International Journal of Hybrid Information Technology
Vol.6, No.5 (2013), pp.103-116
http://dx.doi.org/10.14257/ijhit.2013.6.5.10

Human-like Driving for Autonomous Vehicles using Vision-based Road Curvature Modeling[**]

Wei Li[*]

School of Electronics and Information, Nantong University, Jiangsu 226019, P. R. China

lw@ntu.edu.cn

Abstract

Most autonomous vehicles use GPS to determine vehicle location and heading. Using GPS for vehicle autonomous driving posts a few challenges. It does not look far ahead of the vehicle and requires frequent adjustment of vehicle heading. This results in an unstable control system and increases the chance of unstable driving behavior. Unlike this kind of passive or reactive control system, human drivers look far ahead of the road to determine the road curvature and actively adjust vehicle heading and turning angle in advance. Visual information is essential to enabling this kind of human-like driving behavior for autonomous vehicles and should be an important supplement to the GPS-based system, especially in GPS-denied environments. Road lanes are often curved, making vision-based detection of smooth and continuous curves in front of the vehicle a challenging task. Furthermore, commonly used computer vision algorithms such as edge detectors or Hough transform for line or curvature detection are not robust in changing lighting conditions. This paper presents a vision algorithm designed specifically for detecting and modeling road curvature for human-like active steering control and heading adjustment for autonomous vehicles. The proposed algorithm has been tested for paved and unpaved road conditions and shown very good results.

Keywords: *Lane Detection, Gabor Wavelet Filters, q-Bernstein Polynomials, Bezier Curve, Intelligent Vehicle*

1. Introduction

The Defense Advanced Research Project Agency (DARPA) sponsored grand and urban challenges held in recent years have sparked research interests in autonomous vehicles. Researchers have looked into the development of this technology in all aspects and have seen some successes in commercial and military applications. Many focus on fusing GPS waypoints and LIDAR readings to determine vehicle location and moving path. Vehicle steering and heading are adjusted constantly in order to maintain the vehicle on the calculated path, which results in an unstable control system. Human driving, on the contrary, steers

[**] Project supported by the Nature Science Foundation of Nantong University, Jiangsu, China (No. 11Z060)

[*] Corresponding Author

rather smoothly even at high speeds and on a curved road. This is because human drivers see the entire road curvature in sight and determine heading and steering angle in advance. In order to mimic this type of smooth human driving behavior for autonomous vehicles, road and no-road regions must be distinguished and road curvature must be determined.

Vision-based road detection has been an active research topic in recent years and different methods have been presented to solve this problem [1-6]. In principle, vision-based lane detection methods can be categorized into three main classes: feature-based, region-based, and model-based.

Feature-based methods rely on the features of the road as salient information. First, road features such as lane markings and road texture are extracted. These extracted features are then used to construct a road map [4, 5]. However, features used in these methods should not be easily interfered by features of other irrelevant objects. For example, white and yellow color features are used to represent road lanes [4]. This method is novel but results become unreliable when objects contain the same color features, *e.g.*, white or yellow vehicles moving in front of the camera.

Region-based methods can be viewed as segmentation of color or gray scale image. Image is segmented into different regions, such as none-road and road regions [7, 8]. Authors paired an optical flow algorithm with one-dimensional template matching and used sum of squared differences (SSD) to determine the most similar regions to the template [8]. This method can only be applied to roadways that are wide open since an obstacle or vehicle in front of the vehicle affects SSD accuracy.

Model-based methods generally use a geometric model to represent road edges or lanes. Geometric models for straight line and curve line are used to represent road curvature [6, 9]. A transform or filters are used to extract control points or lines needed by the geometric model to determine the model parameters. The main challenge for model-based methods is to extract necessary parameters to construct the model. Other methods for road lane detection such as [10] are combinations of the above three main methods.

Two major challenges for automatic road and lane detection are objects obstructing camera view and variations in lighting conditions. These include tree or building shadow, other vehicles on the road, and pedestrians, etc. A robust road and lane detection algorithm is needed to address these two major challenges. Y. Wang [6] used a robust algorithm called Canny/Hough Estimation of Vanishing Points (CHEVP) to determine the road edge and vanishing points. It is not reliable because the robustness of CHEVP cannot be fully controlled or adjusted. Unpredicted result occurs when tree or electricity pole casts a shadow on the road that creates an edge segment that has erroneous edge orientation.

