Vanishing Point Constrained Lane Detection With a Stereo Camera

Yingna Su, Yigong Zhang, Tao Lu, Jian Yang, and Hui Kong

***Abstract*—In this paper, we propose a robust vanishing-point constrained lane detection method with a stereo-rig. This method can achieve promising detection performance for both straight and curved lanes without assuming any parametric lane model. First, we propose an accurate and efficient road vanishing point detection scheme based on the v-disparity and visual odometry techniques, where the v-disparity map can significantly reduce the searching space for vanishing point, and the visual odometry can benefit the vanishing point detection of both straight and curved roads. Next, we formulate the lane detection problem as a graph-search procedure, where a vanishing-point constrained Dijkstra shortest-path lane model is proposed to obtain a minimum-cost map. The two lane borders can be detected by finding two optimal paths which originate from the vanishing point to two cost-map derived terminal points, respectively. The proposed method has been tested on the KITTI and the Oxford RobotCar data sets and it works accurately and robustly on a variety of road scenes.**

***Index Terms*—Stereo vision, lane detection, vanishing point, shortest-path search.**

I. Introduction

DURING the past two decades, many algorithms for lane detection purpose have been proposed. Most of them focus on straight or curved lane detection on highways based on a linear or nonlinear parametric lane model. When the lane borders are blocked partially by vehicles or when there are many tessellated shadow patterns, these parametric methods tend to fail to find the correct lane borders. For such cases, we need to find a robust way to locate the lane regions in spite of occlusions or shadows.

This paper proposes a nonparametric lane model to meet such a demand. We formulate the lane detection problem as a graph-search procedure. To use the graph-search scheme to find two paths in a general image, we usually need four terminal points with two per path. However, we only need three terminal points in general for road images. The first one is the joint point of two parallel lane borders at the very far end of road in a perspective image (i.e., the vanishing point of the road), and the other two are the joint points of each lane border with the image bottom.

To find the two optimal paths between the first terminal point and the other two, a vanishing-point constrained Dijkstra model is proposed to create a minimum-cost map with image being treated as a graph and each pixel as a node of the graph. The source node of the graph is set to the road vanishing point, which can be accurately and efficiently detected based on the stereo v-disparity and visual odometry techniques. The stereo v-disparity map can significantly reduce the searching space of vanishing point, and the visual odometry can benefit the vanishing point detection of both straight and curved roads. Once finding the vanishing point (the first terminal point), we can also find the other two terminal points in the last row of the image based on the minimum-cost map. Then the two lane borders can be detected by finding two optimal paths which originate from the vanishing point to the other two terminal points, respectively. Our method can obtain accurate results when applied to both straight and curved roads. Figure 1 shows the framework of our lane detection method. The major contributions of this paper are as follows: (1) we propose a nonparametric lane-border model in a graph-search framework. (2) To realize it, we propose an accurate and efficient vanishing point detection scheme with the stereo v-disparity and visual odometry techniques based on the texture-voting strategy [3]. (3) We propose a new method to create a Dijkstra minimum-cost map based on both gradient and disparity information.

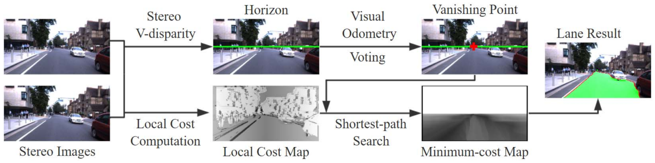


Fig. 1. Flowchart of our lane detection method.

The rest of the paper is organized as follows: Section II presents related works. Section III introduces the details of the proposed method. Experimental results are given in Section IV. We draw the conclusions in the last section. Note that this paper is best viewed in color.

II. Related Work

Since the early 1990s, researches on lane detection have been more and more popular. Most of the existing approaches can be divided into two categories [4]: feature- [5]–[7] and model-based [8]–[10]. Feature-based approaches detect lane borders by using low-level features like lane edges. While model-based approaches represent the lane as a curve model which can be determined by several critical geometric parameters. They both have their pros and cons.

Feature-based approaches are the earliest and most extended methods for lane detection. Bertozzi and Broggi [11] performed an adaptive binarization and filtered the image through a few iterations of a geodesic morphologic dilation on the Inverse Perspective Mapped (IPM) image. Yoo *et al.* [12] proposed a gradient-enhancing conversion for illumination-robust lane detection. In general, feature-based methods mostly require clear and distinct lane borders, and suffer from weak lane borders, occlusions and noise. Compared with feature-based approaches, model-based approaches are less sensitive to noise and weak lane features, and are effective to partial occlusions. Wang *et al.* [13] used the B-Snake to fit lane borders.

