Turkish Journal of Electrical Engineering & Computer Sciences

http://journals.tubitak.gov.tr/elektrik/

Research Article

Turk J Elec Eng & Comp Sci
(2014) 22: 1484 – 1500

doi:10.3906/elk-1209-51

Development and optimization of a DSP-based real-time lane detection algorithm on a mobile platform

Gürkan KÜÇÜKYILDIZ*, Hasan OCAK
Department of Mechatronics Engineering, Umuttepe Campus, Kocaeli University, Kocaeli, Turkey

Received: 11.09.2012 • **Accepted:** 02.01.2013 • **Published Online:** 07.11.2014 • **Printed:** 28.11.2014

Abstract: In this study, image processing-based real-time lane detection, which is one of the significant problems in autonomous vehicle control, is explored. A mobile robot platform is developed for that purpose. The motion of the mobile robot is provided by 4 direct current motors, which are independently controlled. An image processing code is developed in a Visual DSP 5.0 environment and run on a BF-561 processor embedded in the ADSP BF-561 EZ-KIT LITE evaluation board (Analog Devices). In the image processing algorithm, Hough lines obtained from the Hough transform of the captured images are called candidate lane marks. Various elimination methods are implemented on these candidate lane marks to detect the actual lane marks. Once the actual lane marks are determined, the real-world coordinates of these lane marks are computed using inverse perspective mapping. The heading angle of the mobile robot is then determined based on the position of the lane marks and the center of the mobile robot. The developed mobile robot platform and the lane detection algorithm are tested under various conditions, including dashed lane marks, varying lightening conditions, and the presence of one lane mark only. It is observed that the algorithm performs successfully and detects the lane marks even in the presence of various disturbance effects under these conditions. After the optimization of the developed image processing code, the number of frames processed by the algorithm increases from 3/s to 30/s, which should be satisfactory for real-time applications.

Key words: Hough transform, mobile robots, lane detection, DSP, real-time operating

1. Introduction

Due to technological developments, the autonomous vehicle concept, which has been one of the most important dreams of scientists for years, is now closer to becoming a reality. Autonomous vehicles can navigate themselves without driver assistance by sensing their environment with the help of various sensors, such as global positioning system, radar, and camera. They can determine and update their routes by processing sensor data with various control algorithms. The aim in building autonomous vehicles is to decrease the number of driver fault-based traffic accidents and avoid human deaths. Google has been the biggest supporter of autonomous vehicle research by developing 4 autonomous cars. In Nevada, USA, the first law for the usage of autonomous vehicles in traffic was published in June 2011 with strong support from Google [1].

Vision-based lane detection is one of the most studied topics of the autonomous vehicle concept [2]. The most common problems encountered in vision-based lane detection studies are the varying distances between the lane marks, lane mark thickness, unstable light conditions, dashed lane marks, shadows, and noise. Aside from these problems, computational costs are also a major problem in developing vision-based lane detection

*Correspondence: gurkan.kucukyildiz@kocaeli.edu.tr

algorithms. Because of high computational costs, most lane detection algorithms presented in the literature are not suitable for real-time applications. Despite several studies on increasing the algorithm speed and overcoming the problems mentioned above, real-time implementation of lane detection algorithms still constitutes a big problem. In this study, the proposed lane detection algorithm is optimized in several ways to make it satisfactory for real-time applications.

Some of the lane detection algorithms presented in the literature are based on the randomized Hough transform [3–7], B-snake method [8], dynamic programming [9,10], Kalman filtering [11,12], particle filtering [13], support vector machines (SVMs) [14], artificial neural networks (ANNs) [15], likelihood of image shapes [16], and hue-saturation-intensity (HSI) color models [17].

SVM- and ANN-based lane detection algorithms have more computational costs compared to others. Mandalina and Salvucci successfully applied SVMs to various images and proved that the algorithm could perform lane change detection in 1.2 s [14]. Jochem et al. developed an ANN-based lane detection algorithm named the autonomous land vehicle in a neural network (ALVINN) [15]. After the training phase, the multiple ALVINN networks in autonomous control system, which consists of a combination of ALVINN systems, could process 15 frames/s, which allows a car to be driven at 55 mph, based on their claim.

Wang et al. used the randomized Hough transform technique on a specific region of interest for detecting lane marks [7]. By applying the Hough transform on a selected region in a frame instead of the full frame, the computational cost of the Hough transform was noticeably decreased. Wang et al. used the B-snake algorithm for detecting lane marks [8]. They modeled the road as a parabolic notation, which was called B-snake. They determined the start position of the B-snake using the Canny/Hough estimation of the vanishing point algorithm. Borkar et al. used the Kalman filter, which is commonly used for estimation, for lane detection [11]. They took the Hough transform of an image and identified the vanishing points of the lines. They also used the random sample and consensus genetic algorithm for determining the road model. Sun et al. used a HSI color model instead of red-green-blue color representation for lane detection [16]. They used the fuzzy c-means algorithm for image segmentation. After the segmentation and filtering, the connected components labeling algorithm was applied to the image. The proposed frame processing time in their algorithm was nearly 10 ms.