Multi-scale Gabor filter is proposed for detecting road edge [11, 12]. The robustness and sensitivity of multi-scale Gabor filter can be adjusted to detect road edge and its orientation. Interference from shadow is usually not present in the edge detection result when using multi-scale Gabor filter because shadow usually does not have prominent edges. Although objects such as tree trunk and electricity pole have prominent edges, the possibility of their edge orientations being similar to the real road edge is very low. In rare cases that the edge orientation of shadow or other objects is the same as or close to the rode edge, the false edges can be removed based on the detection result from previous frames.

Research work presented in this paper focuses on modeling road curvature to enable human-like driving for autonomous vehicles. The algorithm presented in this paper is illustrated in Figure 1. The first step is to apply multi-scale Gabor filter to small image regions (blue boxes in Figure 1 (b)) on the bottom of the image (close to the vehicle) to locate seed pixels that represent road lines (yellow or white) or road edges. Once these seed pixels in the image are detected, the orientation of the detected edges can be calculated. In some cases such as dashed line or faded line that the seed pixels cannot be detected in the small regions, regions immediate above (green boxes in Figure 1(b)) are used to search the seed pixels. The edge orientation of the seed pixel is used to determine the road edges or lines. A growing and tracking algorithm is then used to detect edge segments. After all potential edge segments are found, an estimate function is used to assign a reliability value for each detected edge segment. This reliability value is used to determine if the detected edge is an actual road edge or from other irrelevant objects, shadow or pedestrians in front of the vehicle. After all edges on both sides are detected and verified, q-Bezier curve [6, 13, 14] is used to model the curve of one of the lines or road edges and perspective geometry information is used to model the other line or edge (Figure 1(d)).

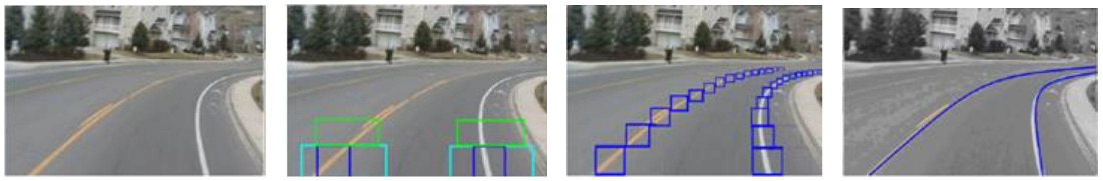

(a) Original image (b) Seeds of potential road lines (c) Result of growing and tracking (d) Curve fitting

Figure 1. Illustration of the proposed algorithm

2. Algorithm

Our road curvature modeling algorithm consists of three steps: seed pixel search using multi-scale Gabor filter, edge growing and tracking, curvature modeling. Details of these steps and related computations are described in this section.

Due to camera perspective effect, the width of the road or lane on the bottom of the image (close to the camera) is different from that on the top of the image (further away from the camera). Multi-scale Gabor filter is needed because different scales of Gabor filter will determine different sensitivities in the same width of edge.

Seed searching uses selected multi-scale Gabor filters to search the seed of every potential edge on each side of the road from the bottom region of the image. The seed actually represents the potential edge on both side of the road. Once the seed is located, seed growing and tracking algorithm is activated to detect the entire road edge according to the location and direction of the seed.

Lastly, after all potential edges being detected, an estimation function is utilized to evaluate the reliability of the potential edges. This reliability measure is used to determine if the edges detected are the actual road edges or the contour of moving vehicles or shadows. The measure in the growing and tracking algorithm method is a major contribution of this algorithm. The detail structure of these components is illustrated in Figure 2.

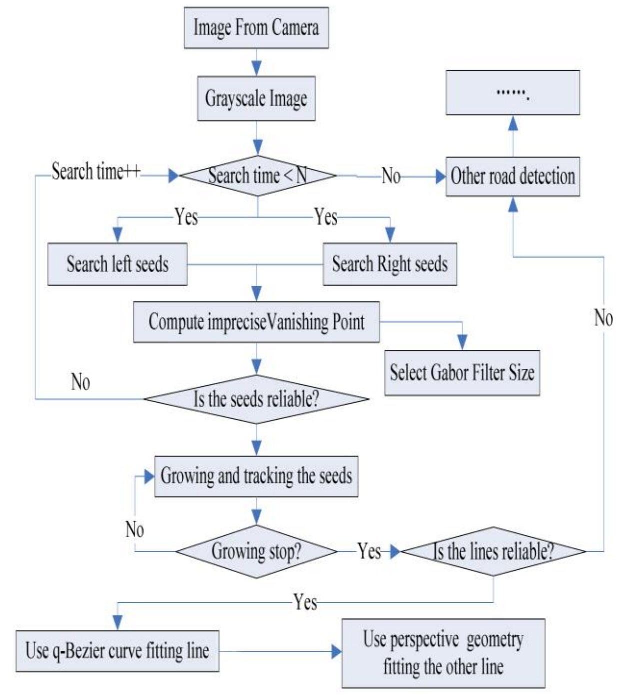


Figure 2. Algorithm flowchart

3. Road edge detection

3.1 Multi-scales Gabor wavelet

In this paper, multi-scale Gabor filters are used to detect road edge. The Gabor wavelet kernels used in this paper are calculated as follows:

$$G_{im}(x,y,\theta,\lambda,\sigma)=e^{-\frac{1}{8\sigma^2}(4a^2+b^2)}\cos(\frac{2\pi a}{\lambda}) \tag{1}$$

$$G_{real}(x,y,\theta,\lambda,\sigma)=e^{-\frac{1}{8\sigma^2}(4a^2+b^2)}\sin(\frac{2\pi a}{\lambda}) \tag{2}$$

$$a=x\cos\theta+y\sin\theta \tag{3}$$

$$b=-x\sin\theta+y\cos\theta \tag{4}$$

The column and row of the Gabor filter are represented by x and y. The kernel center locates at (0, 0). θ is the wavelet orientation and λ is wavelength. σ is the standard deviation of Gaussian kernel. The actual convolution kernel G is obtained by subtracting the DC component (mean value) from the kernel itself and the resulting kernel is normalized so that its *L2*-norm is 1.

Because of the camera perspective effect, as shown in Figure 1, road section that is close to the camera (bottom of the image) appears wider than road section that is far away from the camera (top of the image). Different Gabor wavelets and kernel sizes are used to locate seed pixels depending on the location of search window on the image during seed search. As shown in Figure 1(c), kernel size decreases as the search window moves away from the camera (toward the top of the image).

In this work, Gabor wavelets with 11 Gaussian kernel sizes are used. They are created with

$$\sigma \in \{\frac{41+2i}{6} \mid i = 0, 2..., 10\} \tag{5}$$

For kernel sizes range from σ =6.833 (i=0 and size is 41×41 pixels) to σ =10.167 (i=10 and size is 61×61 pixels). Wavelength λ is calculated as

$$\lambda = 1.6\sigma + 2.4 \tag{6}$$

Thirty Gabor wavelet kernels (in a 6-degree interval) are constructed for each Gaussian kernel size. They are calculated as

$$\theta \in \{\frac{\pi}{30} i \mid i = 1, 2..., 30\} \tag{7}$$

This results in a total of 330 filters (11 kernel sizes × 30 wavelet orientations). Figure 3 shows a example set of 30 Gabor wavelets with σ =7.5 and 45×45 in size.

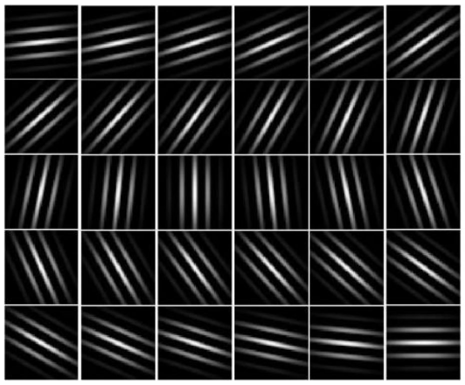


Figure 3. Example Gabor filter set with σ =7.5

3.2 Seed edge point search

As shown in Figure 4, a small region on each side of the image (highlighted in blue) is chosen for initial seed edge point search. A smaller region right above it is chosen for the secondary search if the edge point is not present due to dashed or faded line. The location of these search windows can be adjusted according to previous search result. In order to shorten the seed edge point search time, five initial search points in the search window on each side are used to perform a coarse search. Once the initial search result is obtained, a fine search is performed to refine search result to obtain more accurate seed edge point.

