Edge filter for road white-line detection using brightness gradient approximation by discrete values

Hiroyuki FURUSHOU* and Hiroshi MOURI**

* ITS Research Division, Japan Automobile Research Institute
1-1-30 Shibadaimon, Minato-ku, Tokyo 105-0012, Japan
E-mail: hfurushou@jari.or.jp

** Department of Mechanical Systems Engineering, Tokyo University of Agriculture and Technology
2-24-16 Naka-cho, Koganei-shi, Tokyo 184-8588, Japan

Received: 15 February 2017; Revised: 25 April 2017; Accepted: 4 June 2017

Abstract

We propose an edge filter for road white-line detection. Many methods for white-line detection have been proposed for standardized roads. These methods are composed of two stages of processing, i.e., edge detection on an image taken with a camera and then extraction of edge clusters of white-line contours by model fitting. It is difficult to apply these methods to non-standardized roads of for which modeling are difficult. To expand the scope of white-line detection to common roads in the future, it is necessary to achieve processing through clustering without models. However, while clustering can apply to diverse contour lines, there is concern about degrading noise reduction that has so far been done by model fitting. In this study, for the first-stage processing, we developed an edge filter that utilizes the characteristics of white-line contours and detects noise correctly. This filter uses brightness-gradient approximation by discrete values, for which we obtained an idea for a non-linear filter that approximates a low-pass filter plus differential calculus. By applying the method to the images taken by on-board camera, we demonstrate that white-line detection that can apply to diverse road environments but is hardly affected by noise can be realized through combination with model-less clustering.

Key words: Edge filter, White line, Detection, Model-less, Common road, Brightness gradient

1. Introduction

In recent years, advanced driver assistance systems that use information from on-board sensors have spread remarkably. Application of this system to diverse road environments including common roads is expected and, in conjunction, further evolution of recognition technologies is required.

Lane recognition using a camera is widely utilized as one of these technologies. From the beginning of the 1990s, many road white-line detection methods for use in standardized roads such as highways have been proposed. Dickmanns and Mysliwetz (1992) proposed a method that uses road models consisting of three-dimensional spaces that contain vertical curvature. They then formulated white-line detection as an issue of parameter estimation for the model. Pomerleau and Jochem (1996) and Kreucher et al. (1998) approximated the shapes of white lines using a parabolic curve model in a two-dimensional plane. Estimation that will be relatively unaffected by noise was achieved for standardized roads with few model parameters. However, the accuracy of approximation decreases depending on the shape of the white line. Ma et al. (2000) proposed using circles in a two-dimensional plane. Like Kreucher, the range of approximation by the model is narrow and the problems of low accuracy approximation remain. Lee (2002) proposed approximating the shape with a linear model or a polygonal line model and estimating parameters with Hough transform. Although this method requires no assumption of the number of white lines to detect, detection accuracy decreases for curved white lines that are difficult to approximate with straight lines. It can also apply the line-segment detector proposed by Gioi et al. (2010) to detect white lines. However, this method also assumes straight lines, so the same problem remains.

As a different approach, machine learning represented by deep-learning can be applied (Misono et al., 2016). This
approach shows excellent performance in many fields. Even so, it is thought to be worth considering the principle used in each field because it will explain the soundness of learning results outside evaluation for test data set. This paper’s interest lies in finding the principle for white line detection as well as making detection algorithm. So this paper tried non-learning based approach.

Application to diverse road environments including common roads requires a method that detects white lines without using models. The use of border following (Szeliski, 2011) can be considered as an example. Border following is a technique that clusters the pixel group composing the contour line (edge groups) depending on their mutual proximity. However, this clustering detects the contour lines formed by the texture also. Figure 1 shows an example of applying clustering to the edge detection results by the Gabor filter (Petkov and Wieling 2017). It can be seen from the figure that the noise from the road surface texture may be detected erroneously depending on the threshold of the edge judgment. There is a tradeoff between detecting white lines that have small differences in brightness from the road surface and reducing noise from the texture.