Manuscript received December 14, 2016; revised May 27, 2017 and July 25, 2017; accepted September 2, 2017. This work was supported in part by the Jiangsu Province Natural Science Foundation under Grant BK20151491 and in part by the Natural Science Foundation of China under Grant 61672287. The Associate Editor for this paper was Z. Duric. *(Corresponding author: Hui Kong.)*

The authors are with the School of Computer Science and Engineering, Nanjing University of Science and Technology, Nanjing 210094, China (e-mail: suyingna@njust.edu.cn; lutao@njust.edu.cn; csjyang@njust.edu.cn; konghui@njust.edu.cn; zhangyigong0378@gmail.com).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TITS.2017.2751746

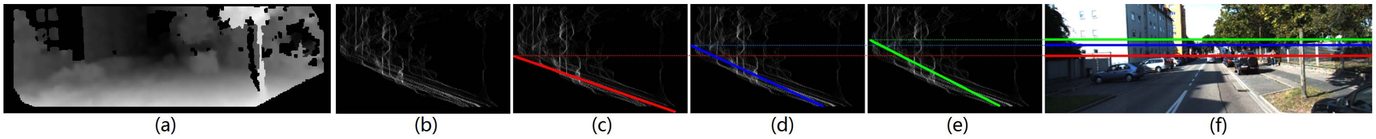


Fig. 2. A comparison example of horizon detection by three methods. (a) the disparity map. (b) the v-disparity map. (c) horizon detection by the WS method. (d) by the RS method. (e) by the HT method. (f) the original road image overlaid by three corresponding horizons.

Kim [8] proposed a method to group the detected lane-marking pixels into cubic splines of five control points. Whereafter, he proposed to group lane-markings into uniform cubic splines of two to four control points in [14]. The parametric models are sensitive to occlusions and shadows.

More recently, there has been a trend on exploiting deep-learning for lane detection [15]–[17]. Alvarez *et al.* [15] proposed to use a convolutional neural network to learn high-order features from noisy labels for road scene segmentation. Brust *et al.* [16] presented convolutional patch networks that are learned to distinguish different image patches. Oliveira *et al.* [17] proposed an improved architecture to increase the efficiency of the segmentation networks by parameter reduction and additional refinements.

III. The Proposed Method

A. Horizon Detection by a Stereo Camera

The first step of our method is to estimate the horizon. Assuming that the road region is a plane in the 3D world, the vanishing point of a road image should lie on the horizon. Therefore, it should be very helpful for vanishing point detection if we can estimate the location of the horizon.

First, we generate a disparity map from the stereo camera by the ELAS algorithm [18]. The ELAS algorithm first computes the disparities of a sparse set of 'support points' which can be robustly matched due to their texture and uniqueness. For robustness, the correspondences are retained only if they can be matched from left to right and right to left. Then, the image coordinates of the support points are used to create a 2D mesh via Delaunay triangulation. To compute the disparity at the other locations, a prior and an image likelihood are computed to decrease stereo matching ambiguities. This allows for efficient exploitation of the disparity search space, yielding accurate dense disparity map without global optimization. Finally, the dense disparity map is computed based on maximum a-posteriori estimation. The post-processing, including the left/right consistency check, small segments removal and filling, can further improve the disparity result. Figure 2 shows the dense disparity map of a road image.

Then, the v-disparity image [19] is computed. In the v-disparity image, the planar road region can be projected as a straight line. Obviously, the pixels on the horizon are at a distance of infinity from the camera and therefore the disparity is 0. Thus the position of the horizon is the intercept of the profile line on the vertical axis in the v-disparity image.

The popular methods of line detection are the Hough transform method ('HT' for short) and the RANSAC method ('RS' for short). In this paper, we use a weighted-sampling approach ('WS' for short) based on the RS method to robustly detect the road profile line. In the sampling stage, we take a weighted-sampling strategy instead of random sampling, in which the weight of each pixel is proportional to the corresponding value of the v-disparity image. So the pixel with higher value (i.e., more likely to be the pixel of the road profile line) has a greater possibility of being selected, which makes the process more efficient. In the fitting stage, a thresholding is needed first in the RS and the HT methods to get a binary image to reduce the sampling space and running time. Therefore, the accuracy of the profile line detection is dependent on the selection of the threshold value. Now we provide a more effective metric by combining the value of a pixel with its distance to the line. The closer to the fitted line the pixel is and the bigger the pixel value is, the more the pixel contributes to the line. Figure 2 shows a comparison example between the results of the WS, the RS and the HT methods. As shown, the WS method can detect the horizon more accurately.

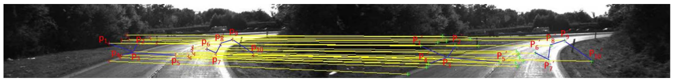


Fig. 3. The consistency of motion of world points relative to the camera (corresponding to two camera positions).

