https://doi.org/10.3311/PPtr.37323

Periodica Polytechnica Transportation Engineering, 53(4), pp. 389–405, 2025

Navigation Framework from a Monocular Camera for Autonomous Mobile Robots

Udink Aulia[1,2], Iskandar Hasanuddin[2], Muhammad Dirhamsyah[2], Nasaruddin Nasaruddin[3*]

[1] Doctoral Program, School of Engineering, Universitas Syiah Kuala, Jl. Tgk. Chik Pante Kulu No. 5, Kopelma Darussalam, 23111 Banda Aceh, Indonesia
[2] Department of Mechanical and Industrial Engineering, Faculty of Engineering, Universitas Syiah Kuala, Jl. Syech Abdurrauf No. 7, Kopelma Darussalam, 23111 Banda Aceh, Indonesia
[3] Department of Electrical and Computer Engineering, Faculty of Engineering, Universitas Syiah Kuala, Jl. Syech Abdurrauf No. 7, Kopelma Darussalam, 23111 Banda Aceh, Indonesia
* Corresponding author, e-mail: nasaruddin@usk.ac.id

Received: 12 May 2024, Accepted: 30 July 2025, Published online: 12 August 2025

Abstract

Ground detection plays a critical role in the perception systems of autonomous mobile robots. The ground is typically characterized as a smooth, drivable surface with uniform texture, making it distinguishable from its surroundings. However, it may exhibit imperfections such as shadows and varying light conditions. This paper presents a framework for the detection of vanishing points, drivable road regions, intersections, and obstacles in the context of autonomous mobile robot navigation. The proposed framework leverages Google's DeepLab v3+ for semantic segmentation of the road, employs the Hough line transform to identify vanishing points and drivable areas, utilizes an intersection analyzer to locate intersections linked to drivable areas, and incorporates a free obstacle detector to identify various objects within drivable regions. Our objective is to simplify the perception of ground-related information in recent methodologies and offer a solution to comprehend and harness the capabilities of these frameworks. The primary significance of this study lies in evaluating the performance of these networks in real-world deployment scenarios. The evaluation results demonstrate that our proposed framework achieves high accuracy across diverse and challenging situations. Consequently, the developed framework holds promise for integration into autonomous mobile robots (AMRs).

Keywords

autonomous mobile robot (AMR), vanishing point, drivable area, segmentation, Hough line transform

1 Introduction

Autonomous mobile robots (AMRs) utilize road images to detect drivable areas, identify lanes, recognize objects close to the robot, and gather other essential information. This information forms the basis for making informed decisions during autonomous navigation. Drivable area detection is a critical technique that involves segmenting road images. Modern methods often consider road detection as pixel-wise classifications. However, this method faces challenges related to computational cost and speed. Road detection has attracted a lot of research attention and has long been regarded as a crucial component of autonomous mobile robots. Significant advancements have been made in road detection thus far. However, in some cases, such as when there are pedestrians or abrupt vehicle maneuvers that render the identified road impassable, depending only on road detection for automated driving decisions may not be sufficient. When the human driver operates a car, they comprehend scenarios by distinguishing between obstacles and non-obstacles rather than solely identifying the road. In image segmentation, object boundaries typically manifest in areas with depth discontinuities. Consequently, it is essential to integrate image segmentation with depth discontinuity for a comprehensive solution.

The foundation of AMRs is a reliable road area detection technique. Many strategies have evolved to address this issue during the previous few decades. Based on the sensors used to collect the data, these techniques may be divided into four groups: monocular cameras, stereo vision, laser sensors, and multi-sensor fusion. In vision-based navigation for AMRs, a robust vanishing point

Cite this article as: Aulia, U., Hasanuddin, I., Dirhamsyah, M., Nasaruddin, N. (2025) "Navigation Framework from a Monocular Camera for Autonomous Mobile Robots", Periodica Polytechnica Transportation Engineering, 53(4), pp. 389–405. https://doi.org/10.3311/PPtr.37323

(*VP*) estimate is crucial. Multiple obstacles arise in line segment refining and the elimination of bogus vanishing point candidates due to the different nature of traffic conditions and the multitude of disruptions resulting from clutter and occlusion. Thus, vision-based autonomous cars continue to have significant difficulty when it comes to *VP* estimates, particularly in complicated environments.

The topic is how a vision-based AMR for a monocular camera extracts useful textures and characterizes their properties to support higher-level inferences such as environmental perception. Perceiving depth in the human visual system does not require any further information. Points with geometric connections are more suited for approximating the relative depth of different features. It makes sense to believe that these geometric *VP*s can be approximated using organized principles (Wei and Wang, 2018). However, complicated environments frequently force current approaches to contend with reduced precision and higher computing overheads because of the noise introduced by several extracted line segments. As a result, the effective estimate of *VP*s is becoming more and more important for autonomous cars' visual navigation. Our method for measuring *VP*s from a monocular camera in this work eliminates the requirement for internal camera calibration or prior training (Shen et al., 2022).

Unlike current methods, which directly use simple line segments, we suggest an approach based on refined lines, which have organized configurations and orientations. This method improves the estimation process's robustness. Easy geometric deductions enable the suggested algorithm to adjust to variations in image brightness and color, making it useful and effective for scene recognition in autonomous mobile robots. The effectiveness of the suggested algorithm in efficiently estimating virtual principals in intricate environments is demonstrated by the experimental results, which are ascertained by comparing the percentage of pixel errors with the ground truth. Moreover, the algorithm accomplishes this with minimal resource usage and maximum efficiency, highlighting its noteworthy potential for broad future applications in scene understanding and visual navigation (Shen et al., 2022).

To tackle this difficult issue, a novel approach to robot navigation is presented in this paper. It uses the boundary pixel-following algorithm to identify sidewalk regions at intersections, the drivable line region method for navigation, and the Hough transformation on a segmentation map of a road image to find vanishing points. As was previously mentioned, the feature data of the image must be extracted in order to identify the *VP*. This includes straight lines and other directional information. On the other hand, excessive real-time processing may result from the extraction of linear components and feature information from every pixel in the input image, lengthening calculation times. Furthermore, current feature voting techniques are unsuitable for real-time applications such as driving scenarios, even though they can accurately detect *VP*s. We present a method that chooses candidate *VP* regions to determine the *VP*. The fast linear components are then obtained by applying the Hough transform to determine the Hough peak point. The next step is determining the possible *VP* by finding the intersection of at least two straight lines. The segmentation map's "*VP*" intersection points are those that are closest to the black-marked areas or road representations. The following is a summary of this paper's contributions:

- We utilize the Hough transformation of the edges and segmentation map to pinpoint the location of the vanishing point for the drivable region.
- We introduce the concept of a drivable line region to establish the path for autonomous mobile robots on the road when traveling between different locations.
- We present a novel algorithm for detecting sidewalk regions, which are crucial for extracting intersections on the main roads.

2 Related work

There are three primary processing phases in the majority of geometrical feature-based *VP* detection methods:

1. Step 1: Different attributes in driving images are used to extract line segments.
2. Step 2: The selection of line segments that tend to converge at *VP* points.
3. Step 3: A voting procedure is utilized to choose the *VP* at the intersections of the chosen line segments.

Traditional *VP* estimation algorithms were presented. Based on three key criteria, a method for looking into *VP*s in stair regions was presented (Khaliluzzaman, 2021). A population-based algorithm approach was presented to improve the effectiveness of the metaheuristic *VP* search (López-Martinez and Cuevas, 2020). A method for improving *VP* detection in landscape images was presented. It included combining various image representations to improve the edge extraction process (García-Faura et al., 2019). An approach utilizing a harmony search (HS) algorithm for *VP* estimation was also

presented (Moon et al., 2018). However, these approaches exhibit limited capability in obtaining *VP*s in cluttered and occluded traffic scenes.

There is a technique for using a monocular camera to comprehend curved alley scenes without any prior training in which coplanar structures and associated *VP*s are estimated, and angle projections are extracted and allocated to clusters. Wang and Wei (2021a) offered a technique for processing the visual signal that incorporates a wide range of noise and color disturbances. The suggested method can handle the signal, even with significant color and noise disruption, by using geometric principles. In Wei et al.'s (2020) article, to achieve temporal consistency and increase the robustness of the *VP* calculation, a temporal average is used. In Nagy and Costa's (2021) article, a technique that, allows non-Manhattan obstructions to be understood from a single image in an indoor environment, without the need for internal camera calibration or previous training (Wang and Wei, 2020a). A quick and easy way to estimate absolute orientation with just one vanishing point is presented in Guo et al.'s (2022) article.