![Fig. 1 Problems in the Conventional Method - Erroneous detection of road-surface texture. Left figure shows the original image and right figure edge image. Edge detection is applied for the area shown by dashed line drawn in the left figure.](image)

This paper proposes an edge filter that can detect the edges of white-line contours that have small brightness differences from the road surface and can reduce erroneous detection of edge noise from the texture. Such a filter judges edges using a new evaluation index based on the distribution of brightness instead of a continuous brightness gradient. This evaluation index is calculated using an idea from non-linear differential calculus to block high-frequency components and can be considered as brightness gradient approximation by discrete values. We demonstrate that this edge filter can detect white lines with reduced errors in detecting noise that can correspond to complex road environments in combination with clustering that does not use models.

2. Edge filters

2.1 Conventional edge filters

This section explains conventional methods to facilitate understanding of the proposed edge filter. The same processing is repeated for all pixels composing the image, so the following will explain the processing for one pixel of interest. Conventionally, many types of edge filters (e.g. the Sobel filter and the Gabor filter) have been proposed. These filters judge edges by calculating the brightness gradient from the brightness near the pixel of interest and comparing it with the threshold. Equation (1) is the edge judgment formula using the Sobel filter. The Gabor filter also adopts the same configuration.

\[
E(x, y) = \sum_{i, j = -1}^{1} w_{ij} I(x+i, y+j) \geq Th_{BRIGHT}
\]

where

\[
W = \begin{bmatrix}
w_{-1,-1} & w_{0,-1} & w_{1,-1} \\
w_{-1,0} & w_{0,0} & w_{1,0} \\
w_{-1,1} & w_{0,1} & w_{1,1}
\end{bmatrix} = \begin{bmatrix}
-0.25 & 0 & 0.25 \\
-0.5 & 0 & 0.5 \\
-0.25 & 0 & 0.25
\end{bmatrix}
\]
In Equation (1), \((x, y)\) represents the coordinates of the pixel of interest, and \(E(x, y)\), the judgment result of that pixel; 1 indicates an edge and 0, no edge. The left side of the inequality sign is an approximation of the brightness gradient in the x direction at pixel \((x, y)\), and the right side is the threshold for edge judgment. \(W\) is the weighting matrix. Thus, the brightness gradient of the pixel of interest will be calculated from the brightness distribution near that pixel, which can reduce the influence of noise. However, if there is a pixel that has an extremely large brightness difference while \(Th_{BRIGHT}\) is set to a low value, the area surrounding such pixel will be judged as an edge regardless of the reality. Figure 1 depicts an example of clustering based on this erroneous judgment.

2.2 Basic concept of the proposed edge filter

The brightness gradient can be considered as the differential of the brightness in the image plane. In Section 2.1, the edge was judged by integrating the differential of brightness within the appointed region to remove noise accompanying the differentiation. Blocking high-frequency components without following any sudden brightness change is also an effective method of removing noise. This can be approximated by limiting the difference output. Ultimately, it results in a process that outputs 1 for any brightness change exceeding the threshold and outputs 0 for other values of brightness. We believe that an edge evaluation index that is hardly influenced by noise can be obtained by combining this idea with integral calculus.

The prerequisite for realizing this new index is that the white-line contour is composed of a smooth curve. This is depicted in Fig. 2. A smooth curve can be approximated by straight lines, so the contour line in the figure is expressed by a straight line. This contour line is the border of the high-brightness region and low-brightness region. Pixels of the high-brightness region (white line) are expressed as white grids; pixels of the low-brightness region (e.g., asphalt), as gray grids; and the contour line, as light-gray grids. The pixel of interest for edge judgment is shown at the center of the figure. In this case, multiple pixel pairs are located symmetrically in the orthogonal direction across the contour line and are expected to meet the following condition.

\[
I(x_{wi}, y_{wi}) - I(x_{bi}, y_{bi}) > Th_{bRIGHT}
\]  

(2)

Here, suffix \(i\) represents the number of pixel pairs and indicate the coordinates of the pixel on the high-brightness side \((x_{wi}, y_{wi})\), and those on the low-brightness side \((x_{bi}, y_{bi})\). In Fig. 2, each pair of pixels is expressed by two circles and a curve connecting them.