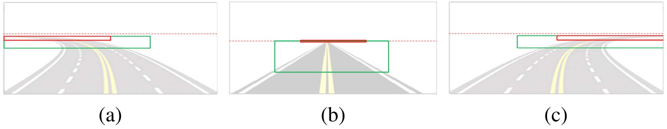


Fig. 4. Choosing of vanishing-point candidates (the red rectangle) and voters (the green rectangle) for three vehicle states. (a) left turn. (b) going straight. (c) right turn.

B. Odometry-Based Vanishing Point Detection

The second step of our method is to find the vanishing point of a road region. We give an improved texture-based method, where we use the voting scheme proposed by Kong *et al.* [20]. In order to improve the accuracy and decrease the computational complexity, we aim to compress the search space for vanishing point detection. The horizon obtained in the above section can limit the search to several rows around the horizon. If the vehicle state (left turn, right turn or going straight) is known, the searching area can be compressed even further according to Fig. 4 (we will explain it later). Therefore, we propose to estimate the vehicle state by analyzing the ego-motion of its onboard cameras based on the visual odometry (VO) method. Our scheme starts with the establishment of feature correspondence in RGB and depth maps, respectively. A real example is shown in Fig.3, where the camera's motion causes the transformation of world points. The initial matches of world points are established by the association of image features in two consecutive frames, e.g., $\mathrm{P}_1, \ldots, \mathrm{P}_{10}$ corresponding to $\mathrm{P}'_1, \ldots, \mathrm{P}'_{10}$. Let the Euclidean distance between P_i and P_j be $d(\mathrm{P}_i, \mathrm{P}_j)$. Obviously, if $(\mathrm{P}_i, \mathrm{P}'_i)$, $i = 1, \ldots, 10$, are correct matches and these world points all belong to static background, $d(\mathrm{P}_i, \mathrm{P}_j)$ should be approximately equal to $d\ (\mathrm{P}'_i, \mathrm{P}'_j)$. Of course, we cannot guarantee that all the established correspondences by image feature association are correct ones. Thus we reject the false matches (outliers) based on the approximate maximum clique method proposed in our previous work [21]. Then the transformation matrix between two consecutive camera positions can be obtained from the set of inliers (correct feature correspondences) as

$$\mathrm{P}'_i = \mathbf{R}\mathrm{P}_i + \mathbf{t} + \mathbf{v}_i \tag{1}$$

TABLE I

CHOOSING OF CANDIDATES AND VOTERS

State	Candidates	Voters
$\mathcal{L}$	$ROI(0, hl + \frac{h}{50}, \frac{w}{2}, \frac{h}{20})$	$ROI(0, hl + \frac{h}{50}, \frac{2w}{3}, \frac{h}{10})$
$\mathcal{R}$	$ROI(\frac{w}{2}, hl + \frac{h}{50}, \frac{w}{2}, \frac{h}{20})$	$ROI(\frac{w}{3}, hl + \frac{h}{50}, \frac{2w}{3}, \frac{h}{10})$
$\mathcal{S}$	$ROI(\frac{w}{3}, hl - \frac{h}{60}, \frac{w}{3}, \frac{h}{30})$	$ROI(\frac{w}{4}, hl, \frac{w}{2}, \frac{h}{3})$

where $\mathbf{R}$ is a standard 3×3 rotation matrix, $\mathbf{t}$ being a 3D translation vector, and $\mathbf{v}_i$ being the noise vector. Solving for the optimal transformation $\widehat{\mathbf{R}}$ and $\widehat{\mathbf{t}}$ can be the minimization of the following equation

$$[\widehat{\mathbf{R}}, \widehat{\mathbf{t}}] = \arg\min_{\mathbf{R},\mathbf{t}} \sum_{i=1}^{N} ||\mathrm{P}_i' - (\mathbf{R}\mathrm{P}_i + \mathbf{t})||^2 \tag{2}$$

Once we have obtained the rotation matrix and the translation vector, the orientation deviation of the vehicle between two frames can be derived from them. We can get the vehicle state by analyzing the orientation deviation of several continuous frames. The criteria are as follows:

$$VehicleState = \begin{cases} \mathcal{L} & \text{if } \bar{\Theta} < -2 \textit{ and } \sigma\ (\Theta) < 5 \\ \mathcal{R} & \text{if } \bar{\Theta} > 2 \textit{ and } \sigma\ (\Theta) < 5 \\ \mathcal{S} & \textit{others} \end{cases} \tag{3}$$

where $\bar{\Theta}$ and $\sigma\ (\Theta)$ are the mean and variance of the vehicle's orientation deviations of k continuous frames (set to 6 in the experiments), respectively. $\mathcal{L}$ means turning left, $\mathcal{S}$ meaning going straight, $\mathcal{R}$ meaning turning right in Eq. 3.