*VP*s have been the subject of recent studies on scene comprehension and visual navigation. An algorithm that combined convolutional neural networks (CNN) and heatmap regression was created to identify unstructured road virtual principals (Liu et al., 2021). Another one used *VP* prediction to investigate 3D interactions between objects and spatial arrangement (Wang and Wei, 2021b). Another method, though in relatively clean environments, inferred wheelchair ramp scenes based on multiple spatial rectangles derived from domain *VP*s (Wang and Wei, 2020b). However, in complicated road environments with obstacles and clutter, these algorithms have trouble estimating *VP*s.

Remarkable models have been developed for traffic situations. One method used line-soft-voting with maximum weight to estimate *VP*s based on road boundary region estimation (Wang et al., 2018). A detection method was proposed for surveillance applications to identify *VP*s in complicated environments with changing exterior variables, such as railroad stations and subterranean spaces (Tarrit et al., 2018). However, these techniques are vulnerable to robust *VP* estimation failures because complex traffic environments contain unpredictable disturbances. Therefore, the creation of a refined line-based algorithm for *VP* estimation from monocular cameras without the need for prior training is urgently needed. In addition, a refined line-based technique for *VP* estimation, which is more cost-effective and efficient, offers greater potential for scene interpretation and visual navigation in autonomous cars.

Scene understanding is a complex process that uses a network of sensors to perceive, analyze, and interpret scenes that are observed. This complex task includes a range of difficulties, from basic image classification to more complex tasks like object detection and Semantic Segmentation (SS). While image classification involves giving an input image a global label, it is not very useful in autonomous driving scenarios where environmental element localization is important. The second task, object detection, uses object recognition and classification to provide a more in-depth description of the scene. The third task, on the other hand, is the hardest since it requires accurately identifying each pixel in an input image in order to provide semantic information. Complex machine-learning architectures are required for this task because of their capacity to produce precise semantic descriptions. It is an essential goal for our main focus in this work, which is a scene understanding pre-processor.

The majority of semantic segmentation techniques were first created to accept input from a single RGB camera (Rizzoli et al., 2022). Road detection within a driving context is a prominent area of study, significantly enhancing the capabilities of on-vehicle intelligent systems for a deeper understanding of the environment and, consequently, improved traffic safety and efficiency. Vision systems play a pivotal role in perception tasks for intelligent vehicles, offering versatile and cost-effective solutions that provide rich information, including color, shape, and depth, all while conserving power. Road detection, among the prominent research areas in vision-based systems, not only supplies essential information for path planning but also enhances obstacle detection accuracy and road profile estimation (Hamandi et al., 2018).

A single image point known as the vanishing point (*VP*) is where a set of parallel lines in three dimensions are projected into two dimensions and converge. The number of sets of parallel lines in 3D object space that project onto an image determines how many *VP*s there are in that image. One of the most fundamental problems in computer vision is vanishing point detection, and creating a quick and precise algorithm for this task will have a significant impact on a wide range of applications. These include wireframe parsing, object detection, camera calibration, 3D reconstruction, and autonomous driving.

Three essential processes are usually involved in traditional *VP* detection methods: *VP* regression, line clustering, and line detection. Line segments are predicted to cluster into three different groups that correspond to the three orthogonal *VP*s in scenarios where the Manhattan world

assumption is true. The geometric assumption that line segments within the same group should converge at a shared point, which represents the vanishing point, has frequently been the foundation of previous methods. This *VP* can be found at the junction of a road lane that runs vertically parallel to the sky or at the intersection of a road lane and a road border structure. To put it simply, image segmentation is essential to finding the intersection of the road area and the sky area, which is the point at which the *VP* is identified. To effectively detect the *VP*, which is a focus point of feature information inside an image, one must locate the area with the highest concentration of feature information.

Semantic image segmentation represents a quintessential computer vision challenge involving the meticulous assignment of distinct categories to individual pixels within an image based on the underlying objects of interest. The availability of large training datasets and the powerful processing capacity of high-performance GPUs have allowed deep learning techniques—especially supervised techniques like deep convolutional neural networks, or DCNNs—to excel in a variety of challenging computer vision tasks during the past few years. These tasks include but are not limited to semantic segmentation, object detection, and image classification.

Deep learning approaches offer a significant advantage in that they can automatically learn high-level feature representations in an end-to-end manner. This allows for far more discriminative features than can be produced with more conventional methods. Building on the success of deep learning methods for classifying images, scientists have been carefully investigating how these networks might be used to handle annotations at the pixel level. As a result, they have presented numerous well-known deep-learning architectures designed especially for semantic segmentation (Zhang et al., 2020).

A fundamental step in computer vision, image segmentation—also referred to as superpixeling—involves breaking a digital image up into several pieces. The main purpose of segmentation is to alter or simplify an image's representation to make it simpler to comprehend and examine. This method is usually used to locate objects and define edges, such as curves and lines, in pictures. More precisely, image segmentation is the process of assigning a label to each pixel in a picture so that similar properties are shared by pixels that have the same label. A super pixel represents a group of pixels that exhibit common features; they are proximate to each other and possess similar colors. Obtaining a comprehensive image segmentation based on super pixels often results in an over-segmentation devoid of any inherent semantic meaning. However, this segmentation approach proves valuable as it reduces the complexity of the image while enabling efficient processing, given that superpixels significantly outnumber individual pixels. Moreover, the shape of the superpixel can be a valuable asset in various applications.

DeepLab series (Chen et al., 2015; 2017; 2018a; 2018b) represents a popular and effective family of semantic segmentation models based on DCNNs. With the introduction of atrous convolution in DeepLab v1 (Chen et al., 2015), the receptive field was effectively expanded without adding more parameters to the network. It combined a fully connected Conditional Random Field (CRF) with the final DCNN layer to improve object boundary localization. Using the PASCAL VOC 2012 dataset, DeepLab v1 was able to attain an accuracy of 71.6% mIOU, thanks to these sophisticated techniques. To enhance DeepLab v1 to provide robust object segmentation across multiple scales, DeepLab v2 (Chen et al., 2018a) introduced atrous spatial pyramid pooling (ASPP). To capture multi-scale features and allow for the segmentation of objects and context at multiple scales, ASPP makes use of multiple parallel atrous convolutional layers with different dilation rates. On the PASCAL VOC 2012 dataset, DeepLab v2 achieved an impressive 79.9% mIOU accuracy by combining atrous convolution, ASPP, and fully connected CRF.

These enhancements were further expanded upon in DeepLab v3 (Chen et al., 2017), which included batch normalization, enhanced ASPP, and a better method for encoding multi-scale context. The accuracy it achieved on the PASCAL VOC 2012 dataset was 85.7% mIOU. Three 3×3 atrous convolutions with different dilation rates, a 1×1 convolution, and image-level feature concatenation were all included in the improved ASPP. Each parallel convolutional layer was followed by batch normalization, and DeepLab v3 did away with the fully connected CRF. ASPP and Encoder-Decoder structures were introduced in DeepLab v3+ (Chen et al., 2018b). The Decoder module improved the segmentation results at the pixel level. Additionally, it investigated the Xception model and used the Decoder module and ASPP's depthwise separable convolution. DeepLab v3+ showed remarkable performance on the Cityscapes dataset and the PASCAL VOC 2012 test set, obtaining 82.1% and 89.0% accuracy, respectively. Although DeepLab v3+ has shown excellent results in semantic image segmentation, there are still some issues with it. The segmentation map comprises a collection of pixels that signify object classes

through color representation within the pixels. In the context of a robot navigation system, it is essential to identify roads, as well as moving and stationary objects, on the segmentation map. Extracting map segmentation for each class consumes time since it necessitates pixel-to-pixel comparisons within the vicinity. The pixel-following technique involves tracking contour pixels according to a predetermined pattern and then storing their coordinates in memory in accordance with the tracing order. This algorithm searches for surrounding black pixels based on directional orders such as left, front-left, front, front-right, right, rear-right, and rear, working its way clockwise from the current pixel. Pixel-following techniques, which rely on a chain code, follow simple rules for contour pixel tracing. These techniques include the simple boundary follower (SBF), modified SBF (MSBF), improved SBF (ISBF), Moore-neighbor tracing (MNT), radial sweep algorithm (RSA), and Theo Pavlidis' algorithm (TPA). These techniques only maintain pixel coordinates; therefore, they require frame-sized memory to track the contour, and they often result in incorrect images when the contour image is extended.

Traditional pixel-following techniques have certain drawbacks. Firstly, some algorithms, like SBF and MSBF, execute unnecessary movement operations on white pixels. Second, not all algorithms can define contour well enough when contour pixels are present. This means that they are unable to describe an object's features or establish an object's connectivity. Moreover, like SBF, MSBF is also unable to distinguish between inner and outer corners as well as straight-line pixels. Regarding ISBF, it can recognize pixels with inner-outer, front-inner, and front-straight lines. Still, it cannot identify some outer-corner pixels, left-straight lines, and left-inner corners. In a similar vein, TPA is unable to distinguish between different kinds of contour pixels, and neither MNT nor RSA can identify inner corners. Finally, the data size of the traced contour must be taken into account. Pixel-following techniques yield larger data sizes than RD code techniques because they maintain every pixel point.