![Fig. 2 Pixel pairs around the pixel of interest.](image)

Multiple pixel pairs lie near the pixel of interest and along the contour line. Actually, the contour line is not a straight line but a smooth curve, and the number of pixel pairs decreases as the radius of curvature decreases. In other words, performing edge judgment by setting the threshold for the number of pixel pairs enables setting the lower limit of the radius of curvature of detectable curves. Furthermore, the conventional problem can also be solved by introducing this judgment method. Any pixel with an extremely high brightness compared to its surroundings will influence the number of pixel pairs, although the size of such influence will not change no matter how large the difference in brightness. Based on these merits, we decided to use the number of pixel pairs along the contour line as a new index for edge judgment. Equation (2) corresponds to the limit of difference output; the number of pixel pairs...
corresponds to the integral calculus.

To detect edges using this index, it is necessary to clarify two points. The first is the relationship between the above-mentioned number of pixel pairs and the radius of curvature. Clarifying this relationship enables freely setting the lower limit of the radius of curvature of detectable white-line contours. The second point is the realizing method. It is impossible to know the straight line composed by the white line contour in Fig. 2 beforehand, so edges are detected by assuming individual straight lines with multiple edge filters called the filter bank. Although a limited number of filters will be prepared, in practice it is necessary to configure the filters by considering the approximation caused by it. The filter configuration and the method of setting parameters such as the threshold are explained below.

2.3 Configuration of edge filters

Brightness gradient vectors of edges composing the white line counter are distributed in all directions in the image, so it is necessary to prepare multiple filters called the filter bank. Here, all direction of the image 0 to 2π[rad] will be quantized by the predetermined number $N$. The $d^{th}$ direction will be defined by Eq. (3).

$$\theta_d = \frac{2\pi}{N} (d - 1)$$

Here, let’s assume that the rightward horizontal direction in the image is the plus direction on the $x$ axis; the upward vertical direction is the plus direction on the $y$ axis; the plus direction on the $x$ axis is 0[rad] and, when the direction goes from the $x$ axis to the $y$ axis, it goes toward 2π[rad].

To detect edges regardless of the brightness gradient direction, the $d^{th}$ edge filter must detect edges that have a brightness gradient in the direction with a width of $2\pi/N$ around $\theta_d$. The brightness of the two regions adjacent to that pixel, $RW_d$ and $RB_d$, will be used to judge edges in the pixel of interest. This is shown in Fig. 3.

The two regions compose an oval, and one axis of the oval constitutes the boundary of the two regions. Let’s express the $i^{th}$ pixels belonging to region $RW_d$ ($RB_d$) as $I(x+xbd_i, y+ybd_i)$ ($I(x+xbd_i, y+ybd_i)$) where two pixels are symmetric about the boundary line. We propose an edge filter that utilizes the brightness of $RW_d$ and $RB_d$ in Eq. (4).

$$E_d(x,y) = \{ n(Gr_d) > Th_{\text{NUMBER}} \}$$

where

$$Gr_d = \{ f(x+xd_i, y+yd_i) - I(x+xbd_i, y+ybd_i) > Th_{\text{BRIGHT}} \}$$

Here, $(x,y)$ represents the coordinates of the pixel of interest. $E_d(x,y)$ represents the edge judgment result; 1 indicates an edge and 0 indicates no edge. $Gr_d$ represents the cluster of those pixel pairs of brightness difference exceeding the threshold $Th_{\text{BRIGHT}}$ among pixel pairs defined around the pixel of interest. $n(Gr_d)$ represents the number
of elements (number of pixel pairs) of this cluster. When this number exceeds the threshold $\text{Th}_{\text{NUMBER}}$, the pixel of interest will be judged as an edge. As indicated by suffix $d$ in Eq. (4), this is the $d^{th}$ filter, and a total of $N$ filters with $d$ varied from 1 to $N$ are prepared.