Most of the algorithms presented in the literature were tested offline using personal computers and high-level programming languages. MATLAB and C with the Open CV library are the most commonly used programs to implement the methods developed in lane detection studies. Most of the mathematical functions used in these studies, such as the Hough transform, edge filtering, and many other image processing operations, are available to developers as prewritten functions in both MATLAB and the Open CV library. Typically, personal computers have faster processors and RAM compared to digital signal processor (DSP) boards. In this study, the ADSP BF-561 evaluation board from Analog Devices, which has a 600-MHZ processor and 120-MHZ SDRAM, is used for developing the image processing code. All of the image processing functions are written and optimized in the Visual DSP 5.0 environment using the C programming language with the absence of an image processing library. The video frames are captured by a National Television System Committee (NTSC) (30 fps) formatted Hitachi KPD-20B model camera with a 4-mm lens attached to it to improve its field of view. A mobile robot platform, as shown in Figure 1, is developed during the study and the image processing code is tested on this platform in real-time.

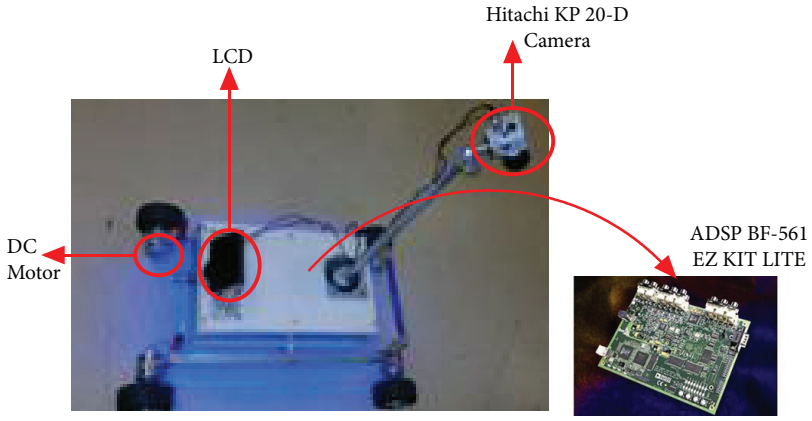

Figure 1. Developed mobile robot platform.

The image processing code developed for lane detection could initially process only 3 frames/s. However, after both the code and DSP-based optimizations, the proposed lane detection scheme is able to process 30 frames/s (NTSC format limit), which is sufficient for real-time applications. This processing speed corresponds to the detection of lane marks at every 1 m for a vehicle running at 120 km/h. In addition to real-time processing, robustness is another important issue that should be considered for lane detection algorithms. Various disturbances, such as dashed lane marks, the presence of a single lane mark, noise, shadows, and varying lighting conditions, can make a lane detection algorithm produce incorrect results. The results reveal that the proposed scheme successfully detects lane marks under all of the conditions tested.

2. Technical background

2.1. Hough transform

Although the Hough transform technique was originally proposed by Duda and Hart in 1972, the technique only started to be used frequently for processing images after a study published by Dana in 1981 [18]. With this study, it was understood that the Hough transform could be used for detecting not only lines, but also circles, ellipses, and any parametrically representable group of points in images [19–23].

The Hough transform can be used to detect lines at any orientation in digital images. The transform maps a point $(x,\ y)$ in Cartesian space to sinusoidal curves in $(\rho,\ \theta)$ space via the following transformation:

$$\rho = x\cos(\theta) + y\sin(\theta). \tag{1}$$

An accumulator matrix $\mathbf{T}(\rho,\ \theta)$ is used to count the number of sinusoidal curves that pass through $(\rho,\ \theta)$ in the parameter space. The points in the parameter space with the highest counts or scores will correspond to lines in the analyzed image. The pseudocode for the Hough transform can be given as:

For each pixel in the binary image
    If the pixel is white {
        For each θ {
            Compute $\rho = x\cos(\theta) + y\sin(\theta)$
            $\mathbf{T}(\rho,\ \theta) = \mathbf{T}(\rho,\ \theta) + 1$ }
    }

The parameter θ can take values from –90° to 90°. The angular resolution used in the computation of the Hough transform determines the number of columns in the accumulator matrix. For a resolution of 1°, the accumulator matrix will have 181 columns. The parameter ρ can take values from 0 to ρ_{max}, where ρ_{max} is the diagonal length of the analyzed image. The resolution in the ρ values determines the number of rows in the accumulator matrix. The higher either the angular or the ρ resolution is, the larger the accumulator matrix is, which leads to increased computational time. Reducing the dimensions of the accumulator matrix decreases the computational time; however, it can make the algorithm produce the wrong results because of decreased resolution.