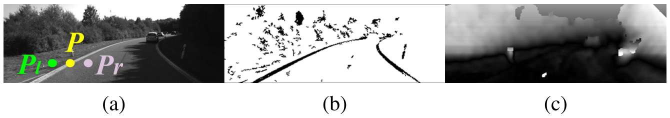
Now that we have obtained the vehicle state, the next step is to detect the vanishing point. Similar to Kong' approach [22], we adopt the voting mechanism to find the vanishing point. Specifically, we need to choose the vanishing-point candidates and the "voters", respectively. The voters are the pixels which vote for each candidate. The candidate which gets the highest score is the vanishing point. Obviously, the selection of candidates and voters is important and can affect the efficiency and accuracy of the algorithm, especially in the case of curvy road. We quantitatively choose the regions of voters and vanishing-point candidates as in Table I. The w and h represent the width and height of the road image, respectively. The $ROI(x, y, W, H)$ is a rectangular region of width W and height H with (x, y) being the upper-left corner.

Figure 4 visually describes the selection of candidates and voters for three different vehicle states. The red dashed line is the estimated horizon in section III-A. The red and green rectangular regions are the regions of candidates and voters, respectively. The motivation triggering us to choose the above voter and candidate regions is as follows: For curved road, the localization of vanishing point is strongly dependent on the texture orientation of a small area just near the horizon. For straight road, a decent-sized image area below horizon plays the major role. For straight road, the horizon detection is more accurate. Therefore, we only choose a couple of rows of pixels around the horizon as the candidates. For curved road, we expand the range of candidates a little more to tolerate the errors in horizon detection.

After defining the candidates and voters, each pixel in the voter region votes for each candidate according to Eq. 4:

$$VpScore = \exp(\frac{-d(p_c, p_v) \cdot |\theta_v - \theta_{vc}|}{\sqrt{h^2 + w^2}}) \tag{4}$$

where $d(p_c, p_v)$ is the distance between the pixel p_c (the candidate) and p_v (the voter). θ_v is the texture orientation at p_v and θ_{vc} is the direction of the vector from p_v to p_c.



(a) (b) (c)

Fig. 5. (a) The original image. (b) the lane-marking cost map. (c) the disparity cost map.

C. The Vanishing Point Constrained Minimum-Cost Map

The final step of our method is to find the lane borders by an improved Dijkstra shortest-path model, where the image could be considered as a graph $G = (V, E)$, $V = \{p_1, p_2, \ldots, p_n\}$ being the set of pixels in the image, and E representing the weighted edges between each pixel and its 8-neighbors. The weight of each edge is determined by the similarity between the neighboring pixels. The optimal path between two nodes (pixels) is defined as the minimum cumulative-cost path where the cumulative cost is the sum of the local costs (or edge links) on the path. Our goal is to find the optimal paths that correspond to the lane borders. The search process of the optimal paths mainly consists of two stages, the definition of the local costs and the computation of the minimum-cost map.

1) The Definition of the Local Costs: Inspired by [23], we define the local cost function as a weighted sum of a couple of component cost functions, which makes pixels on lane borders have low local costs. In [23], the local cost functions are computed as the sum of three components: the gradient magnitude function, $F_{gm}(p)$, the gradient direction function, $F_{gd}(p, q)$, and the Laplacian zero crossing function, $F_z(p)$. $F_{gm}(p)$ is inversely proportional to the gradient magnitude of the pixel p. $F_{gd}(p, q)$ adds a smoothness constraint between p and q by associating a high cost for sharp direction changes at p and q. The value of $F_{gd}(p, q)$ is small when the gradient directions at both pixels are consistent with that of the link between them. $F_z(p)$ is a binary function which equals 0 at edge pixels and 1 otherwise. These definitions guarantee that the edge pixels have lower costs. However, in our case we only want to detect the edges of lane borders. Thus we add two component costs, the lane cost, $F_l(p)$, and the disparity cost, $F_d(p)$, to make the local cost function applicable to the lane detection.

The lane-marking cost function, $F_l(p)$, is a binary function in which the value equal to 0 represents that p is likely to be lane-marking pixel, 1 otherwise. In general, lane marking is a dark-light-dark transition pattern. That is, lane-marking pixels should be brighter than the surrounding non-marking area. As labeled in Fig. 5 (a), p is a lane-marking pixel and p_l, p_r are two non-marking pixels on the left and right side, respectively. Obviously, for a lane-marking pattern, the difference between the intensity at p and p_l (p_r) should be large, and small between the intensity at p_l and p_r. We calculate $F_l(p)$ according to these difference as:

$$F_l(p) = \begin{cases} 0 & \text{if } |p - p_l| + |p - p_r| - |p_l - p_r| > 20 \\ 1 & \textit{others} \end{cases} \tag{5}$$

where $|p - p_l|$ means the difference between the intensity at p and p_l. Figure 5(b) shows the lane-marking cost map.