3 Methods

Many spatial corners in a complicated traffic environment have spatial lines that are frequently projected and detected, some of which meet certain geometric constraints. These lines of space project into 2D angle projections in a variety of configurations, and they are important for determining the direction and associated vanishing points (*VP*s) in this kind of environment. Our suggested approach uses regions of interest (ROIs) to find potential points of line intersections to solve the *VP* detection problem. We choose a line in the middle of the input image to be processed further from the lines that the Hough transform detects.

To identify neighboring pixels of the road region, we compare these candidate points with a segmentation map that has a 32 × 32 pixel filter. The pixel with the lowest y-value position will be chosen as the vanishing point (*VP*) if a road region pixel is found. Fig. 1 provides an overview of the suggested framework, and this section goes into detail on each step. The study's suggested framework includes vanishing point selection, drivable line region, intersection analyzer, and environment segmentation, as shown in Fig. 1. The goal of environment segmentation is to categorize each pixel in an image according to the segmentation map's object of interest. The vanishing point selection then uses the Hough line to extract the position of the potential vanishing point in the segmentation map. The autonomous mobile robot can predict the distance to other objects on the road by organizing the available region and identifying sidewalk and drivable line regions with the intersection analyzer.

3.1 Environment segmentation

The present study used a pre-trained DeepLab v3+ - DCNNs-based semantic segmentation model for object segmentation across 19 classes due to its high effectiveness. As illustrated in Table 1, the segmentation result is an RGB map code for each class, wherein each pixel is assigned colors corresponding to the class name.

3.2 Line detection

In this step, we use a Canny detector to identify edges, and then we use the Hough transform to detect straight lines based on edge information. The Hough transform method is a well-known, traditional, and efficient line segment detection algorithm. However, it needs to employ the appropriate threshold settings, such as the number of edge points that make up a line segment and the length and spacing of line segments, to avoid a lot of false-positive and false-negative errors. Because the Hough transform is specifically made for straight-line detection, it can be used to estimate candidate vanishing points by identifying intersections between line components that contain edge information. By computing ρ ($x\cos(\theta) + y\sin(\theta)$), which represents a parallel straight line that passes through any pixel at an angle (θ) concerning the x-axis, line detection is achieved. The ρ and θ values are then used to create a Hough transform matrix,

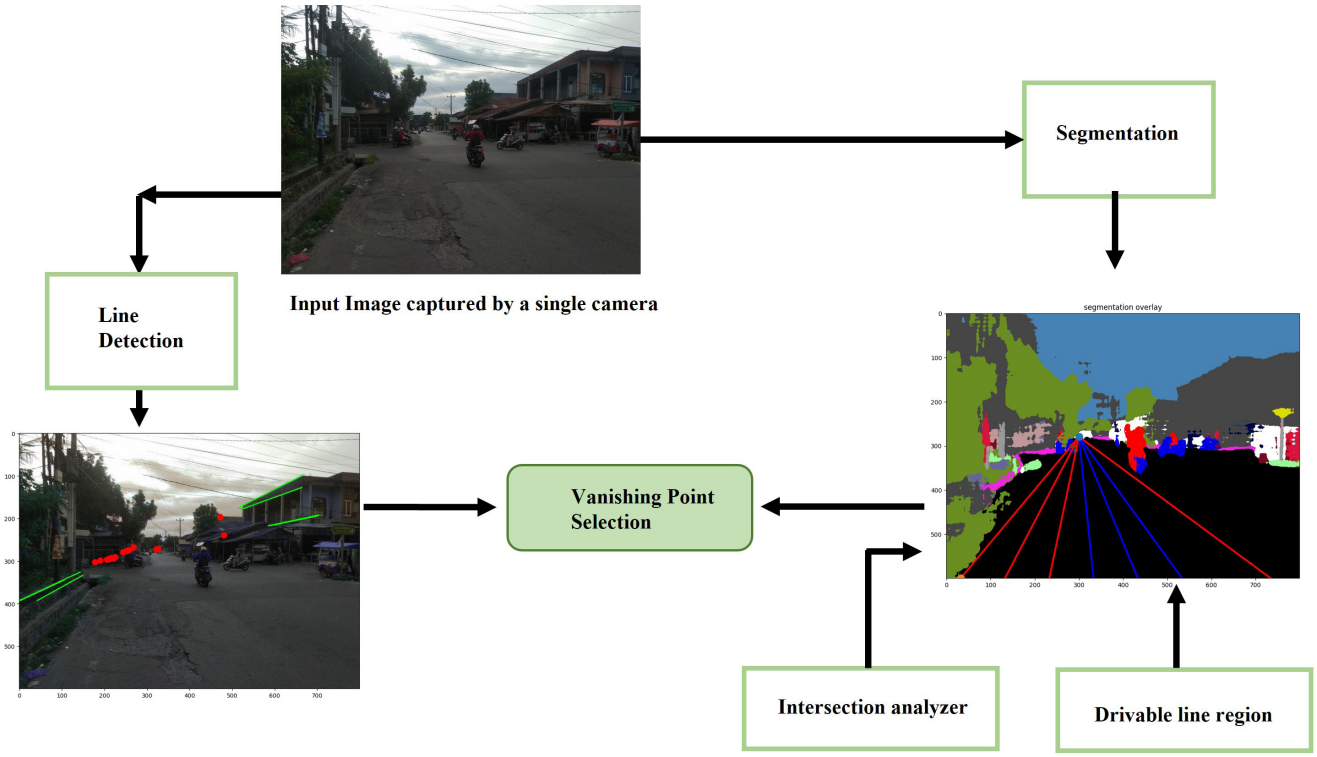

Fig. 1 The proposed framework for navigation of mobile robots

Table 1 DeepLab v3+ Class Name and RGB code (Choi et al., 2022)

ID	class	RGB code
0	road	0, 0, 0
1	sidewalk	244, 35, 232
2	building	70, 70, 70
3	wall	102, 102, 156
4	fence	190, 153, 153
5	pole	153, 153, 153
6	traffic light	250, 170, 30
7	traffic sign	220, 220, 0
8	vegetation	107, 142, 35
9	terrain	152, 251, 152
10	sky	70, 130, 180
11	person	220, 20, 60
12	rider	255, 0, 0
13	car	255, 255, 255
14	truck	0, 0, 70
15	bus	0, 60, 100
16	train	0, 80, 100
17	motorcycle	0, 0, 230
18	bicycle	119, 11, 32

represented as $\boldsymbol{H}$, and the point in $\boldsymbol{H}$ that has the largest peak is regarded as a possible straight line. We looked into ways to shorten the range of pixels used in the computation, restrict the range of line angles (θ), and identify the characteristic information that best characterizes straight-line features in order to speed up the creation of the Hough transform matrix. Generally speaking, edges provide the most accurate depiction of straight lines in an image, and thresholding is the most used method to binarize these edges for the Hough transform.

3.3 Removing invalid line segments

Upon the extraction of straight-line segments from the road image, it becomes necessary to eliminate certain invalid line segments that should not contribute to the voting process. The criteria for removing such segments are as follows:

1. Voting is not allowed for line segments that are vertical or nearly vertical or that are horizontal or nearly horizontal (i.e., the angle between the line segment and the vertical or horizontal direction is less than or equal to 5°). The vertical edges of things like poles, tree trunks, or buildings are usually represented as vertical or nearly vertical line segments.
2. The line segments that are horizontal or almost horizontal and do not coincide with the vanishing point are also not allowed to vote.

3.4 *VP* selection

This stage involved using a mix of line detection and picture segmentation to calculate the vanishing point

(*VP*). But as was indicated in the Introduction, mistakes might occur if *VP* selection relied just on one processing step. Thus, a multistage processing strategy was used to improve the accuracy of *VP* identification.

The following process was used to choose a *VP* that satisfied certain requirements, such as being unique from other pixels and being located at the junction of two or more lines: The *VP* was defined as the pixel with the lowest *y*-axis value that was closest to the edge of a segmented road zone. By taking into account the pixel situated in the center of the position point of the detected junction pixels within a 32 × 32 pixel window applied to the road-segmented region, this technique guaranteed the identification of the optimal *VP*. When two or more windows had black pixels, they were compared, and the window with the highest *y*-coordinate value was chosen.

Then, inside the selected window, a left-to-right and top-to-bottom scanning procedure was carried out to find the *x*-coordinate corresponding to the highest *y*-value. The vanishing points resulting from the vote outcomes of various *VP* candidates, which began at junction locations on the left and right sides of the road, are represented by these *x* and *y* coordinates, as illustrated in Fig. 2.