Although the Hough transform is robust against noise and gaps, the technique has a serious disadvantage, which is its computational cost. Because of this computational cost, Hough transform-based lane detection algorithms are not suitable for real-time applications unless a modified version of the transform is used and/or the corresponding code is optimized [24]. Detailed information on the optimizations performed in this study for making the algorithm applicable to real-time applications is discussed in Section 4.

2.2. Inverse perspective mapping

Perspective effect, which is a common problem in cameras, occurs because of matching the trapezoidal field of view of a camera with a rectangular shape in the image domain [25,26]. This effect is best seen in the way the parallel lines appear to intersect at a distant point, which is referred to as the vanishing point. Various methods were proposed in the literature for reducing or eliminating the perspective effect [27,28]. Inverse perspective mapping uses a 3 × 3 homography matrix to translate points from the image plane to the points in the real-world coordinates. Figure 2 illustrates both coordinate systems for a fixed position of the camera used in this study. The trapezoid shown on the left side of Figure 2 represents the field of the camera on the road plane. The measurements are given in centimeters. The rectangular region shown on the right side of Figure 2 illustrates the image plane.

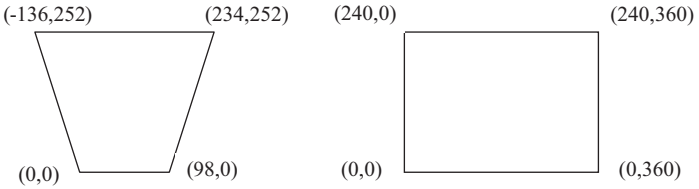

Figure 2. Real-world and image coordinate systems.

Transformation from image coordinates to real-world coordinates can be accomplished by:

$$\begin{bmatrix} x^* \\ y^* \\ k \end{bmatrix} = H \begin{bmatrix} j \\ i \\ 1 \end{bmatrix}, \tag{2}$$

where **H** is known as the 3 × 3 homography matrix, x^* and y^* are the unnormalized real-world coordinates, k is the normalization factor, and i and j are the row and column indices of the pixel in the image coordinate system, as given in Figure 2, respectively. To obtain the real-world coordinates, x^* and y^* should be normalized by k.

A representative image and its inverse perspective mapped version are illustrated in Figures 3 and 4, respectively. It can be seen that in the inverse perspective mapped image, the 2 lanes appear parallel, whereas

they seem to intersect at a distant point in the original image. The inverse perspective mapping is only applied to the detected lane marks instead of the entire image to reduce the computational time.

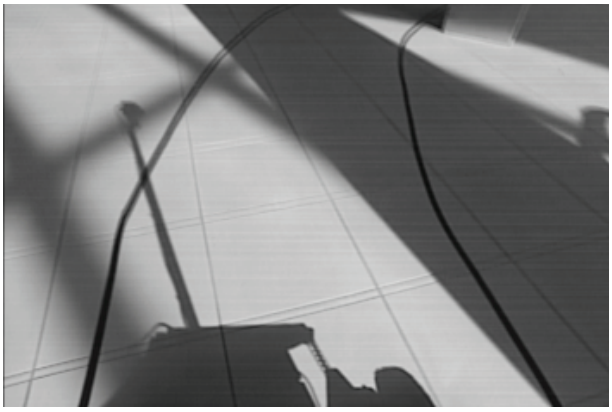

Figure 3. A representative image captured from the camera installed on the robot.

Figure 4. Inverse perspective transform of Figure 3.

3. Methods

A basic flow chart of the proposed algorithm is provided in Figure 5. The image format used by the ADSP BF-561 evaluation board is the YUV format, also known as the YC_bC_r format, where Y is the luma and C_b and C_r are the chrominance components. Since the luma is the luminance or the brightness, the captured frame is converted to gray-scale by simply taking the Y component of each pixel. The gray-scaled image is converted to a binary image by first applying a vertical filter highlighting the vertical lines and then thresholding the filtered image. The vertical filter is used since the lanes are known to be close to vertical. Instead of a typical 3×3 vertical Sobel filter [29], [1 0 –1] is used to save computational time. With the Sobel filter, 2 multiplications, 3 subtractions, and 2 additions are needed per pixel to compute the filter response. On the other hand, the filter used requires only 1 subtraction per pixel. The filtered image is converted to a binary image by applying a threshold. The Hough transform is then applied to the binary image. The local maxima of the Hough matrix are found in the parameter space and the corresponding Hough lines are marked as candidate lane marks. In this study, various elimination methods are used to detect the desired lane marks among all of the Hough lines detected. These methods will be explored in the next section.