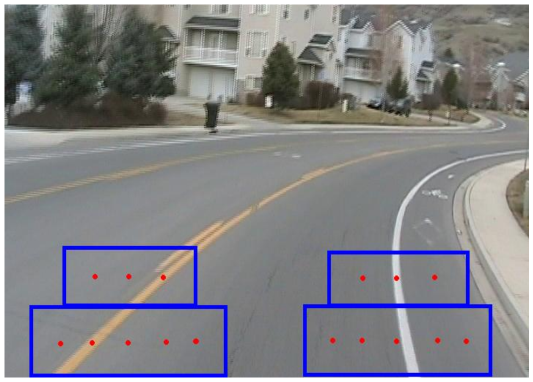


Figure 4. Seed edge point search windows and initial search points

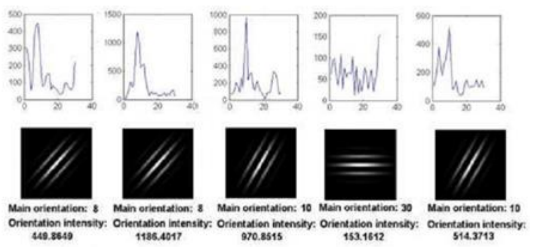
To locate seed edge point, the 30 Gabor filters of the largest size (σ =10.167 and 61×61 pixels) are first convolved with the image at the five initial search points in the search window. The result of this convolution process is stored in arrays V_l and V_r for the left and right search windows as shown in (8) and (9).

$$V_l(i,j) = (I * G_{real}{}^{j})(p^l{}_{(i,1)}, p^l{}_{(i,2)})^2 + (I * G_{im}{}^{j})(p^l{}_{(i,1)}, p^l{}_{(i,2)})^2 \tag{8}$$

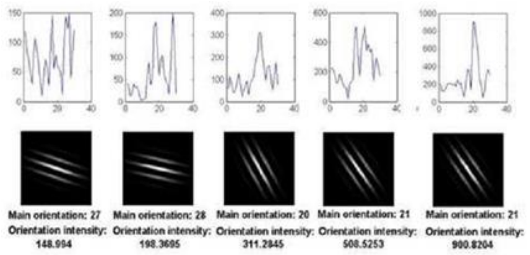
$$V_r(i,j) = (I * G_{real}{}^{j})(p^r{}_{(i,1)}, p^r{}_{(i,2)})^2 + (I * G_{im}{}^{j})(p^r{}_{(i,1)}, p^r{}_{(i,2)})^2 \tag{9}$$

In these equations, i is the index of the initial search points (i=1, 2, ···, 5) and j is the index of the Gabor filter orientations (j=1, 2, ···, 30). p^l and p^r are the initial points in the left and right search windows. G_{real} is the real part of the Gabor wavelet kernel and G_{im} is the imaginary part of the Gabor wavelet kernel.

The convolution result of the 30 Gabor filters at each initial search points is stored in one column of the arrays V_l or V_r and is plotted in Figure 5. The maximum convolution value and Gabor filter index are also shown. The Gabor filter that results in the maximum convolution value at each initial search point is shown below the plot. For example, at Point 1 in the left search window, the Gabor filter 8 (j=8 or 48 degrees) gives the highest convolution value of 449.864 9. Gabor filter 27 or 162 degrees gives the highest convolution value of 148.994 at Point 1 in the right search window. The largest convolution value of the five convolution locations in the left search window is at Point 2 (1 186.401 7) from Gabor filter 8 and the largest convolution value of the five convolution locations in the right search window is at Point 5 (900.820 4) from Gabor filter 21. Point 2 in the left window and Point 5 in the right window are chosen as the initial seed points and the orientation of these initial seed edge points is 48 and 126 degrees respectively.



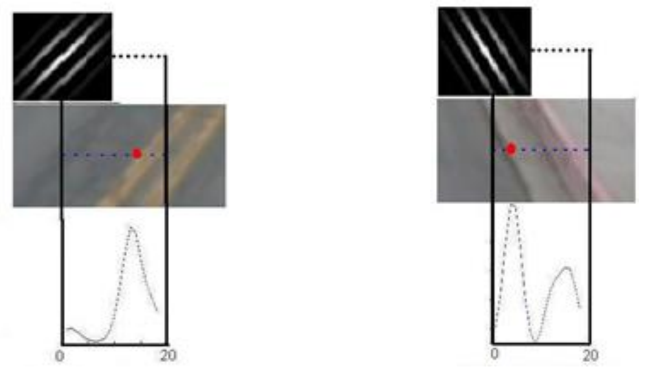
(a)Left side searching result



(b) Right side searching result

Figure 5. Result of convolutions of 30 Gabor filters for seed searching

The same search procedure for the coarse search is applied to the neighborhood of the initial seed points, points 2 and 5 in the left and right search windows, respectively. Figure 6 shows the refinement process. The refine search is performed by convolving the 30 Gabor filters in the neighborhood (half distance on each side) of the initial seed points in a much smaller step. Similarly, the point that has the largest convolution value from any of the 30 Gabor filters is chosen as the final seed edge point.



