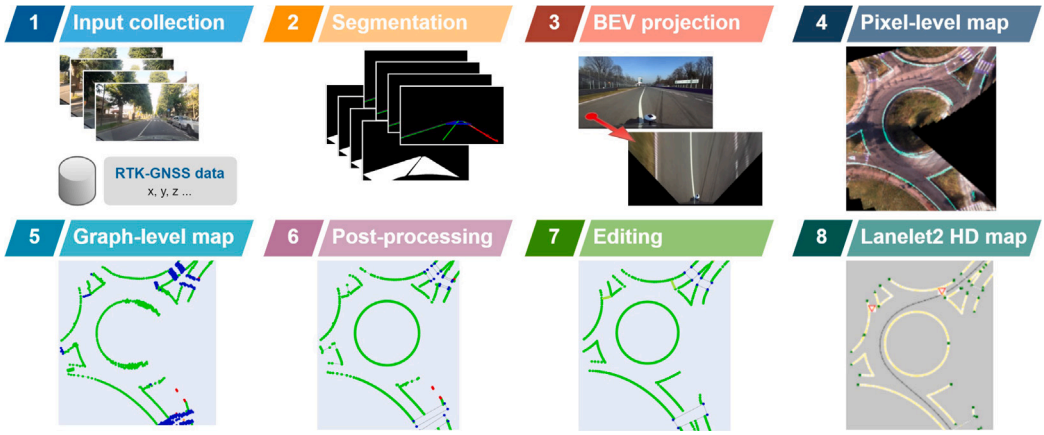


Fig. 1. High-level overview of the proposed pipeline.

of the proposed system is presented in Fig. 1. This pipeline represents a remarkable advancement in the automated generation of HD maps, showcasing an innovative methodology based on deep learning and sensor fusion. Preliminary work was conducted in [9].

The paper is organized as follows. Section 2 provides a comprehensive overview of related work and the current state-of-the-art in the field. In Section 3, we present our cutting-edge deep learning architecture, which uses vision data to obtain semantic information about road line markings, as well as an effective methodology for deriving BEV maps by leveraging precise GNSS data. Section 4 introduces an innovative approach for translating the semantic pixel-level aerial map into a graph-based semantic format, and Section 5 outlines a method for obtaining a valid Lanelet2 HD map format out of such a graph. In Section 6, we validate our approach through an experimental activity on three different datasets, assessing its accuracy, required time to derive precise HD maps, and result quality. Lastly, conclusions and future directions are discussed in Section 7.

2. Related work

Today, the generation of an HD map from images and precise GNSS data requires multiple steps. Of these, line detection and mapping are among the most critical ones. Line detection involves identifying and extracting information related to road line markings from camera images, while mapping converts this information into a meaningful, machine-readable, representation of the road layout. These steps are essential for the accurate and reliable generation of an HD map that can then prove effective in various autonomous driving applications.

2.1. Horizontal road markings extraction

Line detection involves identifying the pixels within an image that belong to the road lines. Traditionally, this process is conducted on the front-view images captured by the camera, although some studies have proposed BEV segmentation approaches [10]. One of the simplest methods for line detection is a color-based approach that involves thresholding the image channels to retrieve a binary mask of the lines [11]. This approach is typically combined with gradient-based filtering to ensure that only the pixels belonging to the lines are identified and not those with the same color. Specifically, the gradient-based approach takes advantage of the sharp contrast between the gray road and the white lines to locate them on the road. This contrast is exploited, for example, in [12]. Similarly, edge detection processing can be used to extract the image's edges and locate the road lines [13].

Despite achieving satisfactory results in simple scenarios, these methods encounter different challenges in complex environments such as urban roads. Consequently, Convolutional Neural Network (CNN)-based algorithms have recently gained popularity and have become the most commonly used solution for road line detection problems. Typically, these methods treat the task as a segmentation problem, either binary or multi-class, and process the data using single-pass encoder–decoder architectures such as U-Net [14] to classify each pixel of the input image. Besides the classic U-Net, other segmentation architectures, such as YOLOP [15], YOLOPv2 [16], and HybridNets [17], employ a single-pass encoder–decoder model to perform segmentation on the front-view image captured by a monocular camera. However, these models do not discern between different line types and only classify the pixels into three categories: line, drivable area, and background. This lack of specificity between line types is severely limiting when generating semantically rich HD maps for autonomous driving. In view of this, RMNet [18] has been designed to extract richer information and it proposes a multi-class road markings segmentation model, designed also to differentiate line types (e.g., dashed, solid line, etc.).

A common limitation of many structural approaches is the need to downsample the input high-resolution image in order to process it. This step is often required as processing high-resolution images is computationally intensive and time-consuming. However, downsampling leads to a reduction in the amount of information in low-resolution images compared to their high-resolution counterparts. To overcome this challenge, a novel approach has been recently proposed that pairs traditional CNNs with Region Proposal Networks (RPNs) [19] to identify areas within the original image where elements of interest may be present, and to process only those regions. To this end, Tian et al. [20] propose a variant of an instance segmentation region-based CNN specifically designed for road markings detection that achieves state-of-the-art performance while maintaining computational efficiency.

2.2. Map building

The process of line detection serves only as an initial step within a map generation pipeline. Indeed, to achieve the creation of an accurate map, the detections must be transformed to build a standardized HD map format. This can be accomplished through a variety of heterogeneous techniques. Zhou et al. [21] employed Open Street Map (OSM) data, a front-view camera, and a Light Detection and Ranging (LiDAR) sensor in order to generate an HD map. The method leverages OSM information to establish a prior on the road structure, and it combines such structure with camera and LiDAR data to improve the precision of the resulting HD map.

Notably, the use of LiDAR data has become increasingly prevalent in recent years in the HD map processing pipeline to enhance the accuracy of camera detections. As the standard camera-based BEV projection assumes the road to be a planar surface, the LiDAR data can be employed to obtain more accurate BEVs. HDMapNet [22], a CNN model

that can fuse camera and LiDAR data, has been proposed to predict vectorized map elements in BEV. Homayounfar et al. [23], instead, exploit aggregated LiDAR intensity images to obtain a directed acyclical graph of the road lane boundaries. Recently, Liu et al. [24] focused on the generation of lane-level road intersections maps from low-channel roadside LiDARs. Ye et al. [25], instead, discuss the advantages and issues encountered when relying on a mobile laser scanning system to retrieve precise line marking points and the road centerline.

Recently, researchers have also explored the possibility of extracting valuable mapping information from images of the area of interest captured from an elevated perspective. For instance, digital maps can be built by exploiting satellite imagery data [26], although it is not easy to provide a high level of precision. Zang et al. [27] explore the use of satellite imagery to automatically extract road lane boundaries. Similarly, He and Balakrishnan [28] propose a mapping pipeline that automatically extracts lane-level street maps from aerial imagery.

2.3. External information sources

Another essential sensor in the mapping process is a localization device such as GNSS. High accuracy is required to combine and map sequences of images captured by the vehicle, often necessitating the use of Real-Time Kinematics (RTK) correction to mitigate small position errors stemming from the GNSS sensor. Recent approaches have introduced the use of a graph structure to combine camera images, perform loop-closure, and graph optimization. Notably, [18,29] both utilize a graph-based structure to represent the relative position of features and make adjustments to enhance the overall accuracy of the map. A similar graph structure was previously used also by Máttyus et al. [30], who integrated their data with aerial and ground images to extract further semantic information.

Instead of using a single instrumented vehicle, Zhou et al. [31] rely on crowdsourced data to extract lane information. In particular, they exploit the data coming from a fleet of vehicles, each one equipped with a low-cost GNSS and a camera. In their proposed pipeline, line detection is performed at the vehicle-level, its output is then aggregated, and the resulting line information fitted with B-splines in a centralized manner. In [32], authors exploit crowdsourced data to derive a graph-based structure of the lane markings, which geometry is then optimized with domain knowledge. Other crowdsourced techniques include the one proposed by Liebner et al. [33], who obtain HD map patches with a method based on road model inference and graph-based Simultaneous Localization and Mapping (SLAM). Kim et al. [34], instead, propose to exploit public transport vehicles equipped with low-cost sensors to maintain HD maps up-to-date through time.