3.1. Elimination methods

The first elimination is performed by applying a threshold to the computed local maximum points. Any local maximum having a value less than a certain threshold is disregarded. By testing with various images, the value for the threshold is set to 20. Thus, candidate lines are required to consist of at least 20 pixels. Next, only a certain number of K, the highest local maximum points, are kept and the lines in the original image that correspond to these maxima are marked as candidate lane marks. Again, by trial and error, the value for K is set to 100. The candidates are analyzed in pairs to locate the pair that corresponds the actual lane marks. The candidate pairs are required to satisfy several criteria for being selected as the actual lane marks. First, the lines represented by the candidate pairs are required to be parallel. However, this criterion should be slightly relaxed since one cannot expect the lane marks to be perfectly parallel. In addition, the parameters of the lines detected by the Hough transform have only a limited resolution. Therefore, if the distance between the bottom intercept points of the lines is equal to the distance between the top intercept points of the lines within a 10%

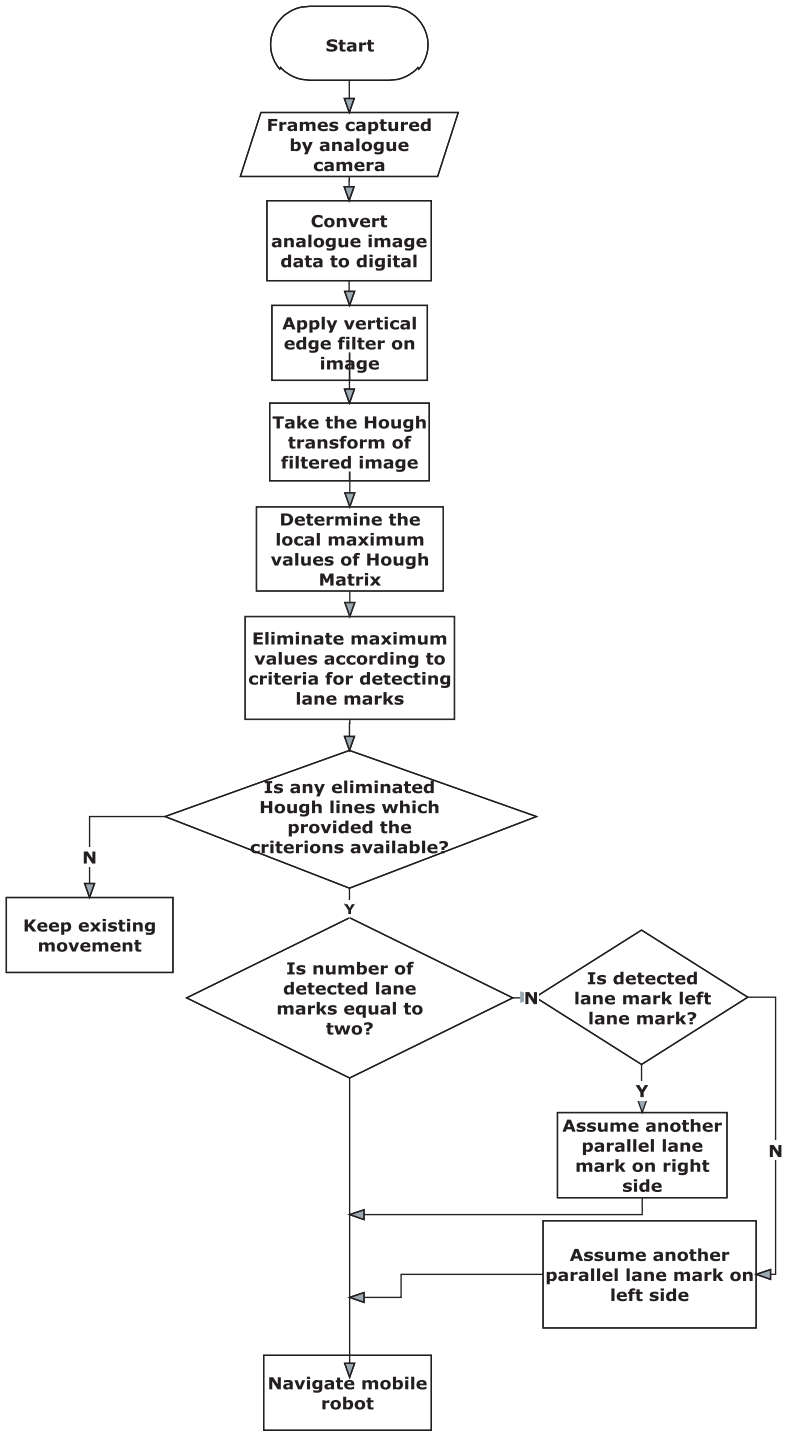

Figure 5. The basic flow chart of the proposed algorithm.

tolerance, the candidate lines are assumed to be parallel to each other. Next, the distance between the lines in the candidate pairs must be reasonably close to the predetermined lane width. Finally, the candidate pairs that satisfy the 2 criteria are given a score as the sum of the Hough scores of the lines in the pair. The lines in the pair with the highest score are selected as the actual lane marks.