3.5 Performance metric

To measure the estimation error between the detected vanishing point (*VP*) and the manually determined ground truth, which was established by human perceptual assessment, we have employed the normalized Euclidean distance approach. We compared the outcomes of our algorithms with the ground truth vanishing points, which were manually noted by a department participant, in order to evaluate the inaccuracy in the vanishing-point estimate.

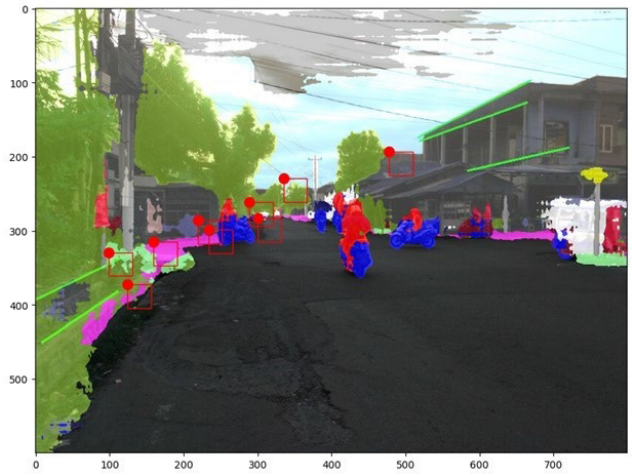

Fig. 2 Vanishing point selection

The notion of road vanishing points was briefly explained to the participants before they were asked to note the position of the vanishing point in each image.

Traditionally, vanishing-point estimation quality has been evaluated in terms of a single metric (e.g., the Euclidean distance in pixels) or by calculating the pixel-wise difference between the estimated vanishing point and the ground truth separately in both rows and columns. However, the inaccuracy in pixels is not a good indicator of how well different approaches actually perform when they are used at different picture resolutions. To overcome this, we have opted to employ a quantitative evaluation based on the normalized Euclidean distance. This metric uses the picture resolution's diagonal size to normalize the Euclidean distance, as follows:

$$\text{NormDist} = \frac{\|P - P_0\|}{\text{Diag Image}}. \tag{1}$$

The estimated vanishing-point location is denoted by $P(x_p, y_p)$ in Eq. (1), whereas the ground-truth vanishing-point location's center is $P_0(x_0, y_0)$. The term "Diag Image" describes how big the image's diagonal is. A number near 1 would suggest an inaccurately assessed vanishing point position when using this suggested measure. A number close to 0, on the other hand, indicates that the anticipated vanishing point position and the ground truth location are quite close.

3.6 Drivable line region

A drivable line region is an area that a robot can traverse. This area is divided into two regions: the obstacle free region, marked with a blue line, and the obstacle region, marked with a red line. It is determined by three key points: the vanishing point and the left and right boundary points of the road area. The left and right boundary points are identified by scanning vertically at the right, left, and bottom of the image to find the road color. The red and blue lines represent drivable paths, determined by drawing a line from the vanishing point to the left and right boundary points of the road, as shown in Fig. 3.

The line is marked in red if it intersects with a moving object such as a motorcycle, person, or car and in blue if no intersection occurs. The area marked with the blue line is considered traversable by the robot.

3.7 Intersection analyzer

An intersection refers to an area where two or more roads meet or cross. These intersections can range from simple

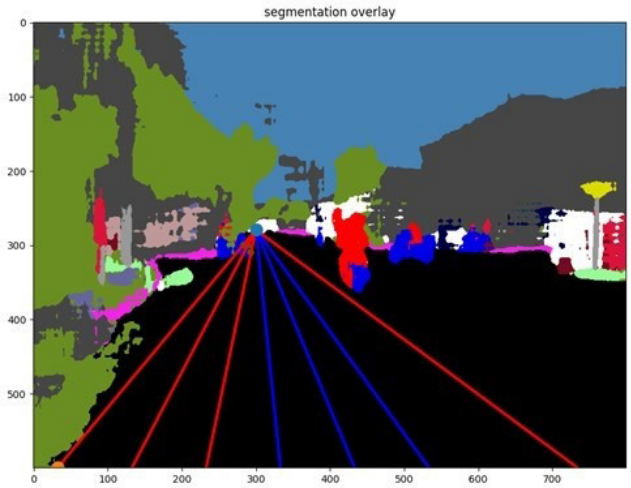

Fig. 3 Drivable line region

junctions formed by the confluence of two roads to more complex ones involving the intersection of several roads.

For the input image, DeepLab v3+ creates a class-indexed score map where each pixel is given an index that corresponds to its expected class. Within this segmentation map, intersections consistently feature a sidewalk region aligned horizontally with the road area. The sidewalk region's pixels are distinguished by their RGB values of 244, 35, and 232 (magenta) for the inner region, while other objects belong to the outer region.

The goal of this algorithm is to delineate the boundary of the sidewalk region and its adjacent areas, such as roads, vegetation, buildings, motorcycles, and more. To achieve this, we use 8-connectivity kernels with neighbors in the east, southeast, south, southwest, west, and north directions. The eight directions are denoted by the letters A, B, C, D, E, F, G, H, and I, as shown in Fig. 4. These kernels help identify pixels of the same color that are neighboring pixels of different colors.

In Fig. 5, the top view of the intersection location is shown, with the viewing direction from right to left as depicted in Fig. 6. The sidewalk detection process to obtain

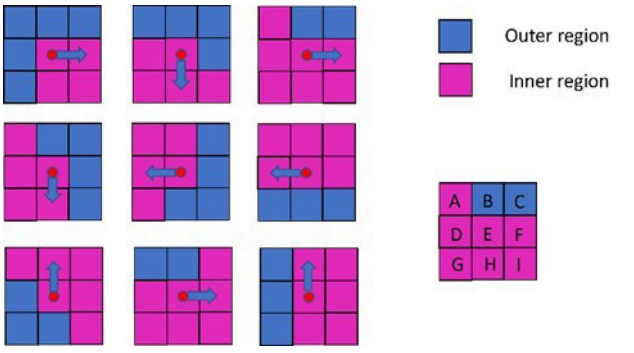

Fig. 4 Cases of Boundary Pixel following algorithm

Fig. 5 Bird's eye view of intersection

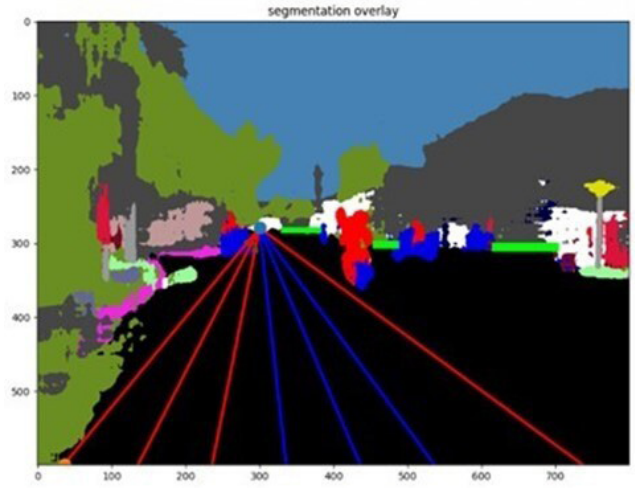

Fig. 6 Sidewalk detection

the intersection before and after the process is shown in Fig. 3 and Fig. 6. In Fig. 7, the sidewalk color is magenta, and the detected region will be changed to light green. The color change after the detection process aims to speed up the detection process so that the detected region is not searched again.

As shown in Fig. 7, a raster scan is performed along the *y*-axis (horizontal direction) to identify the starting point of each sidewalk region. This scan begins at the image's origin and continues until it locates a pixel with a smaller *y*-value. Subsequently, we utilize the kernel for a boundary pixel-following algorithm, tracing the sidewalk region's boundary in a clockwise manner until it reaches the starting point, effectively closing the loop.

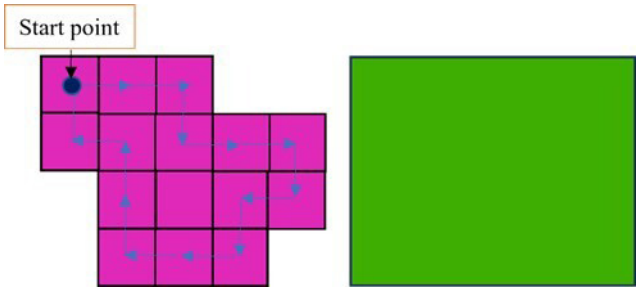

Fig. 7 Finding sidewalk region and light green window

All the coordinates of the traversed points are recorded in an array, facilitating the determination of the region's dimensions. Each detected region will be highlighted with a light green rectangle, facilitating the swift identification of additional sidewalk regions. The raster scan can then proceed to explore other areas until no more sidewalk regions are detected. More details about the algorithm for finding sidewalk regions are described in Algorithm 1.