The disparity cost function, $F_d(p)$, calculates a low cost for pixels in flat region and a large cost for pixels in non-flat region. For a pixel p at the i-th row, we check the difference between its disparity and the corresponding value on the detected road profile line in the

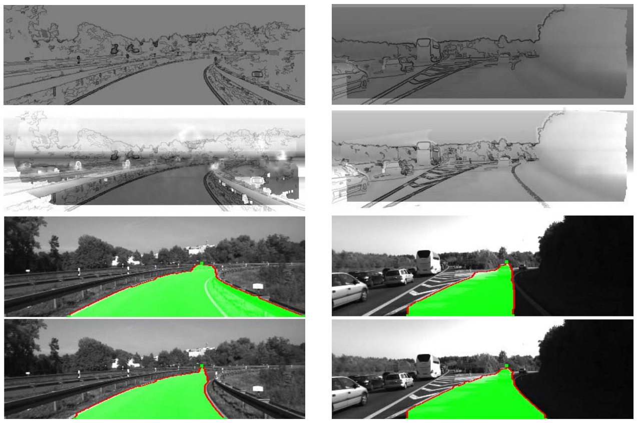

(a) (b)

Fig. 6. Comparison results with and without the improved local cost map. (a) from top to bottom: the local cost map without $F_l(p)$ only, the improved local cost map, lane result without $F_l(p)$, lane result with $F_l(p)$. (b) from top to bottom: the local cost map without $F_d(p)$ only, the improved local cost map, lane result without $F_d(p)$, lane result with $F_d(p)$.

v-disparity map. The larger the difference is, the less likely it is to be a road pixel, and therefore the bigger the F_d should be.

$$F_d(p) = |D_{map}(p) - ProfileLine(p)|/d_{max} \tag{6}$$

where $D_{map}(p)$ means the disparity of p. $ProfileLine(p)$ is the corresponding value on the detected road-profile line in the v-disparity map. The d_{max} is the maximum disparity of the pixels at the same row as p. Figure 5 (c) shows the disparity cost map of Fig. 5 (a).

Finally, the local cost function can be calculated as follows:

$$\begin{aligned} F_{cost}(p,q) &= w_1 * F_{gd}(p,q) + w_2 * F_{gm}(p) \\ &\quad + w_3 * F_z(p) + w_4 * F_l(p) + w_5 * F_d(p) \end{aligned} \tag{7}$$

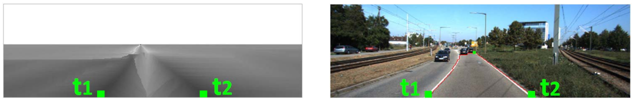
With the following settings, $w_1 = 0.3, w_2 = 0.2, w_3 = 0.1, w_4 = 0.15, w_5 = 0.25$, we obtained promising results. Figure 6 shows the lane border detection results with the improved local cost map. Obviously, the improved local cost function effectively suppresses the sharp edge of non-lane objects (such as road guardrail, curbstone *etc.*) and highlights the edges of lane borders.

2) The Computation of the Minimum-Cost Map: With the above definition of the local cost between two neighbor pixels, we apply the Dijkstra algorithm, with the detected vanishing point being the source node, to get an accumulative-cost map *PathCost*, where the value of each element represents the minimum accumulative cost between the vanishing point and the element. Meanwhile, we can also obtain the corresponding shortest path *PathPtr* between each element and the vanishing point, whose length is recorded in *PathLen*. By the element-wise division between *PathCost* and *PathLen*, we get *PathAvg*. On the last row of *PathAvg*, intuitively, we look for two minimum elements as the terminal points to the left and right of the middle one, respectively. The search criteria are:

$$tp_l = \operatorname{argmin}_{i=1}^{\lfloor \frac{w}{2} \rfloor} PathAvg(h,i) \tag{8}$$

$$tp_r = \operatorname{argmin}_{i=\lfloor \frac{w}{2} \rfloor+1}^{w} PathAvg(h,i) \tag{9}$$

where h and w are the image height and width, respectively. The tp_l and tp_r are the terminal points of the left and right lane borders, respectively. Figure 7 illustrates this process. These two minimum elements are regarded as the two terminal points. From them, we can obtain the shortest paths corresponding to two lane borders based on *PathPtr* (the optimal path from each terminal point to vp is known in *PathPtr*). Figure 8 shows some lane detection results.



(a) (b)

Fig. 7. Localizations of two terminal points. (a) Two minimum elements on the last row of *PathAvg*. (b) Two corresponding shortest paths.