When none of the candidate line pairs satisfy the given criteria, the algorithm checks for the possibility of detecting a single lane mark. For this, the algorithm evaluates the K candidate lines separately. The real-world coordinate of the bottom intercept point of each line candidate is computed using inverse perspective mapping. This point is required to be outside of the mobile robot's footprint within a reasonable distance. If not, the line is eliminated. Among the remaining lines, the one with the highest Hough score is selected to be the actual lane mark. After a single lane mark is detected, it is decided whether it is the right or the left mark based on its location with respect to the mobile robot. Next, the second lane mark is artificially inserted into the display parallel to the existing one at a distance determined by the lane width.

3.2. Mobile robot navigation

The heading angle of the mobile robot is determined using the real-world coordinates of the bottom and top intercept points of the lane marks, which are obtained using inverse perspective mapping. Figure 6 depicts a sample image, where both the detected lane marks and the inverse perspective mapped lane marks are given. The red and blue lines in Figure 6 correspond to the detected and inverse perspective mapped lane marks, respectively. It can also be seen in Figure 6 that the inverse perspective mapped lane marks are parallel to each other, similar to real-world conditions. Figure 7 illustrates the detected lane marks in the real-world coordinate system. The origin in this coordinate system represents the position of the mobile robot. The mobile robot is always steered towards the middle of the lane marks 80 cm ahead of it. The green line in Figure 7 shows the computed heading direction for the mobile robot. To steer the mobile robot to the right, the duty cycles for the pulse-width modulation (PWM) signals for the left motors are increased from its base value by an amount proportional to the heading angle determined from Figure 7. Similarly, the PWM signals for the right motors are decreased from its base value by an amount proportional to the heading angle. The mobile robot is steered to the left in a similar approach. In case where no lanes are detected by the algorithm, the mobile robot keeps moving in its previous direction. This navigation strategy successfully keeps the mobile robot inside the lane marks.

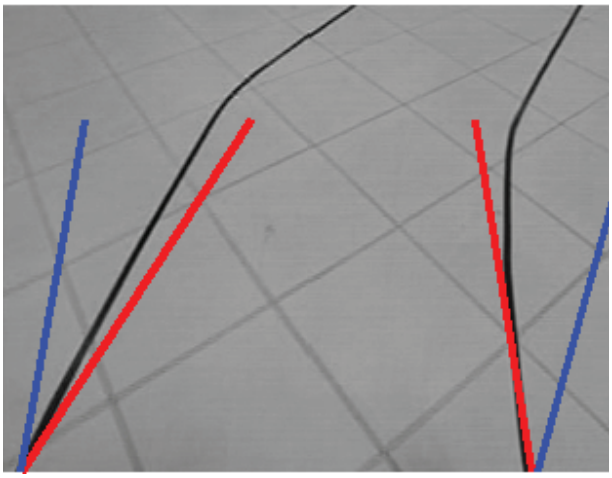

Figure 6. Detected and inverse perspective mapped lane marks.

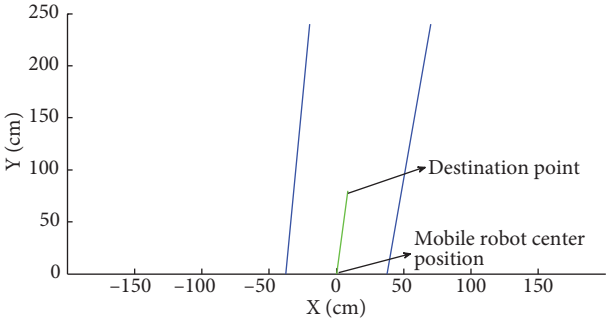

Figure 7. Heading direction computation for the mobile robot.

4. Optimization and real-time operating

The mobile robot developed for this study could be driven at a maximum speed of 4 km/h (1.1 m/s) due to hardware limitations. The mobile robot is about 80 cm long and 60 cm wide, and the lanes are approximately 80 cm wide. Given the maximum speed of the mobile robot platform, 6 frames should be processed each second to be able to send a control signal to the robot at every 20 cm. After both the code and DSP-based optimizations, the proposed lane detection system is able to process 30 frames/s (NTSC format limit), which is sufficient for real-time applications. This processing speed corresponds to the detection of lane marks at every 1 m for a vehicle running at 120 km/h. The optimization methods used in this study can be grouped into 2 categories: algorithmic optimizations, and DSP memory- and coding-based optimizations. Algorithmic optimizations are related to the reduction of the computational cost of the Hough transform. DSP memory and coding optimizations are related to the coding technique of the ADSP BF-561.