Fig. 8 shows the segmentation model of the environment around the robot. The process of forming the drivable line starts with determining the left and right boundary points of the road. This step is done by finding 4 points located on the left and right sides and 2 points from the bottom.

The points on the left side are denoted as L, and on the right side as R. On the bottom side, $B1$ approaches the point $x = 0$, and $B2$ approaches $x = M$, where M and N are the number of pixels in the x and y directions.

There are four combinations to determine the boundaries, as shown in Fig. 9. The reference point at $x = 0$, $y = 0$ is located at the top left corner, with the x and y axes pointing positively. The maximum image coordinate in the x direction is M, while in the y direction, it is N.

Next, we need to determine the DV point starting from point $B1$ or L, which is 100 pixels away. The DV line in the image passes through the VP point towards a set of DV points located at the bottom side. Then, for each formed DV line, a line equation model is created. Furthermore, object detection is performed on each DV line for items such as motorcycles and cars, each of which has a different color. The x-value for each line starts from the VP coordinate to the DV coordinate, where object color detection is executed. If an object is found, the current DV line changes from the default blue color to red. The blue color indicates that the line, which serves as the path for the robot, is safe to traverse. In contrast, the red color indicates a need to maintain distance or overtake, as the detected object has not changed position. The drivable line algorithm can be seen in Algorithm 2.

Algorithm 1 Finding a sidewalk region

Require:

M, image size in x axes

N, image size in y axes

Ensure: $x, y \in M, N$

search point with magenta color from $x = 0$, $y = 0$ (upper left corner)

for each pixel $x \in M$ **do**

  for each pixel $y \in N$ **do**

    If color = magenta **then** break

    end if

  end for

end for

Set start point $P(x, y)$#inner region, magenta color

Save start point $T(x, y)$

Set status direction status A, status B, status C, status D, status E, status F, status G, status H, status I = other

Set init coordinate
$x_A = x_E - 1, y_A = y_E - 1; x_B = x_E, y_B = y_E - 1$
$x_C = x_E + 1, y_C = y_E - 1; x_D = x_E - 1, y_D = y_E$
$x_F = x_E + 1, y_F = y_E; x_G = x_E - 1, y_G = y_E + 1$
$x_H = x_E, y_H = y_E + 1; x_I = x_E + 1, y_I = y_E + 1$

DO

$(r, g, b) = P(x_A, y_A)$ #set status from A to I

if color = magenta **then status A = True**

#read status 8 pixels and make next move in x, y

If (status A = other) **and** (status B = other) **and** (status C = other) **and** (status D = other) **and** (status E = magenta) **and** (status F = magenta) **and** (status G = other) **and** (status H = magenta) **and** (status I = magenta) **then**
  $x_E = x_E + 1$
  $y_E = y_E$ # increment x by 1 for x_E to the next position

#set position for 8 pixels around center position E: x_E, y_E

#Set coordinate A, B, C, D, E, F, G, H, I referenced to E
$x_A = x_E - 1, y_A = y_E - 1; x_B = x_E, y_B = y_E - 1$
$x_C = x_E + 1, y_C = y_E - 1; x_D = x_E - 1, y_D = y_E$
$x_F = x_E + 1, y_F = y_E; x_G = x_E - 1, y_G = y_E + 1$
$x_H = x_E, y_H = y_E + 1; x_I = x_E + 1, y_I = y_E + 1$

#set status for next position from A to I for inner dan outer region

$(r, g, b) = \text{pix}[x_A, y_A]$
if matching_algo(r, g, b, 244, 35, 232):

status A = True

if matching_algo(r, g, b, 255, 255, 255) **or** matching_algo(r, g, b, 0, 0, 0) or matching_algo(r, g, b, 0, 0, 230) or matching_algo(r, g, b, 255, 0, 0):#If the color is different from the sidewalk(road) status A = False

WHILE $T(x, y) <> C(x_E, y_E)$ # loop while start point not equal current position

4 Experimental result

In this study, we estimate vanishing points (VPs) from refined lines using a geometric technique. Unlike deep-learning-based methods, our method does not require extra high-performance GPUs. Comparing the calculated VPs to the ground truth allowed for the evaluation of pixel errors, as shown in Table 1. The outcomes validate the effectiveness of the VP estimate based on improved lines, which forms the basis of our suggested approach. Moreover, the input captures from lines extracted using different edge line recognition techniques were included in our tests. In this work, straight-line components were utilized to estimate

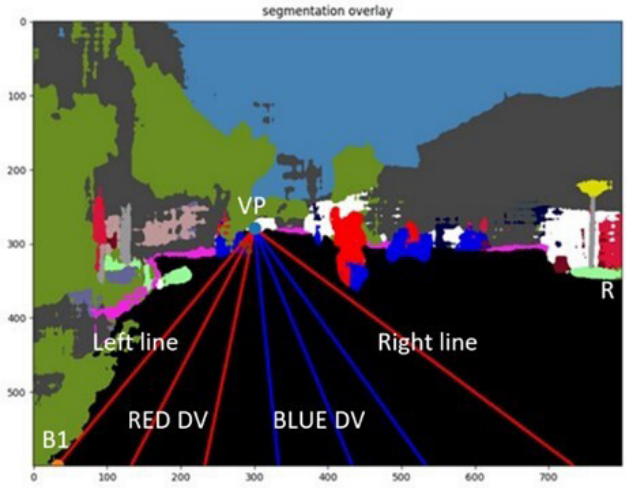

Fig. 8 Drivable line generation

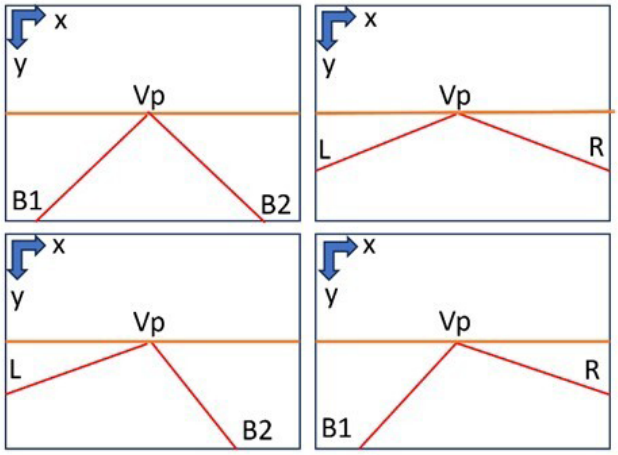

Fig. 9 The road boundary model

candidate vanishing points after edges were removed from the input pictures. As exemplified in Fig. 5, the depicted urban intersection scenario (Intersection 4) features buildings on both sides of the road, devoid of marking lines typically used for *VP* determination.

In the sky, there are power cables that, when detected, may appear nearly horizontal. However, vertical lines originating from buildings, electricity, or telephone poles are not included in the voting process and are considered outliers. Next, the edge detection process using the Canny operator, as illustrated in Fig. 10, provides points for further processing via the Hough transform to identify lines.

To get the lower and upper threshold values in the Hough transformation process, we use the lower threshold values of 50, 60, and 70 and the upper threshold value of 100. At the lower threshold value of 50 and the upper threshold value of 100 in Fig. 11, we can see that there are 7 lines. Green is detected with a distribution of cut points in red, which is more on the left. 2 red points are very close to the vanishing point area (the area directly adjacent to the road).

At the lower threshold value of 60 and the upper threshold value of 100 in Fig. 12, it can be seen that five green lines are detected with the distribution of red points on the left. There are 2 red dots above the vanishing point area.

At the lower threshold value of 70 and the upper threshold value of 100 in Fig. 13, it can be seen that there are 5 green lines detected with an even distribution of intersection points; the red ones are more on the left. There is 1 red dot in the middle of the road area.

By looking at the distribution of red points on the road area in the three combinations of threshold values, the lowest threshold value of 70 and the upper threshold value of 100 were chosen in this experiment. By setting the threshold value for line detection using the Hough method, we exclude most of the lines considered outliers from this process. Then, we limit the angle of the lines to between −40 and −10 degrees, which are included in the line refinement process, as shown in Fig. 14.

This ensures that only the lines oriented toward the vanishing point are included in the voting process, as depicted in Figs. 11–13. The results of the Hough method, as shown in Fig. 13, reveal four valid lines: two originating from the foundation of the house on the left side of the road and two from the top of the building on the right side of the road. The intersection of these four lines generates multiple intersection points, which serve as candidates for vanishing points. The distribution of the lines, spreading more to the left of the road, is due to intersecting lines originating from different heights, where the line on the right side of the road is higher than the one on the left side of the road. Fig. 15 shows a combination of the original image and the segmentation image where 1 red dot is in the black road segmentation area.