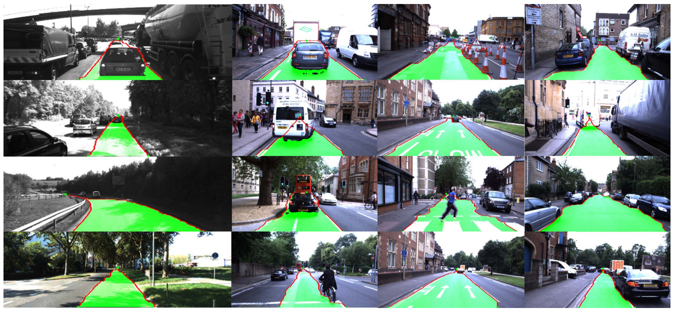
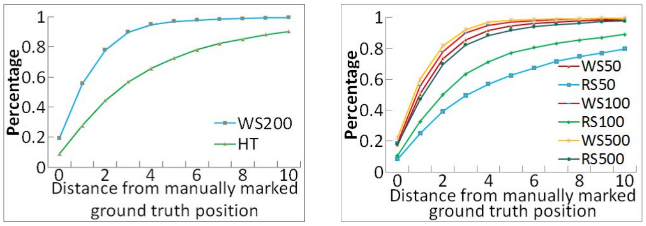
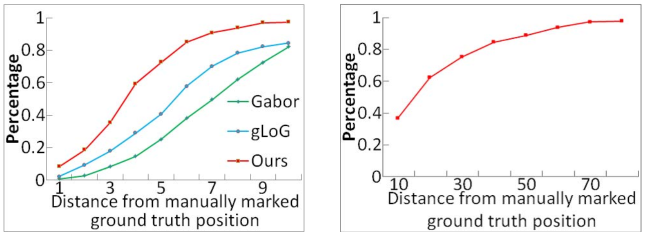


Fig. 8. The results of our lane detection method.



(a) (b)

Fig. 9. Horizon detection results. (a) “WS” vs. “HT”. (b) “WS” vs. “RS”.



(a) (b)

Fig. 10. Vanishing point estimation results. (a) For straight road. (b) For curved road.

IV. Experiments

In this section, we present the statistics to show the performance of the proposed approach in dealing with various road environments. The test images are selected from the KITTI and the Oxford RobotCar datasets which contain various road scenes. Table II gives the average computation time for each step (the horizon detection, the vanishing point detection and the vanishing point constrained minimum-cost path search) of the proposed method. Totally, more than 2700 road images are tested to evaluate the proposed method and all the test images are manually labeled to generate the ground truth data for evaluation purposes.

A. Horizon Detection

We test the method on all the selected images and compare our WR approach with the HT and the RS methods. Figure 2 shows

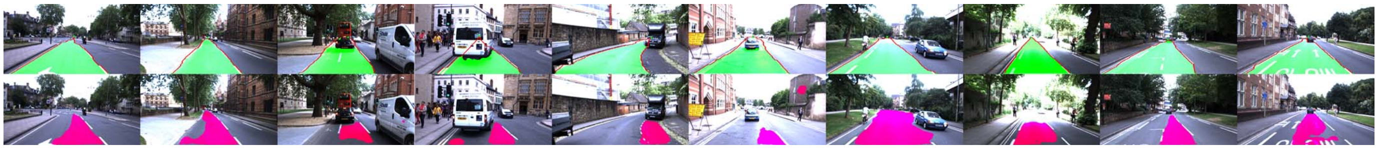


Fig. 11. Comparison results between the Up_Conv_Poly and our method. The first row is the result by our method. The second row is the result by the Up_Conv_Poly method.

TABLE II

THE RUNTIME OF THE PROPOSED METHOD (S/FRAME).

Dataset	Resolution	Horizon	Vanishing point	Lane	Total
KITTI	160×480	0.071	0.114	0.025	0.201
Oxford	221×427	0.089	0.164	0.069	0.322

a simple comparison of horizon detection by the three methods. Figure 9 gives the quantitative comparison. We compute the estimation error by comparing with the ground truth positions, and then calculate the percentage of the images whose detection error is no bigger than a threshold. Figure 9(a) shows the comparison results between the HT method and the WS method with 200 sampling times (denotes as "WS200"). Figure 9(b) shows the comparison results between the WS method and the RS method with different sampling time.

The comparison results show that the WS method is effective to improve the accuracy and reduce the running time. About 99.61% of all images have an error no bigger than ten pixels by the WS method with 200 sampling times while the percentage of the HT method is 90.54%. When the number of sampling is 50 times, the RS method can only achieve an accuracy of 78.9% with the error no bigger than ten pixels, while the WS method can achieve an accuracy of 98.69%.

In practice, we only need to detect the horizon one time to further speed up the process. Because the positions of horizon of the road images which are captured by the same camera are similar. It also helps that our method can still work even if the quality of disparity map is not very good.

B. Vanishing Point Estimation

In this section, we evaluate the performance of the proposed vanishing point estimation method. We evaluate our method on straight and curved road images, respectively. We split all the test images into straight and curved road image sets. Moreover, we compare our method with the other two popular methods: the Gabor-based (Gabor) [20] and the gLoG-filter-based (gLoG) [22] methods. Because the two compared methods are not eligible for vanishing-point detection on curved-road images, we only evaluate them on straight-road images.