2.4 Setting of parameters

The parameters to be set are $\text{Th}_{\text{BRIGHT}}$, $\text{Th}_{\text{NUMBER}}$ and $\gamma$, $r$ to specify the oval shown in Fig. 4. $\text{Th}_{\text{BRIGHT}}$ shall be set below the lower limit of the assumed brightness difference between the white line and road surface. In the following, the brightness is defined from 0 to 255 and $\text{Th}_{\text{BRIGHT}}$ is fixed to 20 except for the case of severe illumination condition. $\text{Th}_{\text{NUMBER}}$ and $\gamma$, $r$ relate to the composition of the filter bank and the lower limit of the detectable radius of curvature. First, we show how to set the threshold $\text{Th}_{\text{NUMBER}}$ from the requirements for composing the filter bank. Next, we show how to set $\gamma$, $r$ from the lower limit of the radius of curvature using $d = 1$. The oval region of the edge filter and the white line contour are depicted in Fig. 4. White grids represent pixels with high brightness that were detected, and gray grids, pixels with low brightness. W (B) represents pixels with high (low) brightness determined by the $d = 1$ edge filter. Here, let’s consider the conditions for the white line contour with brightness gradient vector $\Delta \theta$ to be detected by the $d=1$ edge filter.

![Fig. 4 Edge filter ($d = 1$) applied to contour ($\theta = \Delta \theta$).](image)

The edge filter output of the pixel of interest is expressed by Eq. (5).

$$ E_d(x, y) = \pi \cdot r^2 \cdot \frac{\pi - 2 \cdot \tan^{-1}(\gamma \cdot \tan \Delta \theta)}{2 \cdot \gamma} $$

(5)

Here, $r$ represents the radius in the $y$ direction of the oval composed of regions $RW_d$ and $RB_d$ while $\gamma$ represents the radius in the $y$ direction/radius in the $x$ direction. The edge filter output means the number of pixels contained in region $RP_d$ shown in Fig. 5. We should set threshold $\text{Th}_{\text{NUMBER}}$ as follows to judge this output to be an edge.

$$ \text{Th}_{\text{NUMBER}} \leq \frac{\pi \cdot r^2 \cdot \pi - 2 \cdot \tan^{-1}(\gamma \cdot \tan \Delta \theta)}{2 \cdot \gamma} $$

(6)

Furthermore, to detect the white line contour with brightness gradient vector $\Delta \theta$ by the $d = 1$ edge filter, $\Delta \theta$ must satisfy Eq. (7).

$$ 2 \cdot \Delta \theta \leq \frac{2 \cdot \pi}{N} \therefore \Delta \theta \leq \frac{\pi}{N} $$

(7)
Equation (8) is obtained by substituting Eq. (7) for Eq. (6).

\[
Th_{\text{NUMBER}} \leq \frac{\pi \cdot r^2}{2 \cdot \gamma} \cdot \frac{\pi - 2 \cdot \tan^{-1}(\gamma \cdot \tan(\frac{\pi}{N}))}{\pi}
\]  

(8)

Threshold \(Th_{\text{NUMBER}}\) was introduced using \(d = 1\), but it is a relational expression that can be established universally. As shown in Eq. (8), it is necessary to increase the number of partitions \(N\) to set the threshold to a high value. Although this method is convenient for removing noise, the operational load for edge detection becomes high, so it must be set considering this tradeoff. Threshold \(Th_{\text{NUMBER}}\) is set as explained above. Next, we explain setting parameters \(r\) and \(\gamma\) to specify the regions of the oval, \(RW_d\) and \(RB_d\).

Let’s consider detecting the curve with a radius of curvature of \(R\) in the image. Assume that the distance between adjacent pixels is 1. Edge filter \(d = 1\) is applied by assuming the left end of the circle with radius of curvature \(R\) is the pixel of interest, as depicted in Fig. 6.

Here, if \(R\) is large, the following relations are established between \(\alpha\), \(r\), and \(R\) in Fig. 6.

\[
R = \frac{\cos(\tan^{-1}(\gamma \cdot \tan(\alpha)))}{\sin(2\alpha)} \cdot r
\]  

(9)

If \(\alpha\) is assumed to be \(\pi/N\), the edge filter output agrees with the right side of Eq. (8) as can be understood from Figs. 5 and 6 and Eqs. (5) and (8). Equation (10) is produced by substituting \(\alpha\) of Eq. (9) with \(\pi/N\). The threshold of Eq. (8)
will set the lower limit of the radius of curvature of detectable contour lines to $R$.

$$R = \frac{\cos(\tan^{-1}(\gamma \cdot \tan(\frac{\pi}{N})))}{\sin(\frac{2\pi}{N})} r$$  \hspace{1cm} (10)$$

To detect only smooth contour lines with large $R$, it is necessary to make $r$ larger as expressed in Eq. (10). If parameter $\gamma$ is set to a small value, edges can be judged based on a wide region. While it is scarcely affected by noise, it may become impossible to detect narrower white lines. It is necessary to set both parameters considering the tradeoff between them. Methods for setting $r$ and $\gamma$ are explained above.