4.1. Algorithmic optimization

For each edge pixel in the analyzed image, ρ values are computed for each θ in the interval [–90°, 90°] when computing the Hough transform. For a resolution of 1°, this is equivalent to 181 computations for each edge pixel, which takes a lot of processer time. Instead of iterating through all of the possible values, the θ values for the edge pixels can be estimated from the gradient. The gradient in the y-direction, G_y, is computed by applying a basic vertical filter ([1 0 –1]) to each pixel. If G_y is below a certain threshold (i.e. the pixel is an edge pixel), a basic horizontal filter is applied to the pixel to compute the gradient in the x-direction, G_x. The arctangent of the G_y/G_x ratio is computed to get the gradient direction for the pixel. This is an estimate of the edge direction named θ_{est}. ρ values are now computed for each θ in the interval $[\theta_{est} - \delta_\theta, \theta_{est} + \delta_\theta]$ instead of [–90°, 90°]. Since the gradient direction is merely an estimate of the edge direction, the term δ_θ is used to account for the errors in the estimate. By testing with various values, the optimum value for δ_θ is found to be 6°. This reduces the number of ρ value computations for each edge pixel from 181 to 13, which reduces the computational time of the Hough transform by roughly 14.

4.2. Usage of lookup tables (LUTs)

Trigonometric functions sine and cosine are extensively used in the computation of the Hough transform. In addition, an inverse tangent function is used to compute the gradient direction of each edge pixel. For a fixed point processor, such as the one used in this study, these functions, especially the inverse tangent, take a long time to compute. To avoid this computational cost, the values for these trigonometric functions are computed at the startup of the program for all of the possible angles with a resolution similar to the angle resolution of the Hough transform. The values are then stored in LUTs. The usage of LUTs increases the number of frames/s processed by the algorithm by roughly 3. The number of frames processed by the proposed method before and after the LUTs is given in Table 1.

Table 1. Usage of a LUT.

Before the LUT		After the LUT	
θ is estimated	θ is not estimated	θ is estimated	θ is not estimated
11.25 fps	2.5 fps	30 fps	7.5 fps

4.3. DSP-based optimizations

4.3.1. Dual-core operating

BF-561 is a dual-core processor and each core can operate independently at a speed of 600 MHz. Dual-core operating provides multithreading for image processing codes. In this study, core A of the processor is used for transferring the camera-captured image data to the main memory. Core B of the processor is used for processing the image placed in the memory and for generating the PWM signals that control the 4 DC motors of the mobile robot. Core B is also used for displaying the processed frame on a small LCD monitor, which is installed on the mobile robot. Displaying the processed frame and the results on a LCD aids the authors in the development of the code.

4.3.2. Memory management

The ADSP BF-561 EZ-KIT LITE evaluation board has 3 types of memories: L1, L2, and SDRAM. The L1 memory operates at a core clock frequency of 600 MHz; hence, it has the lowest latency compared to the other memory spaces. It also holds the instructions. However, it has a limited capacity of 64 Kbytes. Therefore, large variables such as the image data and the Hough matrix cannot be placed in this memory. The L2 memory is another type of memory, which operates at a speed of 300 MHz and consists of 128 Kbytes of memory.

The SDRAM, which operates at a speed of 120 MHz, has the biggest capacity of the 3 memory types. It has 256 Mb of capacity and consists of 4 different banks. To minimize the access-based latency, the 2 cores of the processor must not simultaneously access the same bank. To accomplish this, the frames are captured in a circular buffer mode with 2 buffers defined in separate SDRAM banks. Core A of the processor sends a signal to Core B when a frame is captured in the first buffer. Core B then starts processing the data in this buffer, while another frame is being captured into the second buffer. The next frame is then captured in the first buffer, which continues in a circular fashion. This strategy minimizes the latency in accessing the data in the SDRAM by making sure that both cores do not access the same SDRAM simultaneously.

Because of the image size (480 $\times$ 720 NTSC format), each frame must be captured in the SDRAM memory. However, processing the image placed in the SDRAM would be slow since it operates at a processor speed of 120 MHz compared to 600 MHz. Therefore, to increase the processing speed, the captured images are resized to 240 $\times$ 360 so they can be placed in the L2 memory, which operates 2.5 times faster than the SDRAM. The Hough matrix is also defined in this memory since the code frequently accesses it to evaluate the candidate lines and detect the actual lane marks.

5. Results

The developed mobile robot platform and lane detection algorithm are experimented on under various conditions, including dashed lane marks, varying lightening conditions, the presence of one lane mark only, and additive Gaussian white noise.

Lane marks can be solid as well as dashed on real road scenes. Therefore, a real-time lane detection algorithm should function under both scenarios. Dashed lane marks can possibly lead to lane detection failure. The proposed lane detection scheme is tested under the presence of dashed lane marks to evaluate its performance. Figure 8 illustrates the results for this test, where row 1 shows the 2 images captured when the lane marks are dashed and row 2 depicts the Hough lines detected through the Hough transform. Finally, row 3 of Figure 8 illustrates the detected lane marks after the elimination step.