(a) Left side refining (b) Right side refining

Figure 6. Seed edge point refinement

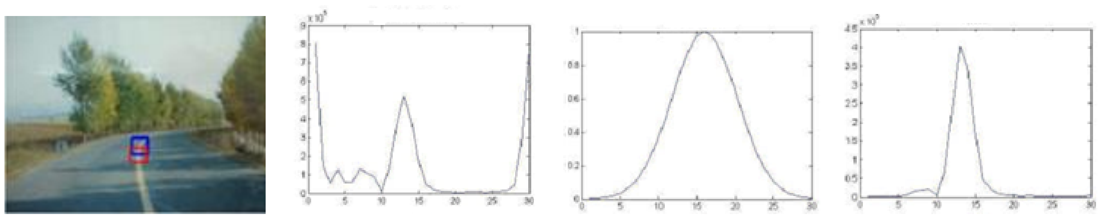
3.3 Growing and tracking the road edge

In order to shorten the computation time, the growing and tracking step is based on the assumption that the road would not curve abruptly in a short distance. Gabor filters that have the orientation close to the orientation of the previous edge point are used to convolute with the next search window. A Gaussian filter is constructed in (10) to give more weight to the Gabor filters with orientation that are close to the orientation of the previous edge point than those with very different orientation. This Gaussian filter is also used to minimize the effect of noise caused by the shadow or irrelevant objects mentioned in Section 1.

$$G(x) = \frac{1}{\sqrt{2\pi\sigma}} e^{-\frac{(x-\mu)^2}{2\sigma^2}} \{x \,|\, x = 1, 2, ..., 30 \,\&\, |x-\mu| < T\} \tag{10}$$

In (10), μ is the edge orientation from the previous search and is used as the mean of the Gaussian function. T is a threshold that determines how sharp the road curve can be detected and hence controls the growing process. The growing process is able to track normal road curve with a small T. If T is set too high, the growing algorithm will be easily affected by shadow or other intensity variations on the road. T is a cut-off threshold to reduce the effect of noise by eliminating the effect of Gabor filters with very different orientation.

The center of the red square shown in Figure 7(a) is the edge point detected from previous search. Its orientation index is 16 (96 degrees). Figure (b) shows the convolution results using 30 Gabor filters (6 degrees interval). Initial search results show that the maximum convolution values are from Filters 1 (6 degrees) and 30 (180 degrees). These two erroneous edge orientations represent an abrupt change in edge orientation from 16 (96 degrees) and are caused by the tree shadow as shown in Figure 7(a). Figure 7(c) shows the Gaussian filter with μ =16 (previous edge point) and σ =4.242.



(a) Image (b) Result of orientation filters (c) Gaussian filter (d) Result of Gaussia filter

Figure 7. Gaussian filter for growing process

As discussed previously, initial search results (6 degrees and 180 degrees) are caused by tree shadow and inaccurate. The final search result (78 degrees or index 13) can be obtained (Figure 7(d)) using the Gaussian filter designed based on orientation from previous search (96 degrees or index 16). Figure 8 shows result from the growing process. The orientation of the seed edge point (red dot in the blue box of Figure 8) is 11. The short line segments represent the search result.

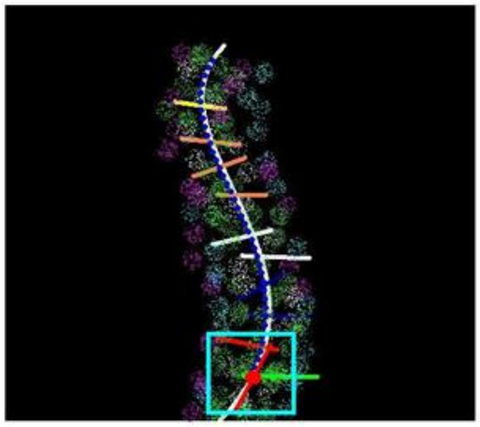


Figure 8. Result of growing process

4. Curve modeling

Road curve or road lane can be determined once all edge points are located. Our road curve modeling algorithm uses the detected edge points to fit a Bezier curve for one side of the road. As shown in Figure 9, the two road curves intersect at the vanishing point. Based on this assumption, a geometrical model is then derived to calculate the edge on the other side. Detail of this algorithm is described in this section.