3. Evaluating the edge filter

This section shows that the parameter setting explained in Section 2 is useful. First, we will demonstrate that all edges can be detected regardless of the brightness gradient vector. We will then show that the lower limit of the radius of curvature of detectable curves has been realized.

3.1 Threshold $Th\text{NUMBER}$ and omni-directional detectability

Here we explain the result of applying the edge filter to the test image. Figure 7 shows the test image, where the brightness on the high-brightness side (light gray) is set to $I_w = 141$, and that on the low-brightness side (dark gray) is set to $I_b = 120$. The brightness is represented by the value from 0 to 255. If the value is large, the brightness is high. The boundary of the high-brightness side and low-brightness side is a circle composed of the edge to be detected. Furthermore, the brightness gradient vector of this edge is distributed from $0$[rad] to $2\pi$[rad]. The radius of the circle is $180$[pixel].

![Fig. 7 Test image.](image)

Next, we explain the method for setting the edge filter. The number of directional partitions $N$ was set to 8. For regions $RW_{d}$ and $RB_{d}$, $r$ was set to 3.5 [pixels] and $\gamma$, to .5[-]. Regions $RW_{d}$ and $RB_{d}$ for $d = 1$ (which corresponds to a brightness gradient of $0$ [rad]) are shown in Fig. 8. $RW_{d}$ ($RB_{d}$) can be approximated by the clusters of pixels represented by $W$ ($B$) in Fig. 8.

![Fig. 8 Example of $RW_{d}$ and $RB_{d}$ for edge filter ($d = 1$).](image)
Threshold $Th_{BRIGHT}$ was set to 20, which is less than the brightness difference of 21. Furthermore, from Equation (11), it is assumed that the edges on the whole circumference can be detected by setting threshold $Th_{NUMBER}$ to 8.

$$Th_{NUMBER} \leq \frac{\pi \cdot \gamma^2}{2 \cdot \gamma} \cdot \pi - 2 \cdot \tan^{-1}(\gamma \cdot \tan \left( \frac{\pi}{N} \right)) \geq 8.3$$

(11)

The results of setting $Th_{NUMBER}$ to 9 and 8 are depicted in Fig. 9.

![Fig. 9 Edge detection by proposed edge filter. Left image shows the result when $Th_{NUMBER} = 9$ and Right Image $Th_{NUMBER} = 8$. These results show that $Th_{NUMBER}$ should be below 9.](image)

$Th_{NUMBER}$ should be set to 8 for detecting the contour of the circle completely. Each edge filter can detect all edges specified by directional partition using the threshold $Th_{NUMBER}$ expressed by Eq. (11). The number of pixel pairs shown in Fig. 8 is 10 and is larger than $Th_{NUMBER}$. If this filter is applied to the edge pixels on the line whose gradient direction precisely equals to 0 [rad], the output becomes 10. There is 2 point margin for the detection. This means that this filter may detect edge pixels near the corner of the line. This consequence holds for all $\theta_d$.

### 3.2 Relationship between ellipsoidal parameter $\gamma$ and the performance of edge detection

It is expected that if the oval area becomes large, the performance of edge detection is improved. In another word, if the parameter $\gamma$ decreases, the performance is improved under the condition that the parameter $r$ is fixed. This relation is shown in the following explanation. The parameter $Th_{NUMBER}$ also changes according to $\gamma$ by Eq. (8). Parameters used in the simulation are shown in Table 1. Number of pixel pairs means the size of oval of edge filter.

