Low-Level Image Processing for Lane Detection and Tracking

Ruyi Jiang[1], Mutsuhiro Terauchi[2], Reinhard Klette[3], Shigang Wang[1], and Tobi Vaudrey[3]

[1] Shanghai Jiao Tong University, Shanghai, China
{jiangruyi,wangshigang}@sjtu.edu.cn
[2] Hiroshima International University, Japan
mucha@he.hirokoku-u.ac.jp
[3] The University of Auckland, Auckland, New Zealand
{r.klette,t.vaudrey}@auckland.ac.nz

Abstract. Lane detection and tracking is a significant component of vision-based driver assistance systems (DAS). Low-level image processing is the first step in such a component. This paper suggests three useful techniques for low-level image processing in lane detection situations: bird's-eye view mapping, a specialized edge detection method, and the distance transform. The first two techniques have been widely used in DAS, while the distance transform is a method newly exploited in DAS, that can provide useful information in lane detection situations. This paper recalls two methods to generate a bird's-eye image from the original input image, it also compares edge detectors. A modified version of the Euclidean distance transform called *real orientation distance transform* (RODT) is proposed. Finally, the paper discusses experiments on lane detection and tracking using these technologies.

Keywords: Lane detection and tracking, DAS, bird's-eye view, distance transform.

1 Introduction

Lane detection plays a significant role in driver assistance systems (DAS), as it can help estimate the geometry of the road ahead, as well as the lateral position of the ego-vehicle on the road [7]. Lane detection is used in intelligent cruise control systems, for lane departure warning, road modelling, and so on. Typically, lane detection and tracking are used for localizing lane boundaries in given road images. In general, a procedure for lane detection includes low-level image processing as pre-processing, lane boundary detection, and some post-processing.

The first step in lane detection is low-level image processing (LLIP), which deals with the input image [from the camera(s) installed on the ego-vehicle] and generates useful information for the detection part.

For edge-based lane detection, three relative stages are to be considered in LLIP. First, a transformation of the input image as lane detection has been conducted (to a large extent) directly on the input image. However, a proper transformation (e.g., a homographic transform) of the input image, prior to further analysis, will facilitate and improve the accuracy of lane detection. Secondly, an edge detection method is

F. Huang and R.-C. Wang (Eds.): ArtsIT 2009, LNICST 30, pp. 190–197, 2010.

performed. All-purpose edge operators (e.g., Sobel, Canny) are also applied for lane detection, some specially designed operators will fit better to the properties of lane marks, which are to be detected. Third, some other operations on the edge map will be discussed below.

Complete LLIP will not always include these three stages according to various lane detection situations and applied methods. This paper will discuss three useful technologies that can be applied as LLIP in lane detection and tracking. Bird's-eye view mapping has been widely used in lane detection [1,5] and robot navigation [4], as it can provide a top-down view through a homographic transform. A specialized edge detector, as introduced in [1], is recalled and compared with other gradient-based edge operators. Furthermore, this paper fully exploits the advantages of distance transforms for lane detection as a powerful way to represent relevant information.

This work is organized as follows: Section 2 discusses bird's-eye view mapping, and provides two methods for such a transformation. Section 3 describes an edge detection method. Section 4 recalls distance transform and discusses its useful properties for lane detection. Experimental results of lane detection and tracking are presented in Section 5 using LLIP technologies as introduced in this paper.

2 Bird's-Eye View Mapping

Generally, detecting lanes directly on the input image will waste no time over transforming the input image into some other space. However, as introduced in [1], at least two disadvantages will come with apparent perspective effects: non-constant lane mark width, and different distances (see Fig. 1). A removal of such perspective effects will greatly facilitate lane detection. The main function of a bird's-eye view mapping is that a remapped image represents a (scaled) view from the top towards an assumed planar ground manifold. From this remapped view, perspective effects are approximately removed (i.e., approximate with respect to the assumed planarity). Two methods are briefly reviewed here to generate a bird's-eye image from the input image.