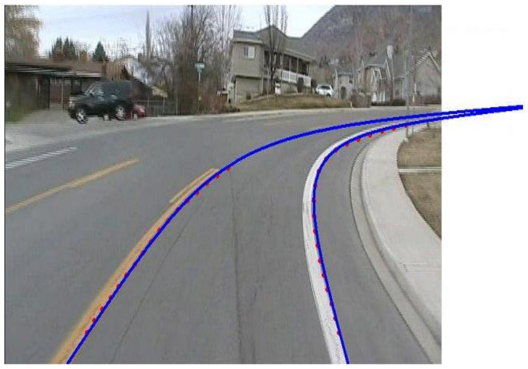


Figure 9. Road curve modeling

4.1 Q-Bezier-curve

Given a set of n+1 control points P_0, P_1, … P_n, the corresponding Bezier curve (or Bernstein-Bezier curve) is given as

$$p(t) = \sum_{i=0}^{n} P_i B_{i,n}(i), \mathrm{t} \in [0,1] \tag{11}$$

where $B_{i,n}(i)$ is a Bernstein polynomial.

The resulting curve model can be very accurate if all edge points detected are used as the control points. However, using too many control points will dramatically increase the order of Bernstein polynomial and require huge amount of computations. This is unnecessary since a square curve can very well represent most real world road curves.

In this work, only three control points are used to model the curve. As shown in (12), our algorithm uses q-Bernstein-Bezier curve [15, 16] which may be considered as a one parameter family of rational Bernstein-Bezier curve.

$$B_i^n(t,q) = \begin{bmatrix} n \\ i \end{bmatrix} t^i \prod_{s=0}^{n-i-1} (1 - q^s t), i = 0,1,...,n, t \in [0,1] \tag{12}$$

In (12), n is the dimension of q-Bernstein-Bezier curves. $B_i^n(t,q)$ is the basis polynomial of q-Bernstein-Bezier. Once the three control points are selected, there is just one parameter q left to be determined. An estimate function and all detected edge points can be used to compute the optimal q.

4.2 Bezier curve fitting

According to the property of the Bezier curve, both ends of detected serial points are used as the first and last control points. The second control point is the intersecting point of the two tangents of the two end control points (Figure 10). Once the three control points of the curve are determined, (11) and (12) can be used to generate one curve with a different parameter q. Figure 11 shows curves resulted from different q values.

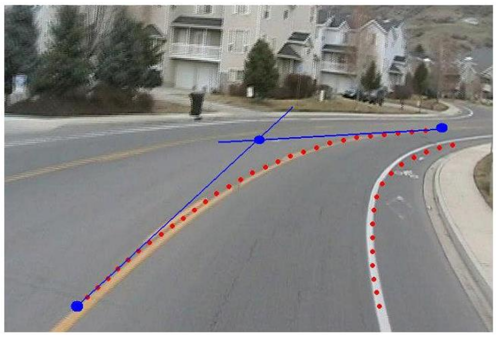


Figure 10. Three control points

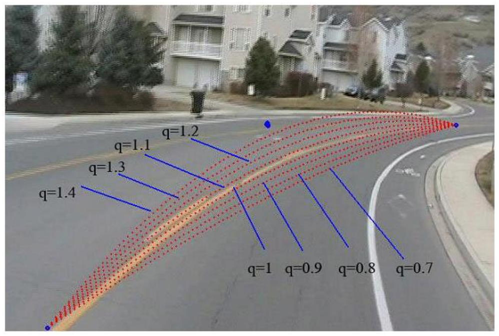


Figure 11. One parameter family with three control points

The goal is to determine q that minimizes $f(q)$ in the equation below for all edge points.

$$f(q) = \sum_{i=0}^{M} \left| p_i - l_{p(t,q)} \right| \tag{13}$$

where p_i is the i^{th} edge point. M is the total number of edge points. $|p_i - l_{p(t,q)}|$ is the distance from p_i to the Bezier line $l_{p(t,q)}$.

4.3 Road geometrical model

In this paper, our focus is on constructing a 2-D road lane mode. Once one side of the road edge is determined, the other side can be calculated without going through the same process again. An important assumption has to be made in order to achieve this goal without the full knowledge of 3-D information [6]. The assumption is that two sides of the road boundaries are parallel or close to parallel on the ground plane as shown in Figure 12.