3. Semantic pixel-level map extraction

The automatic generation of an HD map involves heterogeneous methodologies to extract road line markings from vehicle sensor data. While extracting road line marking information from LiDAR data is feasible, as evidenced by recent studies [35], the conventional approach entails retrieving such information from images captured by the vehicle's camera. This is because camera images provide highly informative attributes about the road surface, including color and texture, which can be harnessed to detect and classify road line markings. Currently, this is being addressed by deploying automatic learning techniques, especially deep learning approaches; indeed, recent breakthroughs in AI, specifically in the realm of deep learning, have transformed how road line markings extraction is carried out. Deep learning techniques have demonstrated remarkable efficacy in accurately and efficiently identifying and classifying road line markings, even in demanding weather and lighting conditions. These techniques typically involve training a neural network leveraging a vast dataset of labeled images, enabling the network to learn the visual features of diverse road line markings, and how to differentiate road line markings from other objects and background elements.

3.1. RoadStarNet architecture

In the context of deep learning architectures, single-task and multi-task CNNs emerge as distinct paradigms. Leveraging the interdependence of different tasks, multi-task networks such as YOLOP [15], YOLOPv2 [16], and HybridNets [17] have generally proven to be superior to single-task models. By training a model to perform multiple tasks simultaneously, the model acquires supplementary knowledge and can better capture intricate relationships between the tasks. In the context of the models mentioned above, which undertake line detection, drivable area segmentation, and object detection, the three tasks are targeting the detection of road elements and the segmentation of road scenes. Consequently, the model can employ the features shared between the tasks to enhance the accuracy of each task. Following this underlying idea, in this study, we propose a novel deep learning-based architecture for extracting simultaneously road line markings, their classes, and the drivable area. As further detailed later, the architecture is composed of a single shared encoder and two decoders; one is dedicated to the multi-class segmentation of road line markings, while the other focuses on extracting the drivable area. Note that, while mentioned state-of-the-art CNNs (i.e., [15–17]) deal also with tasks such as vehicle object detection, our aim is to devise a CNN designed to be suitable specifically for lane-level HD map generation, which also handle multi-class line segmentation.

The proposed CNN model architecture is inspired by HybridNets, a single-pass end-to-end network architecture. HybridNets encoder–decoder structure uses as backbone EfficientNet-B3 [36], alongside a neck network with a weighted Bi-directional Feature Pyramid Network (BiFPN) module [37], and two separate decoder blocks. One decoder block is dedicated to the object detection task, while the other is dedicated to the segmentation. In HybridNets segmentation decoder, the outputs of the BiFPN modules are brought to a common scale through a series of convolutions and upsamples to be finally combined and sent to a segmentation layer.

In this work, we propose RoadStarNet, a novel CNN for multi-class road line markings segmentation. We employ the same backbone encoder as HybridNets, adopting two separate decoders, one dedicated to line markings and an additional one to identify the drivable area. Indeed, we found that the latter, although not generating outputs of direct interest for our tasks, significantly improves the performance of our line detection head through the advantages of multi-task learning. Our two decoders are inspired by U-Net [14] to better exploit the multi-dimensional features extracted by the BiFPN block. We also introduce two skip connections from the backbone to the middle of our decoders. Incorporating skip connections helps preserve more spatial information that may be lost in the BiFPN's features fusion and allows for better back-propagation. The high-level architecture overview of the proposed CNN is shown in Fig. 2. Multi-scale feature fusion is achieved by combining feature blocks with different dimensions. Each feature block is upsampled to match the input size of the next block, and fed with a convolutional layer; a Sigmoid Linear Unit (SiLU) [38] is used as activation function, and a batch normalization operation is applied at alternate levels. To mitigate unwanted noise in the output of RoadStarNet, we post-process its output filtering all clusters smaller than a fixed threshold.

3.2. Training loss functions

We use two distinct losses to train our RoadStarNet architecture: Focal* and Focal+Tversky. Focal* is a generalization of the Focal loss [39] that weighs each class differently, while Focal+Tversky is a weighted sum of the Focal and Tversky [40] losses. Both losses aim to minimize the difference between the predicted and ground truth segmentation masks.

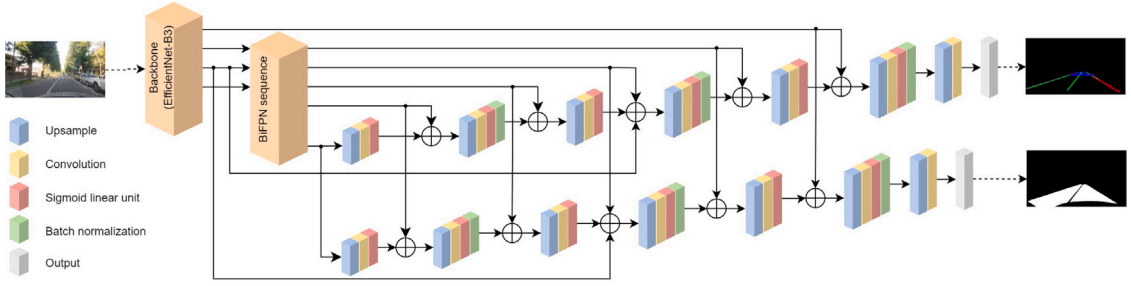


Fig. 2. RoadStarNet simplified architecture overview.

The Focal* loss used to train RoadStarNet (referred to in the following as RoadStarNet-F*) is defined by:

$$L_{F*}(\hat{\mathbf{y}}, \mathbf{y}) = -\frac{\alpha}{N} \sum_{c \in \mathbf{C}} \sum_{i=1}^{N} w_c \cdot \mathbf{y}_{ci} \cdot \hat{\mathbf{y}}_{\bar{c}i}^{\gamma} \cdot \log \hat{\mathbf{y}}_{ci}, \tag{1}$$

where $\mathbf{C}$ is the set of classes, N is the number of image pixels, $\mathbf{y}_{ci}$ is the ground truth label for pixel i in class c, $\hat{\mathbf{y}}_{ci}$ is the predicted probability of pixel i being in class c, w_c is the weight for class c, the subscript $\bar{c}$ denotes not belonging to class c, α is a balancing parameter and γ is the tunable focusing parameter. Note that the classical Focal loss $L_F(\hat{\mathbf{y}}, \mathbf{y})$ is similar to $L_{F*}(\hat{\mathbf{y}}, \mathbf{y})$ but has no class weight balancing parameters, i.e., $w_c = 1, \forall c \in \mathbf{C}$. By assigning weights to all classes, it is possible to force the CNN to give more attention to the lines, for instance by decreasing the weight of the background class.

The Focal+Tversky loss, from [17], used to train RoadStarNet (referred to in the following as RoadStarNet-FT) is defined by:

$$L_{F+T}(\hat{\mathbf{y}}, \mathbf{y}) = L_T(\hat{\mathbf{y}}, \mathbf{y}) + \xi \cdot L_F(\hat{\mathbf{y}}, \mathbf{y}), \tag{2}$$

where $L_F(\hat{\mathbf{y}}, \mathbf{y})$ is the Focal loss, $L_T(\hat{\mathbf{y}}, \mathbf{y})$ is the Tversky loss, and ξ is a hyper-parameter that controls the relative importance of the two loss terms. The Tversky loss is effective in handling class imbalance and allows the trade-off between false positives and false negatives to be adjusted. The Tversky loss L_T is defined as:

$$L_T(\hat{\mathbf{y}}, \mathbf{y}) = \sum_{c \in \mathbf{C}} \left(1 - \frac{\sum_{i=1}^{N} \hat{\mathbf{y}}_{ci} \mathbf{y}_{ci}}{\sum_{i=1}^{N} \hat{\mathbf{y}}_{ci} \mathbf{y}_{ci} + \alpha \sum_{i=1}^{N} \hat{\mathbf{y}}_{ci} \mathbf{y}_{\bar{c}i} + \beta \sum_{i=1}^{N} \hat{\mathbf{y}}_{\bar{c}i} \mathbf{y}_{ci}} \right), \tag{3}$$

where $\hat{\mathbf{y}}$ and $\mathbf{y}$ are the predicted and ground truth pixel-level data, respectively. The Tversky index [41] measures the overlap between the predicted and ground truth segmentation masks for class c, while α and β are pre-defined constants that balance the importance of false positives and false negatives.

3.3. BEV pixel-level map

In order to generate accurate HD maps, survey systems generally collect data from different mapping sensors in a synchronized manner. The camera system captures a stream of images, while RTK-GNSS data contains information about the position of the vehicle. The fusion of these data, together with inertial system data, enables the estimation of the camera's absolute location and orientation when capturing each image. Inverse Perspective Mapping (IPM) is then used to map the image pixels into a relative planar coordinate system to obtain a BEV of the scene [42]. The IPM model leverages a homography projection matrix that mathematically describes the relationship between the frontal and rectified views [43]. Intrinsic and extrinsic camera parameters are needed to define such homography matrix, where the intrinsic parameters depend on the type of camera and are generally constant. Conversely, the extrinsic parameters are dynamic and can be estimated using vision-based calibration algorithms or additional sensors' data [44,45]. By combining the IPM model with the camera's position and orientation, the predicted road line marking pixels can be mapped in the global world reference system. Images from several cameras, when available, can be combined for more precise and complete outputs.