In evaluating the three methods, we compute the L2-norm error of the vanishing point results by comparing with the ground truth positions. The result of our method is shown as the red plot in Fig. 10 (a) and (b), which corresponds to the accuracy on the straight- and curved-road images, respectively. We calculate the percentage of the images whose detection error is no bigger than a threshold. Figure 10 (a) also shows the comparison between ours and the other two methods. More than 97% of straight-road images can get the vanishing point with an error of no bigger than 10 pixels, and more than 76% of curved-road images with an error of no bigger than 30 pixels by our method. Compared with the other two methods, our method can significantly improve the accuracy of the vanishing point detection.

TABLE III

PERFORMANCE [%] OF OUR LANE DETECTION METHOD

Dataset	Frame	MaxF	PRE	REC	FPR	FNR
KITTI	1789	93.47	95.19	91.81	0.69	10.44
Oxford	915	91.21	89.50	94.00	1.51	7.71

TABLE IV

A COMPARISON ON KITTI

Method	MaxF	Runtime	Environment
SAIT	93.25	0.04s	GPU @ 2.5Ghz (C/C++)
Ours	91.68	0.21s	4 cores CPU @ 3.2Ghz (C/C++)
Up-Conv-Poly [17]	89.88	0.08s	GPU @ 2.5Ghz (Python+C/C++)
SPRAY [24]	83.42	0.045s	GTX 580 (Python+OpenCL)
SPlane+BL [25]	69.63	2s	1 core CPU@ 3.0Ghz (C/C++)

C. Lane Border Detection

We detect the lane region as the area enclosed by the two detected lane borders and the last row of the road image. We compare this region with the manually labeled ground truth result. Five evaluation criteria are used, including the MaxF (maximum F1-measure), PRE (precision), REC (recall), FPR (false positive ratio) and FNR (false negative ratio), all of which are defined in [1]. The MaxF is the primary one which provides an insight into an algorithm's optimal performance.

First, we test our method on images selected from the KITTI (1789 images) and the Oxford (915 images) datasets. We obtain the results based on a comparison with the manually labeled ground truth data. As shown in Table III, the proposed method can achieve an accuracy (MaxF) of 93.47% and 91.21% in the two sets of images, respectively.

We also compare our method on the KITTI-UM-Lane dataset with the state-of-the-art methods. We observed that the top-performer methods are almost based on deep-learning techniques. However, most of them (or almost all of them) do not provide source code, actually not even a reference to their methods (e.g., the SAIT method). Therefore, we cannot compare all the deep-learning based methods. The Up_Conv_Poly method is the only deep-learning one that provides source code of all the methods ever submitting their results to the KITTI lane detection benchmark. The comparison is performed in the bird's-eye-view (BEV) space. Table IV shows the comparison results in details. The SAIT method and the Up_Conv_Poly method [17] give the best performance among the reported methods and the referable articles on the KITTI-road website, respectively. Our method is more accurate compared with the Up_Conv_Poly method and the other stereo-vision based approaches [24], [25].

The performance on a single dataset cannot fully reflect the generalization ability of the compared methods. Therefore, we test the Up_Conv_Poly method and ours on the public Oxford Robotcar road dataset. We found that this deep-learning based method has a much worse performance than ours on the Oxford dataset. The MaxF of the

Up_Conv_Poly method is 71.27% while our method can achieve the MaxF of 91.21%. Figure 11 gives some comparison results between the Up_Conv_Poly and our method on some Oxford road images.

Generally, the deep-learning based methods can achieve promising performance when the test images are similar to the training ones. However, the performance of deep-learning approaches usually degrades a lot when the training road scenes are different from that of test ones, especially for lane detection task where the lanes are subject to large changes in widths and layouts, and images are captured by different cameras with varying focal lens, field of view angles, resolutions, aspect ratios *etc*.

V. Conclusions

In this paper, we proposed a vanishing point constrained lane detection method which performs accurately for a variety of road conditions. The proposed method is based on stereo vision, using visual odometry and v-disparity, to find the position of the vanishing point, and then performs a vanishing-point constrained Dijkstra graph search to generate a minimum-cost map. In this map, the optimal path from each point to the vanishing point is known. The lane border corresponds to the path which has the minimum cost. Experiments show that the proposed method can achieve promising results.

References

[1] A. Geiger, P. Lenz, and R. Urtasun, “Are we ready for autonomous driving? The KITTI vision benchmark suite,” in *Proc. IEEE CVPR*, Jun. 2012, pp. 3354–3361.

[2] W. Maddern, G. Pascoe, C. Linegar, and P. Newman, “1 year, 1000 km: The Oxford Robotcar dataset,” *Int. J. Robot. Res.*, vol. 36, no. 1, pp. 3–15, 2017.

[3] H. Kong, J. Y. Audibert, and J. Ponce, “Vanishing point detection for road detection,” in *Proc. IEEE Conf. CVPR*, Jun. 2009, pp. 96–103.

[4] Y. Wang, N. Dahnoun, and A. Achim, “A novel system for robust lane detection and tracking,” *Signal Process.*, vol. 92, no. 2, pp. 319–334, 2012.