Due to changes in lighting in real-world conditions, disturbances such as shadows and noise are inevitable. These disturbances make lane mark detection a very challenging task. In under- and over-lit environments the

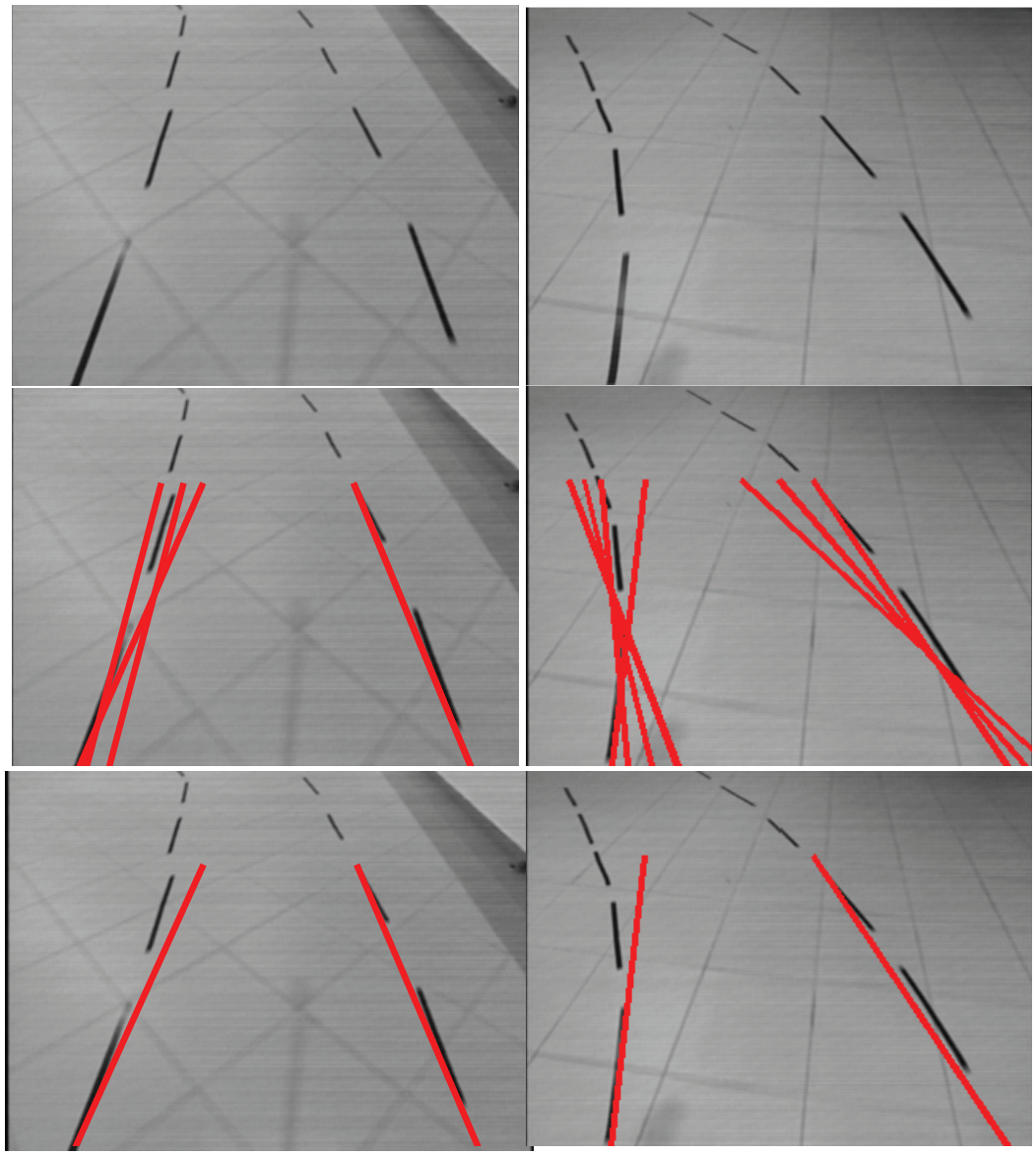

Figure 8. Row 1: The 2 original images for dashed lane marks, row 2: detected Hough lines, row 3: detected lane marks.

algorithm can produce erroneous results. In an under-lit environment, the number of detected Hough lines increases, as does the computational cost. Figure 9 illustrates the lane detection results for the 2 images captured in an under-lit environment. Row 1 of Figure 9 shows the 2 original images, row 2 depicts the Hough lines detected through the Hough transform, and row 3 illustrates the detected lane marks after the elimination step. The experimental results for the 2 images captured in an over-lit environment are shown in Figure 10. Row 1 of Figure 10 shows the 2 images, and shadows due to the sun are clearly visible in both images. Row 2 of Figure 10 depicts the Hough lines detected through the Hough transform and row 3 illustrates the detected lane marks after the elimination step. The experimental results prove the efficacy of the method, even in the presence of various disturbances like shadows and noise under varying lighting conditions.