The proposed method for generating BEV pixel-level maps, which idea is simplified in Fig. 3, is designed to be both modular and extendable, providing flexibility and adaptability to different surveying scenarios. We break the map into small chunks, and this allows us to save memory and reduce RAM consumption, while still maintaining the required level of detail. The first step of our method involves calculating the number of map blocks, or chunks, that will be needed to cover the surveyed area. Each chunk is associated with a selection of camera images whose vertices fall within its corresponding space domain. Using RTK-GNSS data and the IPM model, we are indeed able to project the vertices of each image onto the absolute planar system to accurately determine the size and location of each chunk. Once the chunks have been defined, we can project all pixels of the camera frames into the corresponding chunk of the map. During this projection phase, road line marking pixels are augmented with their estimated road line marker class, allowing a more accurate representation of the road layout.

In the end, the map chunks can be combined to derive the full map of the surveyed area. This modular approach not only reduces the computational burden but also provides the flexibility to adjust the size and number of chunks based on the specific needs of each surveying task. Moreover, our method allows for future map expansion, as additional chunks can be added as needed to incorporate new data or extend the area covered by the map. This extendable approach makes our map generation method particularly well-suited for large-scale surveying projects, where a flexible and scalable solution is sought.

4. Graph-level map extraction

This section outlines the transition from a pixel-level BEV to a more practical and efficient graph-based model for representing road line markings. Indeed, a BEV pixel-level model is often not a practical model to be integrated with other components. Data interpretation is less efficient than other analytical models or models based on precise data structures, as road line markings are represented sparsely by many pixels and noise tends to be always present as it is inherited from the network output and the projection. For this reason, we propose to use a semantic colored graph model representation, which is more practical and easier to handle, as well as cleaner, indeed it reduces noise and requires much less memory for storage. The idea indeed is to progress from a set of georeferenced pixels to a graph in which the road line markings are represented by a series of points belonging to road lines and connected by edges.

4.1. Graph-level mapping algorithm

We present our algorithm as an evolution of the window-based line following (WLF) algorithm presented in [46], in which a double search window locally follows pixels indicating the presence of road

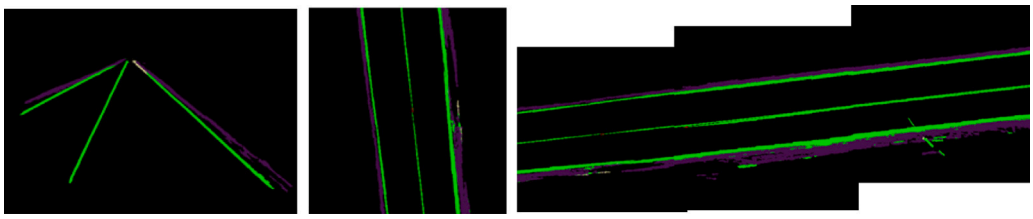


Fig. 3. From left to right: segmented front camera frame, its BEV projection, and derived pixel-level map by combining 3 consecutive BEV frames.

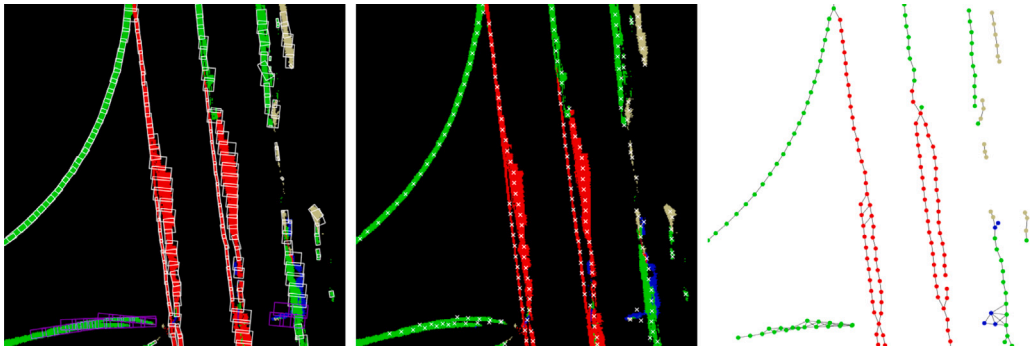


Fig. 4. From left to right: pixel-level map with extracted bounding boxes, extracted centroids, and derived graph connections where each color represents a different class of road line markings.

line markings to obtain a set of distinct points along the direction of the curves. Our algorithm works similarly, but firstly it does not operate frame-by-frame, thus resulting in an efficient solution for modeling the skeleton of road line markings of large maps in fewer steps. Furthermore, whereas the WLF algorithm only allows modeling the two side lines locally, our approach allows modeling all the different types of road line markings at the cost of a small approximation. Such approximation is due to the fact that a graph can at most model a curve by means of a polyline of first-order segments due to the nature of edges, which are intended as segment-level connections of two nodes.

Let us consider a region of interest (ROI) extracted from the segmented BEV pixel-level map starting from a given RTK-GNSS position of the survey vehicle; to perform the conversion into a graph format, our pipeline first locates the vertices of the graph. To do this, a window-based algorithm is employed: following the trend of each road line using a scan window, the algorithm extracts a series of bounding boxes whose size is dynamically adapted. The search starts in the direction of the vehicle. The centroid of each bounding box defines a node of the graph, while the connections are established when two or more bounding boxes overlap each other. For each of these groups, a cluster that defines a particular road line marking is defined, and nodes are connected according to their relative distances and by exploiting the bounding boxes' overlaps to build the set of edges. In addition, the class to which each vertex belongs is also extracted based on the type of road line marking it represents (e.g., dashed double line). Types are determined based on the pixels' classes around the node, by selecting the most frequent class. At the end of the process, we obtain a semantic colored graph $G = (V, E, C, f_c)$, where V is the set of vertices, E is the set of edges, C is the set of colors representing the different classes, and $f_c : V \rightarrow C$ is a function that maps each vertex to an associated color. Fig. 4 summarizes the steps of the pipeline just described.

From a set of RTK-GNSS vehicle positions, a local graph can be derived for each associated ROI. It is possible to merge the graphs dynamically by exploiting the absolute positions of vertices and any areas of intersection between graphs. Indeed, if a set of successive positions is close enough, a typical situation occurs in which the same portion of a road line marking is common to both graphs. As multiple bounding box groups cover the same intersection area (in successive frames), the information can be used to fine-tune the positions of already extracted vertices and to establish new edges to connect possible new nodes. Intuitively, it is possible to achieve complete map coverage and automatically derive a single global semantic colored graph $G = (V, E, C, f_c)$.

4.2. Rule-based post-processing techniques

A series of rule-based post-processing techniques are now proposed to clear the output graph and correct frequent inaccuracies that may appear during the autonomous graph-building procedure. Additionally, these methods help in reducing inaccuracies or inconsistencies that may have resulted from the data collection process. Fig. 5 illustrates the main post-processing techniques and their effects on the starting input graph.

Line fitting. The proposed post-processing operation fits nodes belonging to the same road line with a curve. Road line nodes are identified easily as the set E already defines connections between vertices V. The relative axis with the highest variance is identified and nodes, treated as points, are fitted with a polynomial curve using the Least Squares (LS) algorithm. With this operation, nodes can be rearranged according to the curve shape to reduce distribution sparsity and noise within the graph; this is achieved by sampling a set of nodes from the curve and by establishing connections between them in accordance with the original configuration defined by $G = (V, E, C, f_c)$.

Circle fitting. This post-processing technique is designed to improve roundabout modeling by fitting and replacing entire sub-graphs with circumferences. It creates smooth roundabouts without distortions and it can even fill in missing parts. This process involves the use of the Random Sample Consensus (RANSAC) algorithm [47], which is a non-deterministic algorithm. At each iteration, a circumference is fitted to three randomly sampled sub-graph nodes, ignoring edges. The algorithm then computes the number of outliers that are too far away (above a threshold) to select or reject the fitted circumference. Selected circumferences are then compared using the mean distance to all the points. The circumference with the minimum average distance is used to replace the sub-graph with a new sub-graph sampled from the detected circumference shape.