2.1 Warp Perspective Mapping

As in [5], a four-points correspondence can be used for the mapping from the input image into the bird's-eye image. The mapping is achieved by selecting four points P_W on the ground plane when calibrating the ego-vehicle's camera(s), and by using the

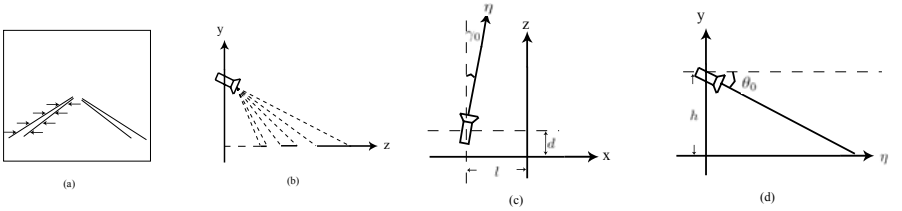

Fig. 1. Perspective effects in the original input image: (a) Non-constant lane mark width, (b) Variation of distances to different pixels. The model of IPM: (c) The xz plane in space $\mathbb{W}$, (d) The $y\eta$ plane.

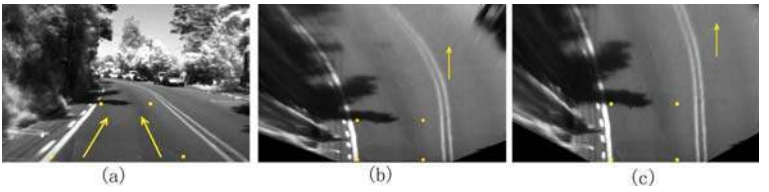

Fig. 2. (a) Input Image. (b) and (c) are bird's-eye images using a WPM but based on different distance definitions. Four-point correspondence (points shown in yellow) is established by calibration; the driving direction is indicated by the arrow.

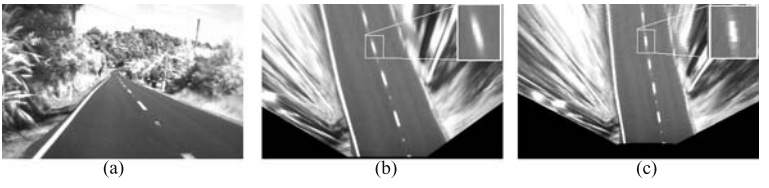

Fig. 3. Bird's-eye view using warp perspective mapping and inverse perspective mapping. (a) The original input image. (b) Bird's-eye view using WPM. (c) Bird's-eye view using IPM. In general, those views generated by WPM appear to be more blurred, and IPM is more pixelated.

planar ground manifold assumption. P_W in the input image will be viewed as P_I. The mapping from P_I to P_W can be achieved by an affine matrix $\boldsymbol{A}$ as $P_W = \boldsymbol{A} \cdot P_I$. The calculation of $\boldsymbol{A}$ using the above equation can be easily achieved. Then, the whole input image can be mapped, pixel-by-pixel, using $\boldsymbol{A}$. One main benefit of this warp perspective mapping (WPM) is that the used distance scale can be adjusted by selecting different sets of four corresponding points (i.e., by scaling the "length" of the rectangle). This proved to be useful for detecting discontinuous lane markers as well as for further forward looking situations. Another benefit is that no intrinsic or extrinsic parameters of camera(s) are needed.

2.2 Inverse Perspective Mapping

Inverse perspective mapping (IPM) as introduced in [6] is another way to generate a bird's-eye view from the input image [1,2]. The knowledge of the camera parameters (extrinsic and intrinsic) is required for the application of the IPM transform:

Viewpoint: camera position in world coordinate system $C = (l, h, d)$.
Viewing direction: optical axis is defined by two angles:
γ_0: the angle formed by the projection of the optical axis on the xz plane [as shown in Fig. 1(c)],
θ_0: the angle formed by the optical axis and axis η [as shown in Fig. 1(d)].
Aperture: camera angular aperture is $2\alpha_u$ in row direction and $2\alpha_v$ in column direction. While [note that (u_0, v_0) is the camera's focal center, and F is focal length]: $\alpha_u = \arctan\{u_0/F\}$, $\alpha_v = \arctan\{v_0/F\}$.
Resolution: camera resolution is $n \times m$.