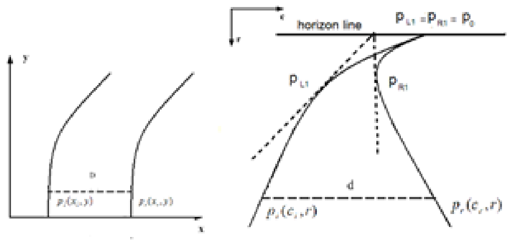


Figure 12. Parallel lines on ground plane and image plane

As discussed in [6], the distance relation between the ground plane and image coordinate system as follow:

$$d = \frac{\lambda^2 D(z - h_z)}{H(\lambda^2 + h_z^2)} \tag{14}$$

where λ is the focal length of the lens, H is the height of the camera location, h_z is the position of vanish line and r is the vertical coordinate used in the image plane (see Figure 10 for reference).

The horizontal distance *d* can be represented as

$$d = k(z - h_z) = kz - kh_z \tag{15}$$

$$k = \frac{\lambda^2 D}{H(\lambda^2 + {h_z}^2)} \tag{16}$$

We note that *d* defined by (15) is a monotone linear operator for (z- h_z). So once we fit one of the edge lines *r = f(c)*, the other edge line can be expressed directly by the following.

$$r = f(c \pm d) \tag{17}$$

where *r* is row in image coordinate system, *c* is column in image coordinate system.

When the other edge is right edge, the sign in f is positive while the sign is negative on the other side.

After we fit the line of one edge, the least squares fitting is introduced to compute the *k* and h_z in (15).

$$\begin{bmatrix} k \\ -khz \end{bmatrix} = \begin{bmatrix} \sum_{i=1}^{N} d_i r_i \\ \sum_{i=1}^{N} d_i \end{bmatrix} \begin{bmatrix} \sum_{i=1}^{N} {d_i}^2 & \sum_{i=1}^{N} d_i \\ \sum_{i=1}^{N} d_i & N \end{bmatrix}^{-1} \tag{18}$$

Where d_i is the transverse distance between the detected point *i* of the other line and the matched Bezier curve. r_i is the row of the point *i* and N is the total number of the detected points. As shown in Figure 13, the blue line is the matched Bezier curve while the red line is the other line fitted by using the perspective geometry road model.

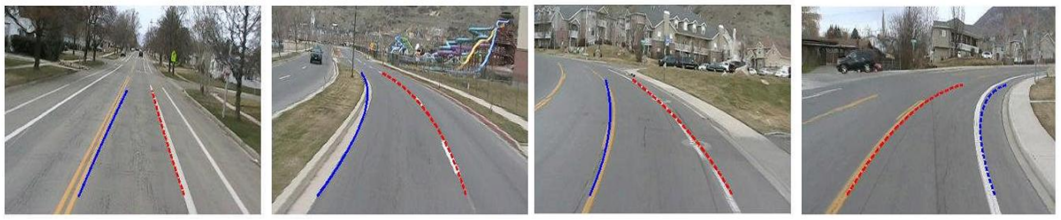


Figure 13. Fitting results

5. Conclusions and future work

The method has been tested on several kinds of road. It can detect the road lane very accurately in most circumstances. The results can be seen in Figure 14.

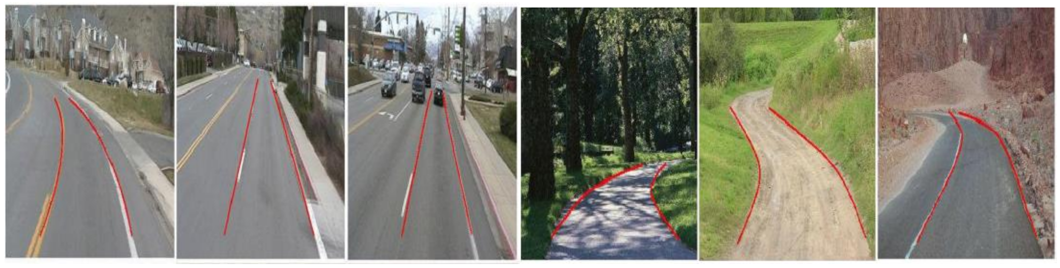


Figure 14. Results of road detection

A new road detection technique has been described in this article. The first module detects the road seeds in the bottom region of image. The road lane may occasionally be a dashed line instead of a solid line. If the bottom region fails to find a seed, the second bottom region which is upside of the bottom region is considered.

When the second module is about to track the seeds, we use the Gabor filter of growing rule to track the line until the stopping growing rule is reached. There are occasionally shadows and other interference on the edge. So we use the Gaussian filter to filter the result of Gabor filter.

The third module uses two methods to fit the lines. First, we use the q-Bernstein polynomials Bezier curve to fit the detected points of road line which has more reliability. At the other side of road line, we use the perspective geometry method to fit the other line according to the determined Bezier curve.