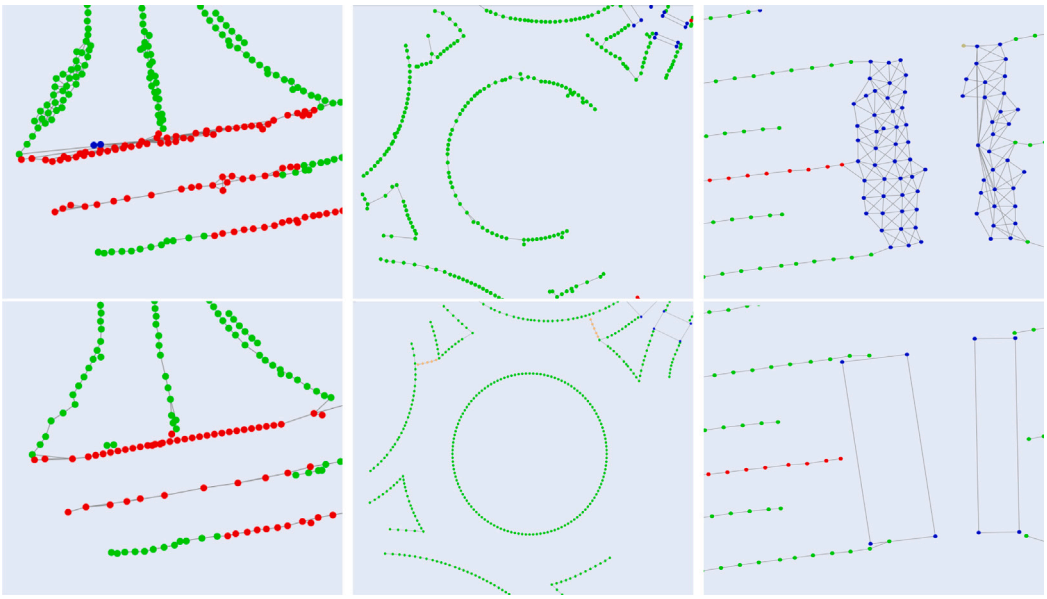


Fig. 5. Examples of rule-based post-processing. The first row displays graphs depicting selected ROIs before applying any post-processing method. The second row (post-processed graphs), from left to right, visually demonstrates the effects of line fitting, circle fitting, and box replacing techniques on the above associated graph.

Box replacing. We propose also a post-processing operation to simplify the graph $G = (V, E, C, f_c)$ in areas where nodes represent road markings that are better identified by nodes arranged in a polygonal shape. In the case of a pedestrian crossing, for instance, it is often sufficient to identify the area that indicates the crossing rather than modeling the zebra crossing lines. The proposed method searches for groups of nodes that form pre-defined polygonal areas and replaces them with a smaller number of nodes that identify the boundaries of the areas of interest.

Additional techniques. Our post-processing pipeline also includes different methods for operations that smoothen the overall output, including joining line sub-graph patches with discontinuities, removing small sub-graphs with a low number of nodes, or replacing very short cycles of nodes with their midpoint. The proposed suite of methods is extensible and methods can be combined to provide a more homogeneous and smooth output.

These proposed techniques for post-processing can be combined depending on the reference domain. For instance, if there are no roundabouts in the area where data is collected, it may not be necessary to use the circle fitting technique. These methods are invoked sequentially, as they are substantially independent. At the end of this process, a manual intervention may be still required to correct the graph to obtain an error-free output. As the output format is particularly simple, a simple viewer can be used to display and manipulate the graph as desired, such as by moving or by deleting nodes. Modifying the graph requires much less time than manually designing it from scratch, making our pipeline particularly suitable for deriving HD maps in an effective, precise, and fast manner.

5. Lanelet2 format extraction

HD maps contain various information about the structure of roads, their morphology, and their connections. There are several formats currently in vogue for representing road-level information in a standardized manner. In this section, we propose to model the output of our system according to the Lanelet2 format [7]. Although various alternatives exist, this format is simple and suitable for the collection of the information extracted with our pipeline.

In Lanelet2, it is possible to model road line markings by directly exploiting the previously extracted semantic colored graph $G = (V, E, C, f_c)$. Indeed, Lanelet2's topological model and relational model are based on primitives including points and linestrings, which form the physical model. A linestring is defined from an ordered list of nodes (vertices) linearly interpolated, which defines a set of connections (edges). Since each linestring has a tag defining its associated type, graph semantic coloration plays a crucial role. Indeed, when a class change occurs or a line splits (i.e., when a node $v \in V$ has $\deg(v) > 2$), the list of nodes is split, and multiple linestrings are created to model the portion of the graph under examination. The analyzed components serve to determine a number of fundamental elements of the road within the Lanelet2 format, such as lanes.

Lanes in Lanelet2 format are defined via the use of lanelets, which are atomic lane sections, i.e., portions of lanes where traffic regulations remain consistent. Each lanelet is intuitively composed of two linestrings that define its lateral borders and which comprise the drivable area. To maintain a similar and consistent number of lanelets across lanes in multi-lane roads, we devise a split propagation algorithm. Whenever a linestring split occurs at a certain node $v \in V$, a perpendicular line is derived to obtain the closest nodes, called propagated nodes, in other linestrings within the same, eventually multi-lane, road (a local search using a scan box is applied). An example is shown in Fig. 6. Split and propagated nodes are exploited to establish and define the lanelets that covers the underlying road, so as to be able to guarantee the atomic property of the lanelet elements. In particular, two linestrings are intuitively paired to form a lanelet when start and end nodes are related to the same graph cuts. Also, other types of road markings can instead be modeled within Lanelet2 as polygons or areas (e.g., parking slots).

Note that the Lanelet2 format requires an elevation measure for each point. While such a measure can be set to zero, in the proposed pipeline, we associate each node of the graph with an estimated elevation data. Specifically, for each node $v_i \in V$, we estimate its elevation z'_i from a set of RTK-GNSS positions $\mathbf{G}_i$ close (in the plane) to node v_i via Inverse Distance Weighting (IDW):

$$z'_i = \left(\sum_{(x_j, y_j, z_j) \in \mathbf{G}_i} \frac{1}{d_{ij}^{\gamma}} \right)^{-1} \cdot \sum_{(x_j, y_j, z_j) \in \mathbf{G}_i} \frac{z_j - h}{d_{ij}^{\gamma}} \tag{4}$$

where γ is a positive pre-defined constant, h is the height of the RTK-GNSS sensor from the ground, and d_{ij} is the associated flat distance between node v_i and (x_j, y_j). When there are no neighboring RTK-GNSS

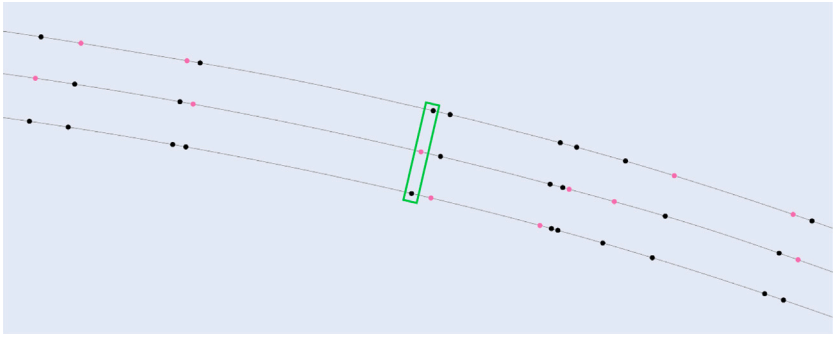


Fig. 6. Example of split propagation algorithm output; linestring cuts occurred at pink nodes, while black nodes represent the propagated nodes. In green, an example of a scan box used by our cut propagation algorithm to identify propagated nodes.

positions, the node elevation is estimated from the k-nearest nodes' elevations.

6. Experimental results

Experiments were carried out using a computer featuring an Intel® Core™ i7-3770K processor and a NVIDIA® GeForce® GTX 970 GPU, with all algorithms implemented in Python. All networks have been trained on a custom and modified version of the Berkeley Deep Drive 100k (BDD100k) dataset [48], which offers a series of annotated images with road line markings and drivable area labels, based on [15]. Our networks have been trained using Adam optimizer [49] with a learning rate of 10^{-4} and batch size of 4. To verify the mapping capabilities of the proposed approach, we utilized a dataset obtained from a moving vehicle equipped with a ZED stereo-camera and a Swiftnav RTK-GNSS, which was captured on the Monza Eni Circuit track (for more information, please refer to [46,50]). The dataset includes ground truth data for the position of road line markings for the entire 5.8-km-long circuit. Furthermore, to evaluate the pipeline more effectively, we divided the circuit into various ROIs that present significant challenges for the mapping system due to their non-straightforward road structures, such as fast direction changes and long curves. Circuit sections are highlighted in Fig. 7. We also qualitatively tested the proposed pipeline in an extra-urban setting in Milan (Italy), for which we collected data using the same survey vehicle. Compared to the race circuit track, this setting features different challenging characteristics, such as multi-lane roadways. In addition, we provide examples of qualitative results achieved on an urban-like dataset captured in Tavagnacco (Italy). In this case, the experimental vehicle was equipped with a SITECO Road-Scanner C mobile mapping system, including a FLIR® LadyBug®5 spherical camera system. In contrast to the circuit dataset, the Tavagnacco dataset did not contain precise ground truth line positions, but LiDAR point cloud data were available, which can be exploited to visualize the overall output results more comprehensively.