Mathematically, IPM can be modeled as a projection from a 3D Euclidean space $\mathbb{W}$, containing elements $(x, y, z) \in \mathbb{R}^3$, onto a planar (2D) subspace of $\mathbb{R}^3$, denoted by $\mathbb{I}$, with elements $(u, v) \in \mathbb{R}^2$. The mapping from $\mathbb{I}$ to $\mathbb{W}$ is as follows,

$$x(u,v) = h \cdot \cot\{(\theta_0 - \alpha_u) + u\frac{2\alpha_u}{m-1}\} \cdot \cos\{(\gamma_0 - \alpha_v) + v\frac{2\alpha_v}{n-1} + l\}$$
$$y(u,v) = 0$$
$$z(u,v) = h \cdot \cot\{(\theta_0 - \alpha_u) + u\frac{2\alpha_u}{m-1}\} \cdot \sin\{(\gamma_0 - \alpha_v) + v\frac{2\alpha_v}{n-1} + d\}$$

while we have the following mapping from $\mathbb{W}$ to $\mathbb{I}$:

$$u(x,0,z) = \frac{[\arctan\{\frac{h\sin\gamma(x,0,z)}{z-d}\} - (\theta_0 - \alpha_u)] \cdot (m-1)}{2\alpha_u}$$
$$v(x,0,z) = \frac{[\arctan\{\frac{z-d}{x-l}\} - (\gamma_0 - \alpha_v)] \cdot (n-1)}{2\alpha_v}$$

An example of bird's-eye view using the above two methods is shown in Fig. 3. Both of them are using the planar ground plane assumption, and parameters from calibration. However, there are some difference between these two methods. First, WPM will not directly use camera's intrinsic or extrinsic parameters as IPM does. That means the calibration will be much easier for WPM. Second, changing a bird's-eye view generated from WPM means selecting another set of four points, while for IPM, the parameters of the camera need to be changed. Also, the quality of the generated bird's-eye views is different, which can be seen in the small (also enlarged) window in Fig. 3.

3 Edge Detection and Denoising

We recall an edge detection method as introduced in [1]. Black-white-black edges in vertical direction are detected in the bird's-eye image by a specially designed simple algorithm. Every pixel in the bird's-eye image, with value $b(x, y)$, is compared to values $b(x-m, y)$ and $b(x+m, y)$ of its horizontal left and right neighbors at a distance $m \geq 1$ as follows:

$$B_{+m}(x,y) = b(x,y) - b(x+m,y)$$
$$B_{-m}(x,y) = b(x,y) - b(x-m,y)$$

Finally, using a threshold T, the edge map value will be

$$r(x,y) = \begin{cases} 1, & \text{if } B_{+m} \geq 0,\ B_{-m} \geq 0, \text{ and } B_{+m} + B_{-m} \geq T \\ 0, & \text{otherwise} \end{cases}$$

This edge detection method has the following properties. First, m can be adjusted to fit various widths of lane marks. Second, pixels within a lane mark are all labeled as being edge pixels, which is different from gradient-based edge operators (e.g., Sobel, Canny). This greatly improves the robustness in detecting points at lane marks. Third, shadows on the road surface do not influence edge detection at lane marks. Thus, the edge detection method can be used under various illumination conditions. Finally, horizontal edges are not detected. For an example of edge detection, see Figure 4.

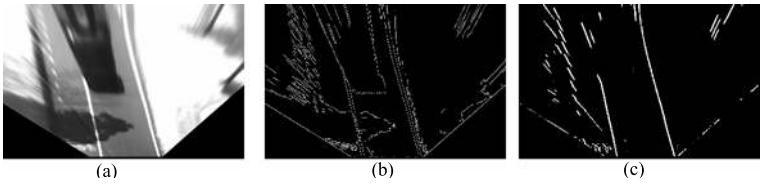

Fig. 4. Edge detection. (a) Bird's-eye image. (b) Edge detection using the Canny operator. (c) Edge detection following [1].

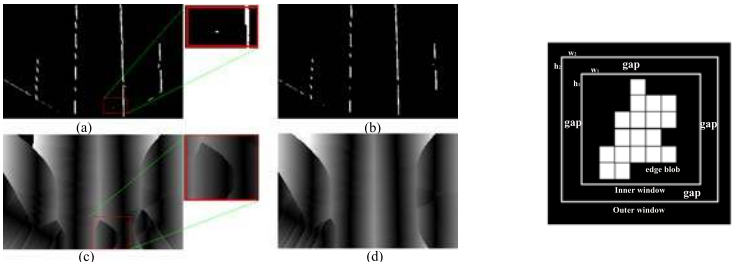

Fig. 5. Effect of an isolated noisy edge pixel for the distance transform. (a) Noisy edge map with an isolated edge point in the middle of the lane. (b) Denoised edge map. (c) RODT based on (a). (d) RODT based on (b). The operator to remove blobs is shown on the right.