<table>
<thead>
<tr>
<th>$\gamma$</th>
<th>Number of pixel pairs</th>
<th>$Th_{NUMBER}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>2.3</td>
<td>7</td>
<td>4</td>
</tr>
<tr>
<td>1.5</td>
<td>10</td>
<td>8</td>
</tr>
<tr>
<td>1.0</td>
<td>19</td>
<td>14</td>
</tr>
</tbody>
</table>

The test image was produced by adding noise to the image in Fig. 7 as shown in Fig. 10. The brightness on the high-brightness side (light gray) and that on the low-brightness side (dark gray) were set to $I_a = 141$ and $I_b = 120$, respectively. 5[%] of the pixels were extracted at random from the whole screen, and their brightness was set to 240.

![Fig. 10 Test image.](image)
Output results of edge filter are shown in Fig. 11. It is observed that the performance of edge detection is improved according to the decrease of parameter $\gamma$.

![Fig. 11 Edge detection for circle contour. From left to the right, $\gamma = 2.3, 1.5, 1.0$.](image)

Performance index is shown in Table 2. Total performance index F-measure is improved according to the decrease of parameter $\gamma$. From the performance view, it is recommended to use small $\gamma$. But it is also noted that the small $\gamma$ invites the increases of processing time and the deterioration of narrower white line detection.

<table>
<thead>
<tr>
<th>$\gamma$ [-]</th>
<th>Recall [-]</th>
<th>Precision [-]</th>
<th>F-measure [-]</th>
</tr>
</thead>
<tbody>
<tr>
<td>2.3</td>
<td>0.975</td>
<td>0.768</td>
<td>0.859</td>
</tr>
<tr>
<td>1.5</td>
<td>0.921</td>
<td>0.959</td>
<td>0.940</td>
</tr>
<tr>
<td>1.0</td>
<td>0.949</td>
<td>0.976</td>
<td>0.962</td>
</tr>
</tbody>
</table>

### 3.3 Comparison with conventional filters

We demonstrate that the proposed edge filter have better performance compared to conventional algorithm. The test image was same as in 3.2. As conventional filter, 2 algorithms are selected. The first one was Gabor filter (Petkov and Wieling 2017). To align conditions, with the threshold of brightness $Th_{BRIGHT}$ was set to 20 and the weighting matrix of the Gabor filter was adjusted so that the reference regions of brightness are roughly aligned. The Gabor filter for $d = 1$ is expressed by Eqs. (12) and (13). Filters other than $d = 1$ were produced by turning the following weighting matrix $W$ by $\pi/4$.

$$EG_i(x, y) = \sum_{i,j=3}^{3} w_{ij} \cdot I(x+i, y+j) \geq Th_{BRIGHT}$$

$$
\begin{bmatrix}
  w_{-2,3} & w_{-1,3} & w_{0,0} & w_{1,3} & w_{2,3} \\
  w_{-2,2} & w_{-1,2} & w_{0,1} & w_{1,2} & w_{2,2} \\
  w_{-2,1} & w_{-1,1} & w_{0,1} & w_{1,1} & w_{2,1} \\
  w_{-2,0} & w_{-1,0} & w_{0,0} & w_{1,0} & w_{2,0} \\
  w_{-2,-1} & w_{-1,-1} & w_{0,-1} & w_{1,-1} & w_{2,-1} \\
  w_{-2,-2} & w_{-1,-2} & w_{0,-2} & w_{1,-2} & w_{2,-2} \\
  w_{-2,-3} & w_{-1,-3} & w_{0,-3} & w_{1,-3} & w_{2,-3}
\end{bmatrix}
\begin{bmatrix}
0.0006 \\
-0.0031 \\
0.0031 \\
-0.0006 \\
0.0097 \\
-0.0533 \\
0.0533 \\
-0.0097 \\
0.0533 \\
-0.2942 \\
0.2942 \\
-0.0533 \\
0.0942 \\
-0.5200 \\
0.5200 \\
-0.0942 \\
0.0533 \\
-0.2942 \\
0.2942 \\
-0.0533 \\
0.0097 \\
-0.0533 \\
0.0533 \\
-0.0097 \\
0.0006 \\
-0.0031 \\
0.0031 \\
-0.0006
\end{bmatrix}$$