6.1. Pixel-level map assessment

We assessed the effectiveness of our mapping pipeline using two distinct metrics. The first metric, called mean prediction distance (Dist.), calculates the average distance between all predicted line pixels and the center of the related ground truth line. It should be noted that this metric overestimates the error due to the fact that the road line markings have a width that is not accounted for in the ground truth annotation. A second metric we considered simultaneously is coverage (Cov.), which measures the linear percentage of the total road line markings that were appropriately mapped. We compared our method, using the two different proposed training losses, against state-of-the-art neural networks for road line markings identification, such as YOLOP [15], YOLOPv2 [16], and HybridNets [17]. We refer to the two proposed models as RoadStarNet-F* (trained with L_{F*}) and RoadStarNet-FT (trained with L_{F+T}). Each weight w_c ($c \in \mathbf{C}$, where $\mathbf{C}$ is the set of classes) of the L_{F*} loss used for training RoadStarNet-F* was set to 1 except for the weight of the background class, which was set to 0.1 to allow the model to focus more on the lines. Note, however, that the weights can be fine-tuned according to the reference scenario. In fact, this is a feature of RoadStarNet-F* that allows the user to select weights in a way that moves the focus of the network to particular types of classes over others. Regarding RoadStarNet-FT, we have set $\xi = 1$ in order to equally balance the two components L_F and L_T. The quantitative results obtained for the pixel-level mapping phase are summarized in Table 1.

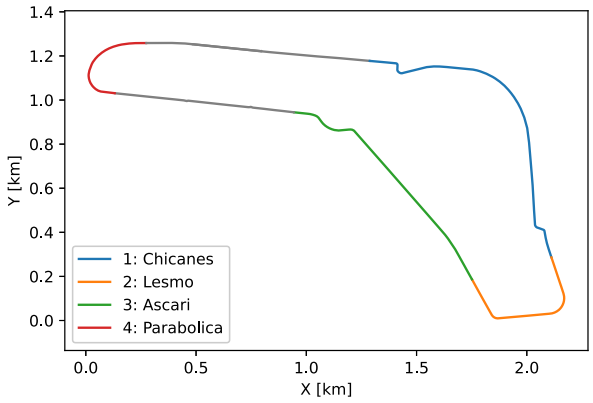


Fig. 7. ROIs considered for the Monza Eni Circuit track.

Our proposed RoadStarNet-F* achieves consistently the highest coverage, at the cost of a slightly higher mean prediction distance (Dist.). Compared to the highest coverage model in the state-of-the-art, i.e., YOLOPv2, our network still performs better in prediction distance in different sections with a slightly improved coverage. In terms of high-coverage requirements, our model achieves the best trade-off among the considered state-of-the-art methods. RoadStarNet-F* results to be more suitable for semantic aerial mapping of road line markings due to its fine-tuning capability and its ability to identify lines more decisively, albeit at the cost of potentially finding more false positive pixels. On the other hand, our RoadStarNet-FT model showed a more relaxed trade-off in coverage, achieving low prediction distances while retaining good and robust coverage capabilities. Therefore, it is a reasonable alternative to RoadStarNet-F* when a reduction in coverage is acceptable. Finally, it can be seen that HybridNets achieved

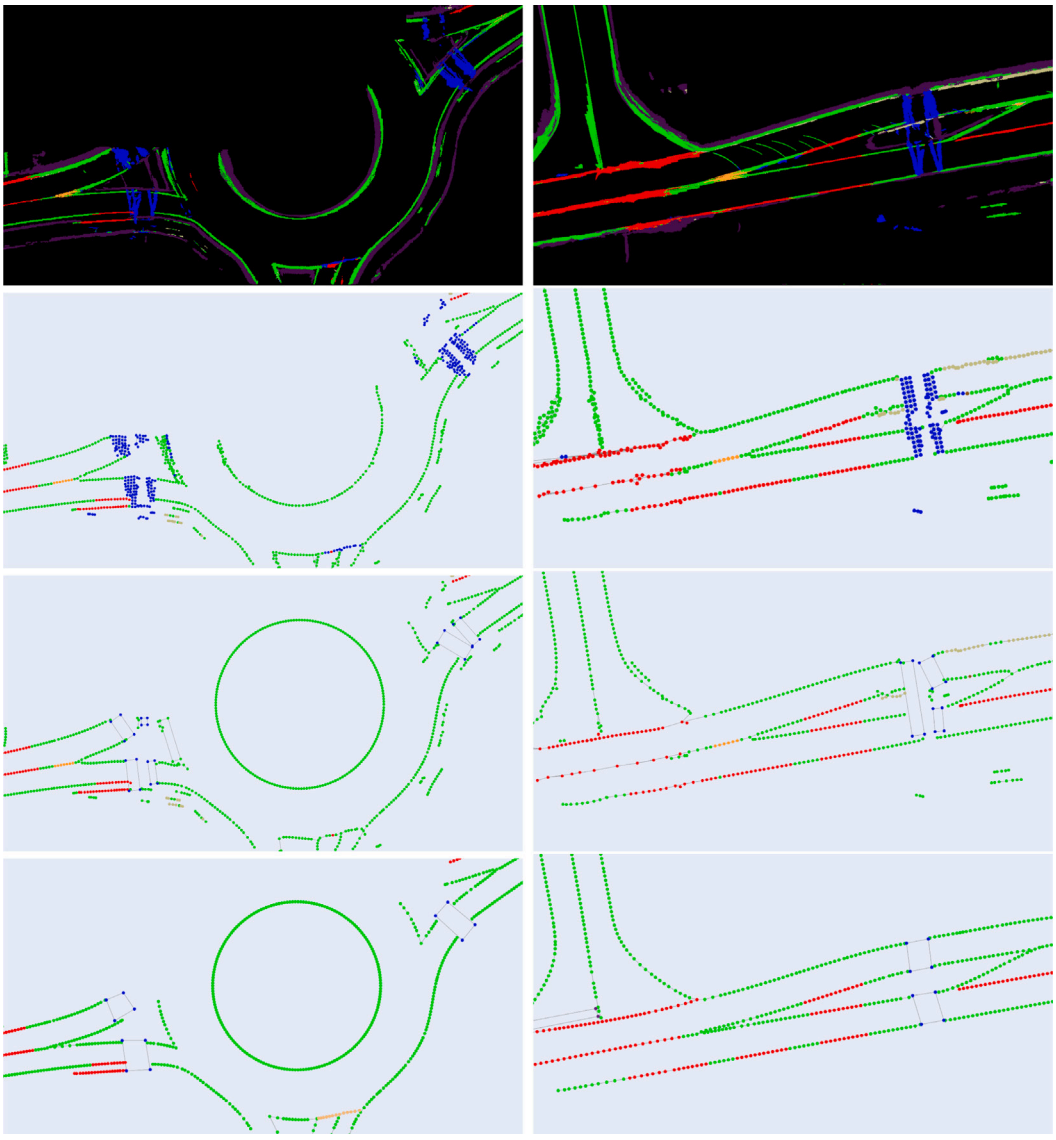


Fig. 8. Examples of qualitative results obtained from the urban dataset by converting the pixel-level map (1st row) into a graph-level map: raw (2nd row), post-processed (3rd row), and edited (4th row). Roundabouts are only partially visible in the original graph as the survey vehicle path did not cover their entirety.

Table 1
Mapping error distance and coverage on sections of the Monza Eni Circuit pixel-level map. For a comprehensive insight, rows are arranged according to the considered metrics' values.