In the future we would like the algorithm to become more intelligent to search the seeds in order to adapt the recognition process to roads of any width. Our method can be easily combined with other modules (e.g. radar and GPS). With the radar, if there is an obstacle in front of the vehicle, we can determine the visibility of the roadway and stop growing the seed immediately in case the growing algorithm will be distorted by obstacle edges. Finally, the whole process will be implemented in our experimental vehicle in order to validate a complete driver-assistance system.

References

[1] A. Broggi and S. Berte, “Vision-based road detection in automotive system: a real-time expectation-driven approach”, Journal of Artificial Intelligence Research, vol. 3, **(1995)**, pp. 325-348.

[2] Y. Q. Wang, D. Y. Chen, C. X. Shi and P. D. Wang, “Vision-based road detection by Monte Carlo method”, Information Technology Journal, vol. 9, **(2010)**, pp. 481-487.

[3] C. Staufer and W. E. L. Grimson, “Learning patterns of activity using real time tracking”, IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 22, no. 8, **(2000)**, pp. 747-757.

[4] Y. He, H. Wang and B. Zhang, “Color-based road detection in urban traffic scenes”, IEEE Transactions on Intelligent Transportation Systems, vol. 5, no. 4, **(2004)**, pp. 309-318.

[5] C. Tan, T. Hong, T. Chang and M. shneier, “Color model based real-time learning for road following”, Proc. of IEEE International conference on Intelligent Transportation Systems, **(2006)**Toronto, Canada, pp. 939-944.

[6] Y. Wang, E. Khwang and D. G. Shen, “Lane detection and tracking using B-snake”, Image and Vision Computing, vol. 22, no. 4, **(2004)**, pp. 269-280.

[7] Y. Q. Wang, D. Y. Chen and C. X. Shi, “Vision-based road detection by adaptive region segmentation and edge constraint”, Second International Symposium on Intelligent Information Technology Application, **(2008)** Shanghai, China, pp. 342-346.

[8] A. Borkar, M. Hayes and M. T. Smith, “A template matching and ellipse modeling approach to detecting lane markers”, Lecture Notes in Computer Science, vol. 6475, **(2010)**, pp. 179-190.

[9] J. Z. Wang, Y. P. Chen, J. M. Xie and H. P. Lin, “Model-based lane detection and lane following for intelligent vehicles”, Proc. of 2nd International Conference on Intelligent Human-Machine Systems and Cybernetics, **(2010)** Nanjing, China, pp. 170-175.

[10] O. O-Khalifa, M. R. Islam, A. A. Assidi, A. A. Hashim and S. Khan, "Vision based road lane detection system for vehicles guidance", Australian Journal of Basic and Applied Sciences, vol. 5, no. 5, **(2011)**, pp. 728-738.
[11] C. Rasmussen, "Grouping dominant orientations for ill-structured road following", Proc. of IEEE Computer Society Conference on Computer Vision and Pattern Recognition, **(2004)**Washington, USA, pp. 470-477.
[12] P. Y. Jeong and S. Nedevschi, "Efficient and robust classification method using combined feature vector for lane detection", IEEE transactions on Circuits and Systems for Video Technology, vol. 15, no. 4, **(2005)**, pp. 528-537.
[13] O. Tunçer, L. Guvenç, F. Coskun and E. Karslıgil, "Vision based lane keeping assistance control triggered by a driver inattention monitor", Proc. of IEEE International Conference on Systems Man and Cybernetics, **(2010)** Istanbul, Turkey, pp. 289-297.
[14] M. Gschwandtner, W. Pree and A. Uhl, "Track detection for autonomous trains", Proc. of the 6th International Conference on Advances in Visual Computing, **(2010)** Las Vegas, USA, pp. 19-28.
[15] J. W. Choi, R. E. Curry and G. H. Elkaim, "Curvature-continuous trajectory generation with corridor constraint for autonomous ground vehicles", Proc. of 49th IEEE Conference on Decision and Control, **(2010)** Atlanta, Georgia, USA, pp. 7166 – 7171.
[16] R. P. Huang, "Construction and application of rational q-Bernstein-Bezier curves", Journal of Computer Applications, vol. 30, no. 5, **(2010)**, pp. 1359-1362.

Author

Wei Li. She received the Master degree(2010) in Electronics and Communications Engineering from Shanghai University, China. Now she is a lecturer of School of Electronics and Information, Nantong University, Jiangsu, China. Her research interests include Electronics and Communications Engineering.

E-mail: lw@ntu.edu.cn