And the second one is BEL algorithm (Dollar, 2006) trained on the Berkeley training images. BEL got high score on Berkeley data base and is quoted by many papers. Threshold of brightness $Th_{BRIGHT}$ was also set to 20. Edge detection results are shown in Fig. 12. The Gabor filter and BEL detects much noise in addition to the contour line of the circle. The proposed method detects only those edges that compose the contour line of the circle, demonstrating that the proposed method is superior to the conventional methods in detecting edges with small brightness differences while scarcely detecting noise from the texture.
4. Application to White Line Detection

4.1 White-line detecting algorithm

The white-line detecting algorithm prepared by combining the proposed edge detection and clustering is shown in Fig. 13. Directions defined in the image plane will be divided into $N$ partitions, and direction list $ListDirection = \{1:N\}$ will be defined. An edge will be detected in each direction of $d \in ListDirection$ and then clustered. $E_d(OUT_d)$ is the $d$th edge filter output and cluster list (list of contour lines). Small components of the cluster will be removed in the clustering process. All of them will then be integrated, and cluster list $OUT$ will be generated. When there are many white line contours, the corresponding number of clusters will be generated. Furthermore, continuous white line contour portions will be generated to compose broken white lines. Step 3 is clustering, which utilizes the proximity of edges. The edge filter, the core of this paper, is Step 2.

Algorithm: Road White-Line Detector

<table>
<thead>
<tr>
<th>Algorithm: Road White-Line Detector</th>
</tr>
</thead>
<tbody>
<tr>
<td>input: image $I$</td>
</tr>
<tr>
<td>output: cluster list $OUT$</td>
</tr>
<tr>
<td>1 for each direction $d \in ListDirection$ do</td>
</tr>
<tr>
<td>2 $E_d = EDGE_FILTER(I, d)$</td>
</tr>
<tr>
<td>3 $OUT_d = CLUSTERING(Ed)$</td>
</tr>
<tr>
<td>4 Add $OUT_d \rightarrow OUT$</td>
</tr>
<tr>
<td>5 end</td>
</tr>
</tbody>
</table>

Fig. 13 Algorithm of road white-line detection.

4.2 Application to onboard camera images

Five example of application to actual images are shown below. In each figure, the original image, the result of clustering in combination with the Gabor filter, the result of BEL and the result of clustering in combination with proposed edge filter are shown in this order from the left. Clusters for which the number of edges is less than the threshold ($=5$) will be removed as noise. While the conventional methods detect the texture of road-surface asphalt, the proposed method scarcely detects it in each case. Although some road white lines shown in Fig. 16 have low brightness, the proposed method can detect original white line contours by removing noise. Furthermore, as shown in Fig. 18, the proposed method can cluster white line contours with arbitrary curve shapes.

Fig. 14 Scene1 (Roundabout): From the left to the right, original image, white line detection result by conventional method (Gabor filter + clustering), BEL, and proposed method
Almost all the same “Recall” performance are observed for conventional and proposed method; nevertheless the proposed method shows far superior performance for “Precision”. As the result, proposed method shows superior performance for total harmonized index “F value”.

Thus, we showed that the proposed edge filter can detect edges of white line contours with low brightness differences from the road surface and rarely makes errors detecting noise from the texture at the same time and that it can achieve white-line detection corresponding to complex road environments including common roads through combination with model-less clustering.

Table 3 shows the comparison of evaluation results for each scene. Almost all the same “Recall” performance are observed for conventional and proposed method; nevertheless the proposed method shows far superior performance for “Precision”. As the result, proposed method shows superior performance for total harmonized index “F value”.

Thus, we showed that the proposed edge filter can detect edges of white line contours with low brightness differences from the road surface and rarely makes errors detecting noise from the texture at the same time and that it can achieve white-line detection corresponding to complex road environments including common roads through combination with model-less clustering.

Table 3 Comparison of evaluation results.