	Monza Eni Circuit track							
	1: Chicanes		2: Lesmo		3: Ascari		4: Parabolica	
	Dist. (m)↓	Cov. (%)↑	Dist. (m)↓	Cov. (%)↑	Dist. (m)↓	Cov. (%)↑	Dist. (m)↓	Cov. (%)↑
RoadStarNet-F*	0.4047	**98.73**	0.6277	**99.96**	0.5457	**99.82**	0.4750	**98.90**
YOLOPv2 [16]	0.3723	94.56	0.7340	98.77	0.5955	98.07	0.4545	97.58
YOLOP [15]	0.2933	86.73	0.4222	90.03	0.4162	93.38	0.4074	78.00
RoadStarNet-FT	**0.2872**	83.64	0.4222	83.69	0.3752	90.20	0.3842	80.30
HybridNets [17]	0.3163	85.49	**0.3236**	61.69	**0.3377**	64.76	**0.3109**	64.70

a much lower coverage rate than the other methods, thus mitigating the advantage of achieving a lower pixel placement error.

6.2. Graph-level map assessment

To quantitatively evaluate the graph-based map, we evaluated the average error (in meters) and the coverage percentage for graph vertices. Different versions of the graph were considered, starting with the raw version obtained without applying any post-processing techniques. Then, the aforementioned metrics were evaluated using the post-processed graph map and, finally, the manually edited graph map (within a given correction time limit). In particular, we used our manual editing tool on the graph map in its post-processed state and measured the metrics after 15, 30, 45, and 60 minutes of manual correction. These time intervals represented edits of 6.65%, 13.43%, 20.82%, and 31.82%, respectively, of the initial 10121 nodes. Results are summarized in Table 2. The results indicate that even with a

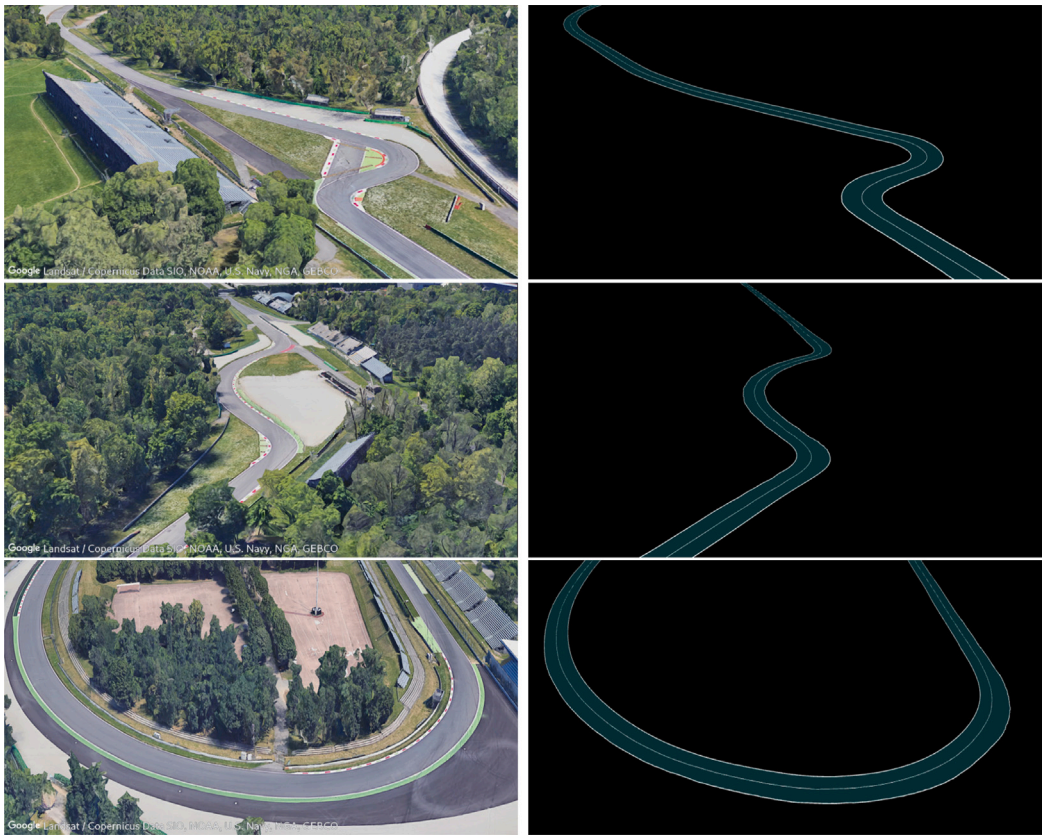


Fig. 9. Examples of qualitative results (right) obtained by projecting obtained Lanelet2 model of the Monza Eni Circuit track via Autoware.AI and RViz.

Table 2
Average node-ground truth distance and coverage on sections of the Monza Eni Circuit graph-level map generated using RoadStarNet-F*. Time values are expressed in minutes.

	Monza Eni Circuit track							
	1: Chicanes		2: Lesmo		3: Ascari		4: Parabolica	
	Dist. (m)↓	Cov. (%)↓	Dist. (m)↓	Cov. (%)↓	Dist. (m)↓	Cov. (%)↓	Dist. (m)↓	Cov. (%)↓
Raw	0.4338	98.52	0.6673	98.93	0.5659	99.39	0.4495	99.18
Post-processed	0.3293	99.33	0.4628	97.32	0.4235	99.49	0.3698	99.20
Edited in 15'	0.2966	99.86	0.3029	98.52	0.4012	99.75	0.3659	99.79
Edited in 30'	0.2966	99.86	0.2956	98.52	0.3408	99.87	0.3968	99.85
Edited in 45'	0.2788	99.86	0.2513	98.76	0.3188	99.91	0.3968	99.85
Edited in 60'	0.2571	99.76	0.2513	98.76	0.3188	99.91	0.3923	99.85

short manual intervention, it was possible to obtain a complete map of the circuit with high precision and a high percentage of coverage. Indeed, while the proposed pipeline achieves effective accuracy and coverage capabilities in the mapping phase, some inaccuracies may still occur, requiring manual corrections to fine-tune the overall output. Furthermore, it is important to note that the proposed pipeline is limited to mapping road line markings and lanes, so integrating our algorithm's capabilities with others that can identify other road elements (e.g., traffic signs) will be necessary for a comprehensive representation of the surveyed environments.

In addition to the quantitative results, we also qualitatively evaluated the graph map conversion in critical selected road sections (Fig. 8) and the 3D positioning of the lanelets by exploiting the Tavagnacco urban dataset which contains also processed LiDAR point cloud data. From Fig. 8, it can be noted that the graph conversion pipeline is effective, and the post-processing techniques are capable of cleaning and smoothing the initial output while correcting some inaccuracies present in the raw, non-post-processed graph. Next, a manual editing allowed in a short time to refine the graph to make it more accurate and reduce noise. Regarding the 3D positioning of the lanelets, using Autoware.AI and RViz, it is possible to visualize a projection of the Lanelet2 model obtained with our pipeline on the global point cloud of the navigation environment. Examples of these qualitative results are shown in Fig. 11, from which it can be seen that our pipeline has effective capabilities for positioning road line markings and lanes. Moreover, additional qualitative results from the Monza Eni Circuit track are shown in Fig. 9. Lastly, we also evaluated the placement of the lanelets on the extra-urban Milan dataset we collected; despite the presence of a large number of (long) laneways in the environment, the graph-level map and final reconstruction are accurate and well overlaid with the background aerial view, as shown in Fig. 10.

7. Conclusions

In this work, we presented a novel semantic-oriented pipeline to derive lane-level HD maps starting from common sensors on a survey vehicle. Sensors provide vehicular and surrounding environment information in a numeric, machine-oriented format; to interpret this data, we first introduced RoadStarNet, a deep learning model that detects