Fig. 16 illustrates the process of determining the *VP* from several points of intersection of lines using a 32 × 32 kernel to generate vanishing points.

Within each kernel, we search for a point located in a road-type region, denoted by the black region. Points within the road region are highlighted in yellow. The voting results for the *VP* candidates are shown as dark blue dots. These points are determined by identifying the smallest *y*-value among the coordinates included in the yellow 32 × 32 kernel, as depicted in Fig. 17.

To present the experimental results, we have compiled a selection of images from the Banda Aceh City Dataset in this section. All images have been uniformly resized

Algorithm 2 Finding a drivable line

Require: ***Vp*(*x*, *y*)** #vanishing point
M, image size in *x* axes
N, image size in *y* axes
Ensure: $x, y \in M, N$
search point *B*1(*x*, *y*) with black color from $x = 0$ to $x = M$, $y = N$ (bottom side) #left to the right
$y = N$
for each pixel $x = 0$ to *M* **do**
if color = black **then** break
end if
end for
return *B*1(*x*, *y*)
search point *B*2(*x*, *y*) with black color, from $x = M$ to $x = 0$, $y = N$ (bottom side) # right to the left
$y = N$
for each pixel $x = M$ to 0 **do**
if color = black **then** break
end if
end for
return *B*2(*x*, *y*)
search point *L*(*x*, *y*) with black color from $y = 0$ to $y = N$, $x = 0$ (left side) from Top to Bottom
$x = 0$
for each pixel $y = 0$ to *N* **do**
if color = black **then** break
end if
end for
return *L*(*x*, *y*)
search point *R*(*x*, *y*) with black color from $y = 0$ to $y = N$, $x = M$ (right side) from Top to Bottom
$x = M$
for each pixel $y = 0$ to *N* **do**
if color = black **then** break
end if
end for
return *R*(*x*, *y*)
# draw left line and right line from *Vp* to *B*1, *Vp* to *B*2
Line(*Vp*(*x*, *y*)) − (*B*1(*x*, *y*)), Line(*Vp*(*x*, *y*)) − (*B*2(*x*, *y*))
# draw drivable line from left line to right line *Vp* to *DV*
for each line $x = 0$ to *N* step 100 **do**
Line(*Vp*(*x*, *y*)) − (*DV*(*x*, *M*)),
$Y = m \times x + c$, # find *c* and *m*
#search object in each drivable line by using point, if found point in the object then color = red otherwise blue
for each point in line $x = Vp(x)$ to $DV(x)$ **do**
$y = m \times x + c$
If (color = motorcycle_color) or (color = car_color) or
(color = person_color) then line color = red
end if
end for

to dimensions of 800 × 600 pixels, ensuring no distortion. Fig. 17 shows cases and a set of example images illustrating the detection of vanishing points and road regions using our proposed method.

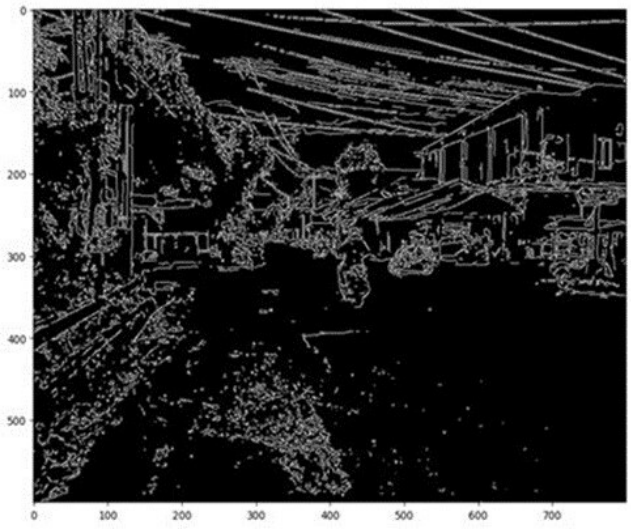

Fig. 10 Result of Canny edge detector process

Fig. 11 Result of line detection by Hough transform and intersection point at the lower threshold value 50 and upper threshold value 100

Fig. 12 Result of line detection by Hough transform and intersection point at the lower threshold value 60 and upper threshold value 100

Fig. 13 Result of line detection by Hough transform and intersection point at the lower threshold value 70 and upper threshold value 100

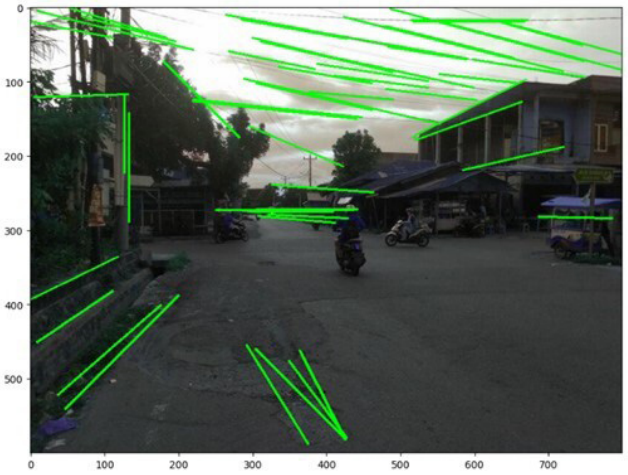

Fig. 14 The line refinement process before filtering

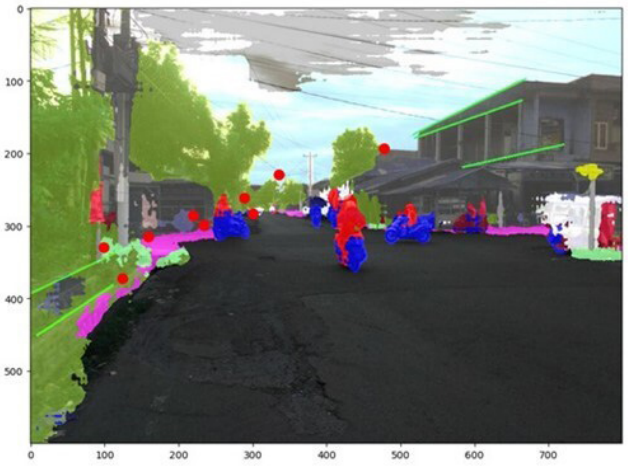

Fig. 15 Overlay of Intersection point on segmentation map

These examples include images captured in challenging structured road environments. The intersection shown by the green lines in each picture indicates the location of the road's identified vanishing point.

In comparison, the ground-truth vanishing point position is shown by the green circle's center. Our technique

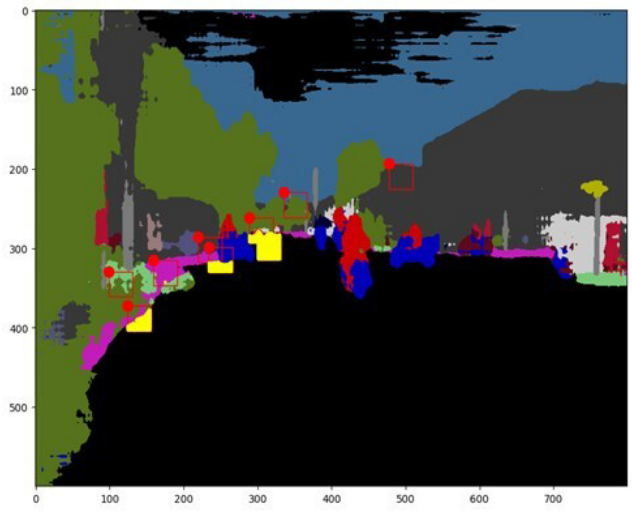

Fig. 16 Identification of candidate *VP* from the intersection point

Fig. 17 The estimated *VP* by our method

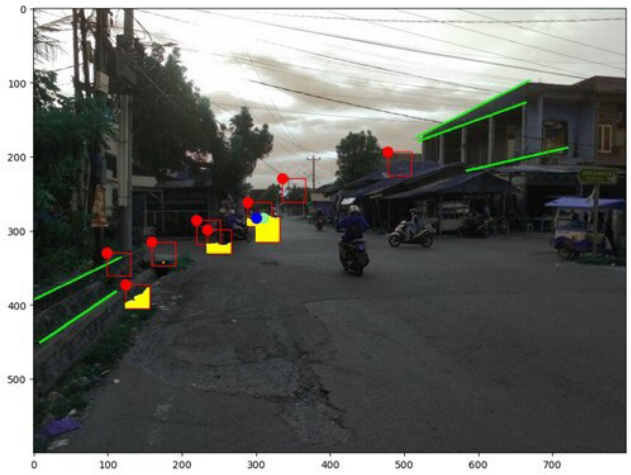

Fig. 18 The estimated *VP* and the ground truth for the straight road by our method

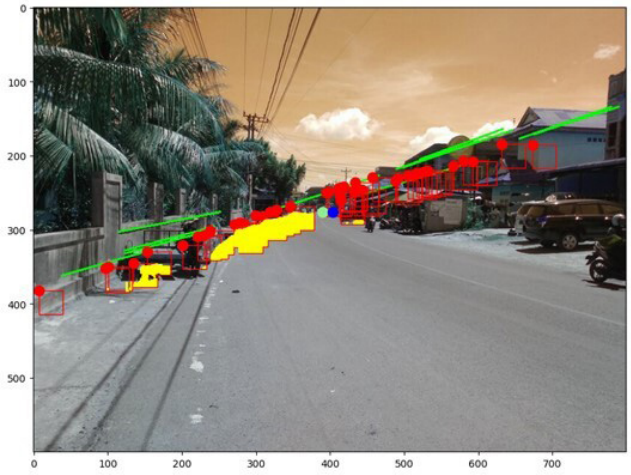

Fig. 19 The estimated *VP* and the ground truth for the curved road by our method

is more flexible for structured urban roadways and can extract valid lines even from structures like reservoirs and buildings that converge near the road's ground-truth vanishing point.