<table>
<thead>
<tr>
<th>Scene</th>
<th>Recall [-]</th>
<th>Precision [-]</th>
<th>F-measure [-]</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Gabor+ Clustering</td>
<td>BEL</td>
<td>Proposed</td>
</tr>
<tr>
<td>Roundabout</td>
<td>0.986</td>
<td>0.995</td>
<td>1.000</td>
</tr>
<tr>
<td>Toll gate1</td>
<td>0.981</td>
<td>0.889</td>
<td>0.991</td>
</tr>
<tr>
<td>Toll gate2</td>
<td>0.954</td>
<td>0.910</td>
<td>0.956</td>
</tr>
<tr>
<td>Intersection</td>
<td>0.985</td>
<td>0.934</td>
<td>0.997</td>
</tr>
<tr>
<td>Fork road</td>
<td>0.966</td>
<td>0.943</td>
<td>0.939</td>
</tr>
</tbody>
</table>

Table 4 shows the processing time for proposed algorithm. The algorithm repeats calculation for 8 gradient
directions. So the processing time can be 1/8 using parallel processing. More precisely, edge filter, one part of algorithm, can be 1/12 because roughly 2/3 can be done by parallel processing. Even so, time for clustering is so huge. The reason is thought to be “if” sentences used in clustering. Additional improvement is required for real time processing.

<table>
<thead>
<tr>
<th>Scene</th>
<th>Image Size [pixel × pixel]</th>
<th>Processing time [s]</th>
</tr>
</thead>
<tbody>
<tr>
<td>Roundabout</td>
<td>1280×850</td>
<td>1.408</td>
</tr>
<tr>
<td>Toll gate1</td>
<td>800×542</td>
<td>0.632</td>
</tr>
<tr>
<td>Toll gate2</td>
<td>800×600</td>
<td>0.690</td>
</tr>
<tr>
<td>Intersection</td>
<td>700×525</td>
<td>0.539</td>
</tr>
<tr>
<td>Fork road</td>
<td>640×427</td>
<td>0.446</td>
</tr>
</tbody>
</table>

4.3 Application to severe illumination condition

In fine weather condition, the brightness difference between white line and road surface is expected to be larger than 20. But in severe illumination condition such as the case against the sun light, the difference becomes much smaller. The parameter $T_{BRIGHT}$ should be set smaller than 20 for such case. In the following, the detection results are explained for the test image shown in Fig. 19.

Fig. 19 Test image of severe illumination condition. Edge detection is applied for the area shown by dashed line.

Fig. 20 is the result using $T_{BRIGHT} = 20$. The brightness difference is much smaller than 20, Gabor filter and proposed method can hardly detect the white line. BEL algorithm shows better detection performance.

Fig. 20 Detection results for $T_{BRIGHT} = 20$. From the left to the right, Gabor, BEL and Proposed method.

Fig. 21 is the result using $T_{BRIGHT} = 5$. Gabor filter and proposed method detect the white line without detecting much noise on road surface. On the other hand, BEL algorithm detects much noise for that.

Fig. 21 Detection results for $T_{BRIGHT} = 5$. From the left to the right, Gabor, BEL and Proposed method.
Table 5 shows the performance for Fig. 20 and Fig. 21. From the view of F-measure index, proposed method is a little superior to the other method. But the value shows that the improvement is required for severe illumination condition.

<table>
<thead>
<tr>
<th>Threshold</th>
<th>Recall [%]</th>
<th>Precision [%]</th>
<th>F-measure [%]</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Gabor+ Clustering</td>
<td>BEL</td>
<td>Proposed</td>
</tr>
<tr>
<td>20</td>
<td>0.007</td>
<td>0.694</td>
<td>0.026</td>
</tr>
<tr>
<td>5</td>
<td>0.573</td>
<td>0.983</td>
<td>0.955</td>
</tr>
</tbody>
</table>

5. Conclusions

We proposed an edge filter based on a new evaluation index instead of judging the threshold of brightness change. This evaluation index, provided by an idea of non-linear filter that approximates a low-pass filter plus differential calculus operation, approximates the brightness gradient with discrete values. We realized a white-line detection method that can successfully apply to complex road environments including common roads by combining this edge filter with clustering that does not use models. High-speed processing of edge filter by parallel computers can be expected. We presented the result of off-line processing in this study, but it is necessary to realize real-time processing for edge clustering. Furthermore, we will continuously improve the performance under severe illumination condition and the method to cope with objects such as shades in combination with understanding of road environments.

References