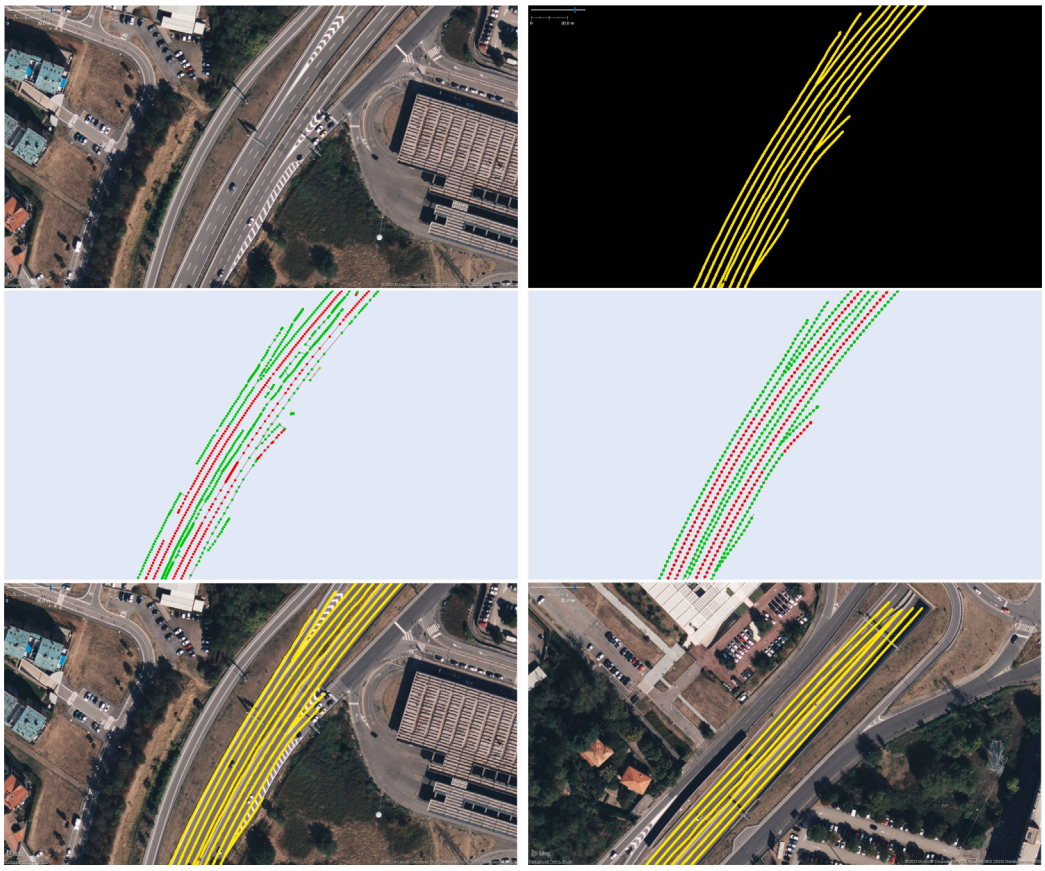


Fig. 10. Examples of qualitative results showing our final reconstruction of the surveyed extra-urban area of Milan. In order: aerial view of the section of interest, reconstructed Lanelet2 model, post-processed graph, and final edited graph; on the last row, we showed an overlay of the model with the associated aerial background (left), as well as a similar result on a different (subsequent) section of the area (right).

Fig. 11. Examples of qualitative results obtained by projecting obtained Lanelet2 model on the Tavagnacco dataset point cloud via Autoware.AI and RViz.

and classifies road line markings from imagery data. We then fused RKT-GNSS information to create a semantic BEV of the road line markings and represent it as a semantic colored graph to model complex structures. This representation is then used to create an HD map in the Lanelet2 format. Our pipeline represents a remarkable and significant advancement in the automated generation of HD maps, showcasing an innovative methodology based on deep learning and sensor fusion. We assessed our approach in both urban and extra-urban scenarios, where it demonstrated effective accuracy and coverage capabilities.

There are many opportunities for future work to further advance the creation of HD maps, as well as the development of novel techniques aimed at enhancing the proposed pipeline's effectiveness. In particular,

upcoming future work will focus on developing methods for the integration of other sensor data, such as LiDAR and radar data, to provide a more complete representation of the environment, while including also different road elements into the reconstructed map, such as road signs or traffic lights.

Declaration of competing interest

The authors declare the following financial interests/personal relationships which may be considered as potential competing interests: Matteo Matteucci reports financial support was provided by European Union NextGenerationEU program (PNRR); Matteo Matteucci reports financial support was provided by AnteMotion S.r.l.

Data availability

The data that has been used is confidential.

Acknowledgments

This paper was supported by “Sustainable Mobility Center (Centro Nazionale per la Mobilità Sostenibile – CNMS)” project funded by the European Union NextGenerationEU program within the PNRR, Mission 4 Component 2 Investment 1.4. Also, this work was supported by AnteMotion S.r.l., Trento, Italy. Any opinions, findings, conclusions, or recommendations, either expressed or implied, in this material are those of the authors and do not necessarily reflect the views of the sponsoring organizations.