Figs. 18 and 19 show how well our technique performs in structured road environments for vanishing point recognition and road region detection. The suggested approach obtains an inaccuracy of 0.008 (or 8.0 pixels in distance) for straight roads in Fig. 18 and 0.014 (or 14.0 pixels in distance) for curved roads in Fig. 19. The graphic illustrates that the vanishing point positions in structured road environments are accurately estimated by the proposed method. There are several cases where this method does not work because the conversion process from image to segmentation does not produce regional classes according to the original image. The road area has only been slightly converted to black and the rest to gray, as shown in Fig. 20. The original image is 400 × 225 pixels, and the DeepLab v3 model is not able to convert images with small pixel sizes.

In Fig. 21, most of the road area has been converted to black. The original image is 320 × 214 pixels, and the DeepLab v3 model cannot convert small-size images. For images with a size of 800 × 600 pixels, the conversion results show very accurate results.

To see the effect of the number of lines and the presence of buildings on the left and right of the road, we can compare them in Figs. 22 and 23. In Fig. 22, there are 24 lines detected, and there are no buildings on the left. In Fig. 23, there are 9 lines detected, and there are buildings on the left and right of the way.

From the comparison of the two images, it is found that the number of lines does not determine the accuracy of vanishing point detection. The position of the building to the left and right of the road greatly determines the accuracy of vanishing point detection. This is because not all lines can be detected by the Hough method after the Canny process, which is determined by the threshold value.

Fig. 20 Failure cases of our method in the segmentation of the road with image 400 × 225 pixels

Fig. 21 Failure cases of our method in the segmentation of the road with image 320 × 214 pixels

Fig. 22 Vanishing point detection results with 24 lines detected

5 Discussion

In our proposed method, the vanishing point is determined using an 800 × 600 image. This is necessary so that the segmentation results on the road section do not cause errors in the segmentation process. Comparison with other methods that use a smaller pixel size of 320 × 240 can be done by looking at Eq. (1) with the diagonal variables of different images.

Experimental findings showed that the suggested technique can accurately estimate *VP* by calculating the proportion of pixel errors from the ground truth. The process of determining the threshold value for the parameters of the Hough transform is necessary so that the resulting intersection point approaches the black road area resulting from the segmentation. In the line refinement process, it is used to remove lines that are not needed to produce a valid line for the process of determining the vanishing point, there are still unwanted lines, although the detected vanishing points are still accurate, as shown in Fig. 24. Fig. 25 shows *VP* detection in environmental

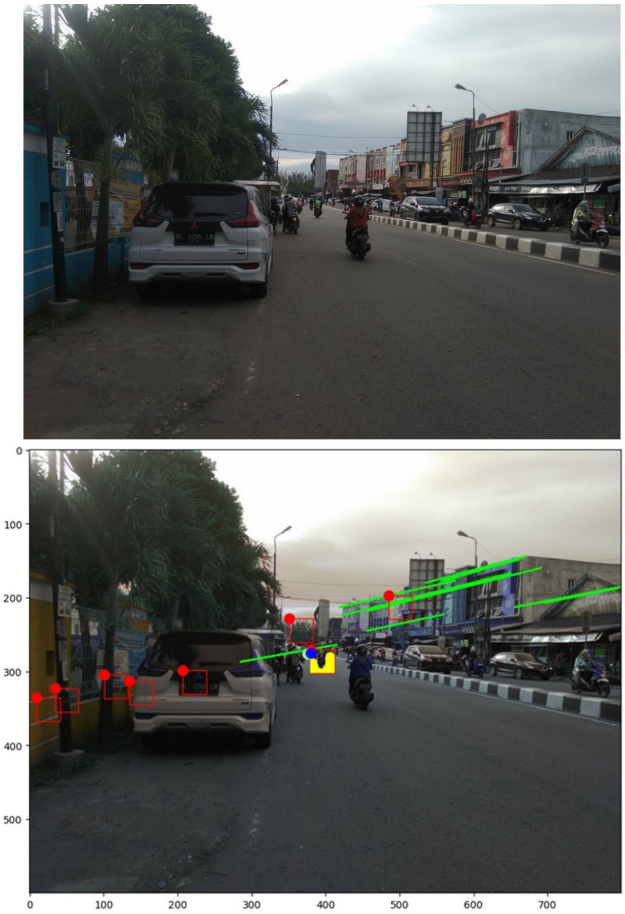

Fig. 23 Vanishing point detection results with 9 lines detected

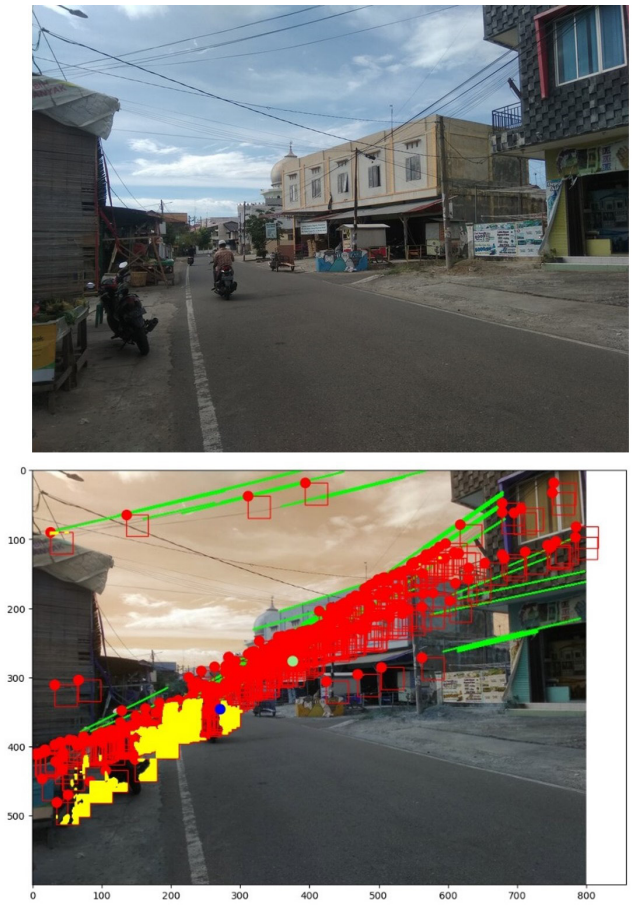

Fig. 24 Vanishing point detection results with many false lines detected

conditions, there are trees on the right side of the road and the left, there is an arrangement of wooden planks connecting the building above the water with the main road. The detection accuracy between ground truth and prediction results is 20 pixels, which is still good for an image size of 800 × 600 pixels.

We evaluated the vanishing point detection outcomes in urban areas for a thorough qualitative study and quantitative comparison. For comparison, we evaluated three new texture-based approaches in addition to the traditional edge-based approach. In particular, we contrasted our suggested strategy with the methods put forth by Kong et al. (2009), Moghadam et al. (2012) and Rasmussen (2004).

Table 2 presents the accuracy comparison of vanishing point identification in the urban road category. The value of the result is the average of the normalized Euclidean distance error for each technique. To calculate the overall mean error, the mean value of the normalized Euclidean distance error across all images for each technique is employed. A lower mean error suggests that the algorithm is more accurate overall.