The edge detection scheme as discussed above may generate some isolated small blobs (including single pixels) beside edges of real lane marks. These noisy blobs will greatly affect the result of the subsequent distance transform (see Fig. 5; the distance transform is discussed in Section 4). In order to remove such noise, a specified operation is applied. The general idea is first to find such isolated blobs, and then set them to zero (i.e., to the non-edge value). Two small windows (inner and outer) are used (see Fig. 5) at the same reference pixel, but the outer window is slightly larger than the inner one in width and height. The isolated blobs can be detected by moving at these two windows through the whole edge map, and comparing the sums of edge values within them. If two sums are equal, that means that the gap between two windows contains no edge points, then the edge blobs in the inner window are detected as being isolated, and set to zero. For computation efficiency, an integral image of the edge map is used for calculating the sum in the windows.

4 Distance Transform

The distance transform applied to the binary edge map labels each pixel with the distance to the nearest edge pixel (see [8] for details). Edge pixels are obviously labeled by 0, and this is shown as black in the generated distance map. Pixels "in the middle of a lane" are supposed to receive large labels, shown as bright pixels in the distance map (see Figure 6).

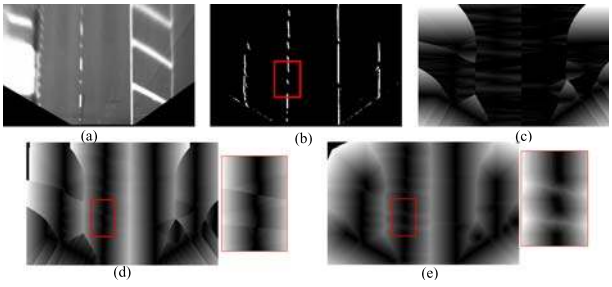

Fig. 6. EDT and ODT on a bird's-eye road image. (a) Bird's-eye image. (b) Edge map (the area in the rectangle is a discontinuous lane mark). (c) EDT. (d) Real part of ODT (absolute value). (e) Imaginary part of ODT. (c)(d)(e) have been contrast adjusted for better visibility.

The Euclidean distance transform (EDT) is, in general, the preferred option; using the Euclidean metric for measuring the distance between pixels. [3] proved that a 2D EDT can be calculated by two 1D EDTs, and this greatly improves the computational efficiency. A modified EDT was proposed in [9], called the *orientation distance transform* (ODT). This divides the Euclidean distance into a contributing component in the row and column direction. (Note that the use of a 4- or 8-distance transform would *not* lead to the same row and column components; however, practically there should not be a big difference with respect to the given context.) A complex number is assigned to each pixel by the ODT, with the distance component in the row direction as the real part, and the distance component in the column direction as the imaginary part. Then the magnitude and the phase angle of such a complex number, at a non-edge pixel, represent the Euclidean distance and the orientation to the nearest edge pixel, respectively. Note that the distance component in row direction is signed, with a positive value indicating that the nearest edge point lies to the right, and a negative value if it is to the left. See Figure 6 for an example. The imaginary part is mostly dark because the nearest edge pixels are, in general, in the same row; due to the applied edge detection method. Thus, we decided to ignore the imaginary part.

This paper uses only the Euclidean distance in row direction, and we call this the *real orientation distance transform* (RODT). The RODT of our edge map offers various benefits. First, every pixel indicates the nearest edge pixel, which will greatly facilitate the edge-finding procedure. Secondly, discontinuous lane marks will make almost no difference with continuous ones in the RODT (and this is different to the EDT of the edge map), as illustrated in Figure 6. Third, more information about the lane is provided by the distance transform compared to the edge map. For example, there's nothing provided by the edge map for the (virtual) centerline of a lane. While in distance map, pixels on the centerline will be assigned a large distance value.

The distance transform is sensitive to some isolated points or blobs in lane detection situation. As indicated in Fig. 5, an edge point on the middle of lane will greatly change the distance value for the surrounding pixels. So a denoising method on the edge map as introduced in Section 3 is necessary, and proves to be useful.

5 Experiments

Experimental results (using the test vehicle HAKA1) for lane detection (see [10]) are shown in Fig. 7, and for lane tracking in Fig. 8. The size of images used is 752×480, recorded at 25 Hz.