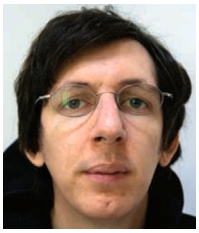
References

[1] P. Cudrano, B. Gallazzi, M. Frosi, S. Mentasti, M. Matteucci, Clothoid-based lane-level high-definition maps: Unifying sensing and control models, IEEE Veh. Technol. Mag. 17 (4) (2022) 47–56.
[2] H.G. Seif, X. Hu, Autonomous driving in the iCity—HD maps as a key challenge of the automotive industry, Engineering 2 (2) (2016) 159–162.
[3] C. Schwarz, Z. Wang, The role of digital twins in connected and automated vehicles, IEEE Intell. Transp. Syst. Mag. 14 (6) (2022) 41–51.
[4] L. Bao, Q. Wang, Y. Jiang, Review of Digital twin for intelligent transportation system, in: Proc. of International Conference on Information Control, Electrical Engineering and Rail Transit, 2021, pp. 309–315.
[5] B. Yu, C. Chen, J. Tang, S. Liu, J.-L. Gaudiot, Autonomous vehicles digital twin: A practical paradigm for autonomous driving system development, Computer 55 (9) (2022) 26–34.
[6] G. Chen, G. Esch, P. Wonka, P. Müller, E. Zhang, Interactive procedural street modeling, ACM Trans. Graph. 27 (3) (2008) 1–10.
[7] F. Poggenhans, J.-H. Pauls, J. Janosovits, S. Orf, M. Naumann, F. Kuhnt, M. Mayr, Lanelet2: A high-definition map framework for the future of automated driving, in: Proc. of International Conference on Intelligent Transportation Systems, 2018, pp. 1672–1679.
[8] S. Kato, E. Takeuchi, Y. Ishiguro, Y. Ninomiya, K. Takeda, T. Hamada, An open approach to autonomous vehicles, IEEE Micro 35 (6) (2015) 60–68.
[9] M. Bellusci, P. Cudrano, S. Mentasti, R.E.F. Cortelazzo, M. Matteucci, Semantic bird’s-eye view road line mapping, in: Proc. of International Conference on Intelligent Transportation Systems, 2023, In print.
[10] L. Peng, Z. Chen, Z. Fu, P. Liang, E. Cheng, BEVSegFormer: Bird’s eye view semantic segmentation from arbitrary camera rigs, in: Proc. of IEEE/CVF Winter Conference on Applications of Computer Vision, 2023, pp. 5935–5943.
[11] K. Dinakaran, A.S. Sagayaraj, S.K. Kabilesh, T. Mani, A. Anandkumar, G. Chandrasekaran, Advanced lane detection technique for structural highway based on computer vision algorithm, Mater. Today Proc. 45 (2021) 2073–2081.
[12] J. Wang, T. Mei, B. Kong, H. Wei, An approach of lane detection based on inverse perspective mapping, in: Proc. of International Conference on Intelligent Transportation Systems, 2014, pp. 35–38.
[13] A.A. Assidiq, O.O. Khalifa, M.R. Islam, S. Khan, Real time lane detection for autonomous vehicles, in: Proc. of International Conference on Computer and Communication Engineering, 2008, pp. 82–88.
[14] O. Ronneberger, P. Fischer, T. Brox, U-net: Convolutional networks for biomedical image segmentation, in: Proc. of International Conference on Medical Image Computing and Computer-Assisted Intervention, 2015, pp. 234–241.
[15] D. Wu, M.-W. Liao, W.-T. Zhang, X.-G. Wang, X. Bai, W.-Q. Cheng, W.-Y. Liu, YOLOP: You only look once for panoptic driving perception, Mach. Intell. Res. 19 (6) (2022) 550–562.
[16] C. Han, Q. Zhao, S. Zhang, Y. Chen, Z. Zhang, J. Yuan, YOLOPv2: Better, faster, stronger for panoptic driving perception, 2022, arXiv preprint arXiv:2208.11434.
[17] D. Vu, B. Ngo, H. Phan, HybridNets: End-to-end perception network, 2022, arXiv preprint arXiv:2203.09035.
[18] W. Jang, J. Hyun, J. An, M. Cho, E. Kim, A lane-level road marking map using a monocular camera, IEEE/CAA J. Autom. Sin. 9 (1) (2022) 187–204.
[19] S. Ren, K. He, R. Girshick, J. Sun, Faster R-CNN: Towards real-time object detection with region proposal networks, Adv. Neural Inf. Process. Syst. 28 (2015).
[20] J. Tian, J. Yuan, H. Liu, Road marking detection based on mask R-CNN instance segmentation model, in: Proc. of International Conference on Computer Vision, Image and Deep Learning, 2020, pp. 246–249.
[21] Y. Zhou, Y. Takeda, M. Tomizuka, W. Zhan, Automatic construction of lane-level HD maps for urban scenes, in: Proc. of IEEE/RSJ International Conference on Intelligent Robots and Systems, 2021, pp. 6649–6656.
[22] Q. Li, Y. Wang, Y. Wang, H. Zhao, HDMapNet: An online HD map construction and evaluation framework, in: Proc. of International Conference on Robotics and Automation, 2022, pp. 4628–4634.
[23] N. Homayounfar, W.-C. Ma, J. Liang, X. Wu, J. Fan, R. Urtasun, DAGMapper: Learning to map by discovering lane topology, in: Proc. of IEEE/CVF International Conference on Computer Vision, 2019, pp. 2911–2920.
[24] H. Liu, C. Lin, B. Gong, D. Wu, Automatic lane-level intersection map generation using low-channel roadside LiDAR, IEEE/CAA J. Autom. Sin. 10 (5) (2023) 1209–1222.
[25] C. Ye, H. Zhao, L. Ma, H. Jiang, H. Li, R. Wang, M.A. Chapman, J.M. Junior, J. Li, Robust lane extraction from MLS point clouds towards HD maps especially in curve road, IEEE Trans. Intell. Transp. Syst. 23 (2) (2022) 1505–1518.
[26] S. Liu, J.-L. Gaudiot, Autonomous vehicles lite self-driving technologies should start small, go slow, IEEE Spectr. 57 (3) (2020) 36–49.
[27] A. Zang, R. Xu, Z. Li, D. Doria, Lane boundary extraction from satellite imagery, in: Proc. of ACM SIGSPATIAL Workshop on High-Precision Maps and Intelligent Applications for Autonomous Vehicles, 2017, pp. 1–8.
[28] S. He, H. Balakrishnan, Lane-level street map extraction from aerial imagery, in: Proc. of IEEE/CVF Winter Conference on Applications of Computer Vision, 2022, pp. 2080–2089.
[29] W. Jang, J. An, S. Lee, M. Cho, M. Sun, E. Kim, Road lane semantic segmentation for high definition map, in: Proc. of IEEE Intelligent Vehicles Symposium, 2018, pp. 1001–1006.
[30] G. Máttyus, S. Wang, S. Fidler, R. Urtasun, HD maps: Fine-grained road segmentation by parsing ground and aerial images, in: Proc. of IEEE Conference on Computer Vision and Pattern Recognition, 2016, pp. 3611–3619.
[31] J. Zhou, Y. Guo, Y. Bian, Y. Huang, B. Li, Lane information extraction for high definition maps using crowdsourced data, IEEE Trans. Intell. Transp. Syst. 24 (7) (2023) 7780–7790.
[32] D. Pannen, M. Liebner, W. Burgard, Lane marking learning based on crowdsourced data, in: Proc. of IEEE/RSJ International Conference on Intelligent Robots and Systems, 2019, pp. 7040–7046.
[33] M. Liebner, D. Jain, J. Schauseil, D. Pannen, A. Hackelöer, Crowdsourced HD map patches based on road model inference and graph-based SLAM, in: Proc. of IEEE Intelligent Vehicles Symposium, 2019, pp. 1211–1218.
[34] K. Kim, S. Cho, W. Chung, HD map update for autonomous driving with crowdsourced data, IEEE Robot. Autom. Lett. 6 (2) (2021) 1895–1901.
[35] K.-W. Chiang, H.-Y. Pai, J.-C. Zeng, M.-L. Tsai, N. El-Sheimy, Automated modeling of road networks for high-definition maps in OpenDRIVE format using mobile mapping measurements, Geomatics 2 (2) (2022) 221–235.
[36] M. Tan, Q. Le, EfficientNet: Rethinking model scaling for convolutional neural networks, in: Proc. of International Conference on Machine Learning, 2019, pp. 6105–6114.
[37] M. Tan, R. Pang, Q.V. Le, EfficientDet: Scalable and efficient object detection, in: Proc. of IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2020, pp. 10781–10790.
[38] S. Elfwing, E. Uchibe, K. Doya, Sigmoid-weighted linear units for neural network function approximation in reinforcement learning, Neural Netw. 107 (2018) 3–11.
[39] T.-Y. Lin, P. Goyal, R. Girshick, K. He, P. Dollár, Focal loss for dense object detection, in: Proc. of IEEE International Conference on Computer Vision, 2017, pp. 2980–2988.
[40] S.S.M. Salehi, D. Erdogmus, A. Gholipour, Tversky loss function for image segmentation using 3D fully convolutional deep networks, in: Proc. of International Workshop on Machine Learning in Medical Imaging, 2017, pp. 379–387.
[41] A. Tversky, Features of similarity, Psychol. Rev. 84 (4) (1977) 327.
[42] H.A. Mallot, H.H. Bülthoff, J.J. Little, S. Bohrer, Inverse perspective mapping simplifies optical flow computation and obstacle detection, Biol. Cybernet. 64 (3) (1991) 177–185.
[43] Y. Chen, Z. Xiang, W. Du, Improving lane detection with adaptive homography prediction, Vis. Comput. (2022) 1–15.
[44] M. Miksch, B. Yang, K. Zimmermann, Homography-based extrinsic self-calibration for cameras in automotive applications, in: Workshop on Intelligent Transportation, 2010, pp. 17–22.

[45] M. Bellusci, M. Matteucci, Advances in real-time online vehicle camera calibration via road line markings parallelism enforcement, in: Proc. of IEEE Intelligent Vehicles Symposium, 2022, pp. 1511–1516.
[46] P. Cudrano, S. Mentasti, M. Matteucci, M. Bersani, S. Arrigoni, F. Cheli, Advances in centerline estimation for autonomous lateral control, in: Proc. of IEEE Intelligent Vehicles Symposium, 2020, pp. 1415–1422.
[47] M.A. Fischler, R.C. Bolles, Random sample consensus: A paradigm for model fitting with applications to image analysis and automated cartography, Commun. ACM 24 (6) (1981) 381–395.
[48] F. Yu, H. Chen, X. Wang, W. Xian, Y. Chen, F. Liu, V. Madhavan, T. Darrell, BDD100k: A diverse driving dataset for heterogeneous multitask learning, in: Proc. of IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2020, pp. 2636–2645.
[49] D.P. Kingma, J. Ba, Adam: A method for stochastic optimization, in: Proc. of International Conference on Learning Representations, 2015, pp. 1–15.
[50] M. Bersani, S. Mentasti, P. Cudrano, M. Vignati, M. Matteucci, F. Cheli, Robust vehicle pose estimation from vision and INS fusion, in: Proc. of International Conference on Intelligent Transportation Systems, 2020, pp. 1–6.

Matteo Bellusci is a Ph.D. student in computer science and engineering at the Dipartimento di Elettronica Informazione e Bioingegneria, Politecnico di Milano, Milano, Italy. He received his M.Sc. degree in computer science and engineering from the Politecnico di Milano. His main research focuses on artificial intelligence, 3D reconstruction, and robotics.

Paolo Cudrano is a Ph.D. student in computer science and engineering at the Dipartimento di Elettronica Informazione e Bioingegneria, Politecnico di Milano, Milano, Italy. He received his M.Sc. degree in computer science and engineering from the Politecnico di Milano and his M.Sc. degree in computer science from McMaster University in 2019. His research focuses on perception systems for robotics and autonomous vehicles.



Simone Mentasti is a researcher at the Dipartimento di Elettronica Informazione e Bioingegneria, Politecnico di Milano, Milano, Italy. He received his M.S. degree in computer science from the Università Statale di Milano, Italy, in 2017 and his Ph.D. degree in computer engineering from the Politecnico di Milano in 2022. His research interests focus on robotics, perception, sensor fusion, sensor calibration, and deep learning for autonomous driving cars.

Riccardo Erminio Filippo Cortelazzo received his M.Sc. degree in computer science and engineering at Politecnico di Milano, Milano, Italy, in 2023 and his B.Sc. degree in computer engineering at Politecnico di Milano in 2020. His interests include deep learning and computer vision.

Matteo Matteucci is a full professor at the Dipartimento di Elettronica Informazione e Bioingegneria, Politecnico di Milano, Milano, Italy. He received his Ph.D. degree in computer engineering and automation from the Politecnico di Milano. His main research topics are pattern recognition, machine learning, machine perception, robotics, computer vision, and signal processing.