[5] T.-Y. Sun, S.-J. Tsai, and V. Chan, “HSI color model based lane-marking detection,” in *Proc. IEEE Intell. Transp. Syst. Conf.*, Toronto, ON, Canada, Sep. 2006, pp. 1168–1172.

[6] Z. Ying and G. Li, “Robust lane marking detection using boundary-based inverse perspective mapping,” in *Proc. IEEE Int. Conf. Acoust., Speech Signal Process. (ICASSP)*, Mar. 2016, pp. 1921–1925.

[7] U. Ozgunalp and N. Dahnoun, “Robust lane detection & tracking based on novel feature extraction and lane categorization,” in *Proc. IEEE Int. Conf. Acoust., Speech Signal Process. (ICASSP)*, May 2014, pp. 8129–8133.

[8] Z. Kim, “Realtime lane tracking of curved local road,” in *Proc. IEEE Conf. Intell. Transp. Syst.*, Sep. 2006, pp. 1149–1155.

[9] J. Deng and Y. Han, “A real-time system of lane detection and tracking based on optimized RANSAC B-spline fitting,” in *Proc. Res. Adapt. Converg. Syst.*, 2013, pp. 157–164.

[10] J. Ruyi, K. Reinhard, V. Tobi, and W. Shigang, “Lane detection and tracking using a new lane model and distance transform,” *Mach. Vis. Appl.*, vol. 22, no. 4, pp. 721–737, 2011.

[11] M. Bertozzi and A. Broggi, “GOLD: A parallel real-time stereo vision system for generic obstacle and lane detection,” *IEEE Trans. Image Process.*, vol. 7, no. 1, pp. 62–81, Jan. 1998.

[12] H. Yoo, U. Yang, and K. Sohn, “Gradient-enhancing conversion for illumination-robust lane detection,” *IEEE Trans. Intell. Transp. Syst.*, vol. 14, no. 3, pp. 1083–1094, Sep. 2013.

[13] Y. Wang, E. K. Teoh, and D. Shen, “Lane detection and tracking using B-snake,” *Image Vis. Comput.*, vol. 22, no. 4, pp. 269–280, 2004.

[14] Z. Kim, “Robust lane detection and tracking in challenging scenarios,” *IEEE Trans. Intell. Transp. Syst.*, vol. 9, no. 1, pp. 16–26, Mar. 2008.

[15] J. M. Alvarez, T. Gevers, Y. LeCun, and A. M. Lopez, “Road scene segmentation from a single image,” in *Proc. Eur. Conf. Comput. Vis. (ECCV)*, 2012, pp. 376–389.

[16] C. A. Brust, S. Sickert, and M. Simon. (2015). “Convolutional patch networks with spatial prior for road detection and urban scene understanding.” [Online]. Available: https://arxiv.org/abs/1502.06344

[17] G. L. Oliveira, W. Burgard, and T. Brox, “Efficient deep methods for monocular road segmentation,” in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS)*, Oct. 2016, pp. 1–7.

[18] A. Geiger, M. Roser, and R. Urtasun, “Efficient large-scale stereo matching,” in *Computer Vision*. Berlin, Germany: Springer, 2010, pp. 25–38.

[19] R. Labayrade, D. Aubert, and J.-P. Tarel, “Real time obstacle detection in stereovision on non flat road geometry through ‘v-disparity’ representation,” in *Proc. IEEE Intell. Vehicle Symp.*, Jun. 2002, pp. 646–651.

[20] H. Kong, J.-Y. Audibert, and J. Ponce, “General road detection from a single image,” *IEEE Trans. Image Process.*, vol. 19, no. 8, pp. 2211–2220, Aug. 2010.

[21] H. Kong, J. Yang, and Z. Chen, “Accurate and efficient inspection of speckle and scratch defects on surfaces of planar products,” *IEEE Trans. Ind. Informat.*, vol. 13, no. 4, pp. 1855–1865, Aug. 2017.

[22] H. Kong, S. E. Sarma, and F. Tang, “Generalizing Laplacian of Gaussian filters for vanishing-point detection,” *IEEE Trans. Intell. Transp. Syst.*, vol. 14, no. 1, pp. 408–418, Mar. 2013.

[23] E. N. Mortensen and W. A. Barrett, “Interactive segmentation with intelligent scissors,” *Graph. Models Image Process.*, vol. 60, no. 5, pp. 349–384, 1998.

[24] N. Einecke and J. Eggert, “Block-matching stereo with relaxed fronto-parallel assumption,” in *Proc. IEEE Intell. Vehicles Symp.*, Jun. 2014, pp. 700–705.

[25] I. V. Gheorghe, “Semantic segmentation of terrain and road terrain for advanced driver assistance systems,” Ph.D. dissertation, Dept. Eng. Comput., Coventry Univ., Coventry, U.K., 2015.