6 Conclusion

For vanishing point (*VP*) identification in urban road images, a unique line space voting system has been developed in this article. Using the Hough technique, which is based on the Canny operator and computes gradient and texture orientations, we first identify line segments in road images. The points of these lines then cast their collective votes for the vanishing point in the associated segmentation map, together with the points of their nearby lines. It then determines the estimated vanishing point by looking at the place with the greatest voting score. The main advantage of our suggested strategy over other approaches is its high accuracy, simplicity, and ease of implementation. As such, our approach is suitable to fulfill the needs of autonomous mobile robots for route detection. However, it is important to remember that in some road images, the Hough approach could have trouble identifying legitimate line segments, particularly those with very smooth surfaces or intricate texture directions. There are several cases where this method does not work because the conversion process from an image with a small size of pixels to

Fig. 25 Vanishing point detection with tree objects on the side of the road

Table 2 Accuracy comparison for *VP* detection

Method	Accuracy
Rasmussen (2004)	0.0515
Kong et al. (2009)	0.0332
Edge-based method	0.0538
Moghadam et al. (2012)	0.0318
Proposed work	0.0320

segmentation does not produce regional classes according to the original image. We plan to address this limitation by exploring the integration of depth maps in our future work.

Acknowledgments

This work was supported by the Ministry of Education, Culture, Research, and Technology of The Republic of Indonesia Grant No. 575/UN11.2.1/PT.01.03/DPRM/2023.

References

Chen, L.-C., Papandreou, G., Kokkinos, I., Murphy, K., Yuille, A. L. (2015) "Semantic image segmentation with deep convolutional nets and fully connected CRFs", presented at International Conference on Learning Representations (ICLR 2015), San Diego, CA, USA, May, 9. https://doi.org/10.48550/arXiv.1412.7062

Chen, L.-C., Papandreou, G., Schroff, F., Adam, H. (2017) "Rethinking Atrous Convolution for Semantic Image Segmentation", [preprint] arXiv, arXiv:1706.05587v3, 05 December 2017. https://doi.org/10.48550/arXiv.1706.05587

Chen, L.-C., Papandreou, G., Kokkinos, I., Murphy, K., Yuille, A. L. (2018a) "DeepLab: Semantic Image Segmentation with Deep Convolutional Nets, Atrous Convolution, and Fully Connected CRFs", IEEE Transactions on Pattern Analysis and Machine Intelligence, 40(4), pp. 834–848. https://doi.org/10.1109/TPAMI.2017.2699184

Chen, L.-C., Zhu, Y., Papandreou, G., Schroff, F., Adam, H. (2018b) "Encoder-Decoder with Atrous Separable Convolution for Semantic Image Segmentation", In: Computer Vision – ECCV 2018, Munich, Germany, pp. 833–851. ISBN 978-3-030-01233-5 https://doi.org/10.1007/978-3-030-01234-2_49

Choi, K., Lim, W., Chang, B., Jeong, J., Kim, I., Park, C.-R., Ko, D. W. (2022) "An automatic approach for tree species detection and profile estimation of urban street trees using deep learning and Google street view images", ISPRS Journal of Photogrammetry and Remote Sensing, 190, pp. 165–180. https://doi.org/10.1016/j.isprsjprs.2022.06.004

García-Faura, Á., Fernández-Martínez, F., Kleinlein, R., San-Segundo, R., Díaz-de-María, F. (2019) "A multi-threshold approach and a realistic error measure for vanishing point detection in natural landscapes", Engineering Applications of Artificial Intelligence, 85, pp. 713–726. https://doi.org/10.1016/j.engappai.2019.08.001

Guo, K., Ye, H., Gu, J., Tian, Y. (2022) "A Fast and Simple Method for Absolute Orientation Estimation Using a Single Vanishing Point", Applied Sciences, 12(16), 8295. https://doi.org/10.3390/app12168295

Hamandi, M., Asmar, D., Shammas, E. (2018) "Ground segmentation and free space estimation in off-road terrain", Pattern Recognition Letters, 108, pp. 1–7. https://doi.org/10.1016/j.patrec.2018.02.019

Khaliluzzaman, M. (2021) "Analytical justification of vanishing point problem in the case of stairways recognition", Journal of King Saud University - Computer and Information Sciences, 33(2), pp. 161–182. https://doi.org/10.1016/j.jksuci.2019.01.004

Kong, H., Audibert, J.-Y., Ponce, J. (2009) "Vanishing point detection for road detection", In: 2009 IEEE Conference on Computer Vision and Pattern Recognition, Miami, FL, USA, pp. 96–103. ISBN 978-1-4244-3992-8 https://doi.org/10.1109/CVPRW.2009.5206787

Liu, Y.-B., Zeng, M., Meng, Q.-H. (2021) "Unstructured Road Vanishing Point Detection Using Convolutional Neural Networks and Heatmap Regression", IEEE Transactions on Instrumentation and Measurement, 70,5002708. https://doi.org/10.1109/TIM.2020.3019863

López-Martinez, A., Cuevas, F. J. (2020) "Vanishing point detection using the teaching learning-based optimisation algorithm", IET Image Processing, 14(11), pp. 2487–2494. https://doi.org/10.1049/iet-ipr.2019.0516

Moghadam, P., Starzyk, J. A., Wijesoma, W. S. (2012) "Fast vanishing-point detection in unstructured environments", IEEE Transactions on Image Processing, 21(1), pp. 425–430. https://doi.org/10.1109/TIP.2011.2162422

Moon, Y. Y., Geem, Z. W., Han, G.-T. (2018) "Vanishing point detection for self-driving car using harmony search algorithm", Swarm and Evolutionary Computation, 41, pp. 111–119. https://doi.org/10.1016/j.swevo.2018.02.007

Nagy, T. K., Costa, E. C. M. (2021) "Development of a lane keeping steering control by using camera vanishing point strategy", Multidimensional Systems and Signal Processing, 32(2), pp. 845–861. https://doi.org/10.1007/s11045-021-00763-2

Rasmussen, C. (2004) "Grouping dominant orientations for ill-structured road following" In: Proceedings of the 2004 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'04), Washington, DC, USA, vol. 1, pp. 1–1. ISBN 0-7695-2158-4 https://doi.org/10.1109/cvpr.2004.1315069

Rizzoli, G., Barbato, F., Zanuttigh, P. (2022) "Multimodal Semantic Segmentation in Autonomous Driving: A Review of Current Approaches and Future Perspectives", Technologies, 10(4), 90. https://doi.org/10.3390/technologies10040090

Shen, S., Wang, S., Wang, L., Wei, H. (2022) "A Refined-Line-Based Method to Estimate Vanishing Points for Vision-Based Autonomous Vehicles", Vehicles, 4(2), pp. 314–325. https://doi.org/10.3390/vehicles4020019

Tarrit, K., Molleda, J., Atkinson, G. A., Smith, M. L., Wright, G. C., Gaal, P. (2018) "Vanishing point detection for visual surveillance systems in railway platform environments", Computers in Industry, 98, pp. 153–164. https://doi.org/10.1016/j.compind.2018.03.005

Wang, E., Sun, A., Li, Y., Hou, X., Zhu, Y. (2018) "Fast vanishing point detection method based on road border region estimation", IET Image Processing, 12(3), pp. 361–373. https://doi.org/10.1049/iet-ipr.2017.0030

Wang, L., Wei, H. (2020a) "Avoiding Non-Manhattan Obstacles Based on Projection of Spatial Corners in Indoor Environment", IEEE/CAA Journal of Automatica Sinica, 7(4), pp. 1190–1200. https://doi.org/10.1109/JAS.2020.1003117

Wang, L., Wei, H. (2020b) "Understanding of wheelchair ramp scenes for disabled people with visual impairments", Engineering Applications of Artificial Intelligence, 90, 103569. https://doi.org/10.1016/j.engappai.2020.103569

Wang, L., Wei, H. (2021a) "Curved Alleyway Understanding Based on Monocular Vision in Street Scenes", IEEE Transactions on Intelligent Transportation Systems, 23(7), pp. 8544–8563. https://doi.org/10.1109/TITS.2021.3083572

Wang, L., Wei, H. (2021b) "Indoor scene understanding based on manhattan and non-manhattan projection of spatial right-angles", Journal of Visual Communication and Image Representation, 80, 103307. https://doi.org/10.1016/j.jvcir.2021.103307

Wei, H., Wang, L. (2018) "Understanding of indoor scenes based on projection of spatial rectangles", Pattern Recognition, 81, pp. 497–514. https://doi.org/10.1016/j.patcog.2018.04.017

Wei, H., Wang, L., Wang, S., Jiang, Y., Li, J. (2020) "A Signal-Processing Neural Model Based on Biological Retina", Electronics, 9(1), 35. https://doi.org/10.3390/electronics9010035

Zhang, S., Ma, Z., Zhang, G., Lei, T., Zhang, R., Cui Y. (2020) "Semantic image segmentation with deep convolutional neural networks and quick shift", Symmetry, 12(3), 427. https://doi.org/10.3390/sym12030427