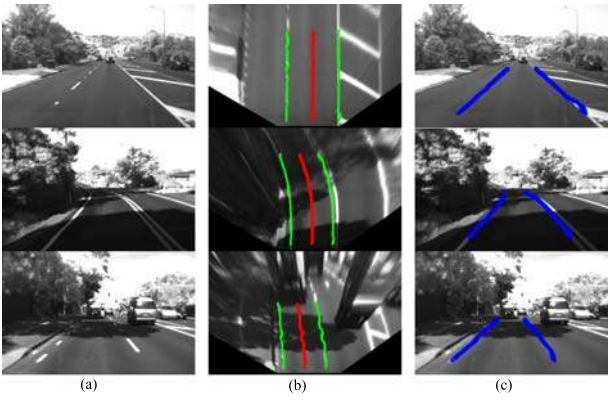

Fig. 7. Experimental results for lane detection. (a) Input images. (b) Lanes detected in the bird's-eye image. (c) Lanes detected.

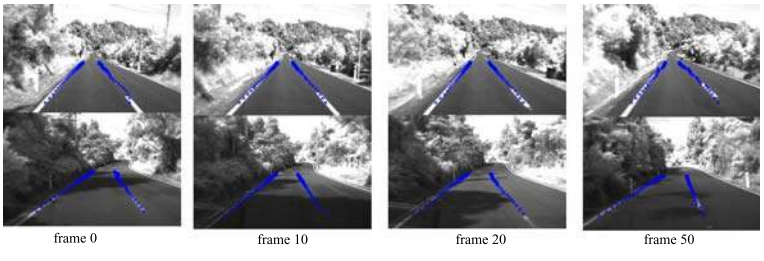

Fig. 8. Experimental results for using the robust lane tracking technique

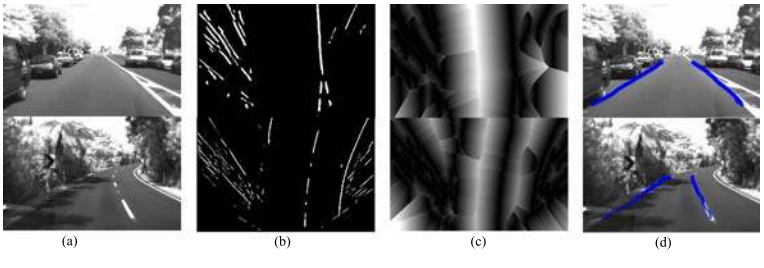

Fig. 9. Two examples of usefulness of RODT in detection of a lane boundary. (a) The input image. (b) The edge map. (c) RODT of (b). (d) Lane detection result.

The usefulness of distance transform can be seen from examples in Fig. 9. Lane marks are not apparent (or occluded) on the left hand side in these two situations, but the distance map provides sufficient information for identifying the lane boundaries.

Acknowledgments. This work is supported by the National Natural Science Foundation of China under Grant 50875169.

References

1. Bertozzi, M., Broggi, A.: GOLD - A parallel real-time stereo vision system for generic obstacle and lane detection. IEEE Trans. Image Processing. 7, 62–81 (1998)
2. Bertozzi, M., Broggi, A., Fascioli, A.: Stereo inverse perspective mapping: theory and applications. Image and Vision Computing 16, 585–590 (1998)
3. Felzenszwalb, P.F., Huttenlocher, D.P.: Distance transform of sampled functions. Technical report, Cornell Computing and Information Science (2004)
4. Gaspar, J., Winters, N., Santos-Victor, J.: Vision-based navigation and environmental representation with an omnidirectional camera. IEEE Trans. Robotics and Automation 16, 890–898 (2000)
5. Kim, Z.: Robust lane detection and tracking in challenging scenarios. IEEE Trans. Int. Trans. Sys. 9, 16–26 (2008)
6. Mallot, M.A., Hulthoff, H.H.B., Little, J.J., Bohrer, S.: Inverse perspective mapping simplifies optical flow computation and obstacle detection. Biological Cybernetics 64, 177–185 (1991)
7. McCall, J.C., Trivedi, M.M.: Video-based lane estimation and tracking for driver assistance: survey, system, and evaluation. IEEE Trans. Int. Trans. Sys. 7, 20–37 (2006)
8. Klette, R., Rosenfeld, A.: Digital Geometry. Morgan Kaufmann, San Francisco (2004)
9. Wu, T., Ding, X.Q., Wang, S.J., Wang, K.Q.: Video object tracking using improved chamfer matching and condensation particle filter. In: SPIE-IS & T Electronic Imaging, vol. 6813, pp. 04.1–04.10 (2008)
10. Jiang, R., Klette, R., Wang, S., Vaudrey, T.: Lane detection and tracking using a new lane model and distance transform. Technical Report, The University of Auckland (2009)