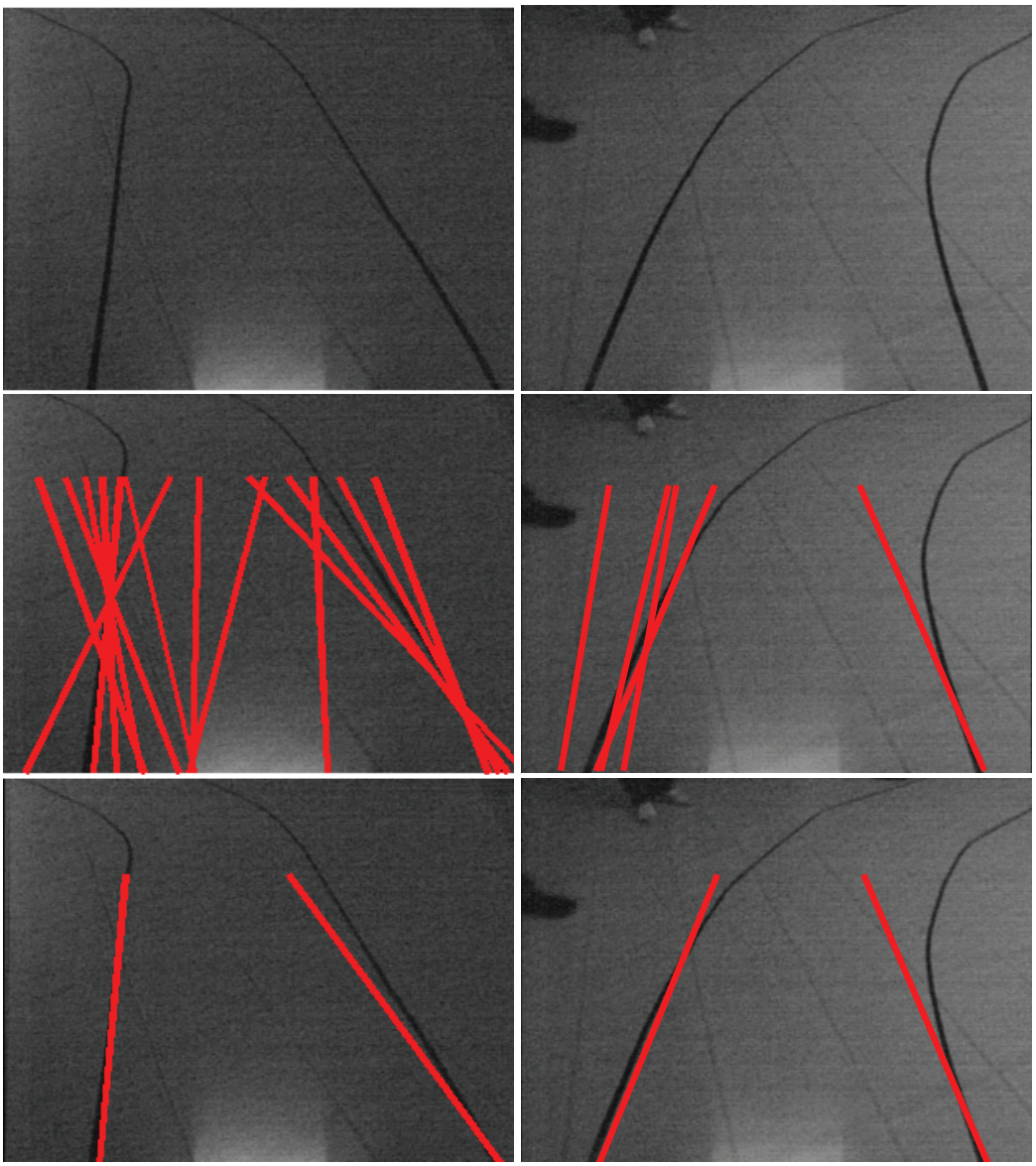

Figure 9. Row 1: The 2 original images captured under low-light conditions, row 2: detected Hough lines, row 3: detected lane marks.

In some road scenes, only a single lane might be present, constituting a problem for lane detection algorithms. In such a case, another lane mark is assumed to exist parallel to the detected one at a distance determined by the lane width. This helps the algorithm compute the heading angle for the mobile robot, as illustrated in Figure 6. The experimental results for the 2 images captured for 1 lane mark case are shown in Figure 11. Row 1 of Figure 11 shows the 2 images and row 2 depicts the Hough lines detected through the Hough transform. Finally, row 3 of Figure 11 illustrates the detected lane mark after the elimination step.

To test the robustness of the proposed lane detection scheme with quantitative measures, Gaussian white noise is added to the images with varying signal-to-noise ratio (SNR) levels, from 3 dB to 8 dB. Tests are conducted with 100 frames acquired from the camera mounted on the mobile robot platform. The success of the algorithm in terms of percentages is noted under different SNR levels. The results for the test are

summarized in Table 2. The algorithm successfully detects lanes in 99%, 97%, 95%, 92%, 75%, and 59% of the frames for SNR levels of 8 dB, 7 dB, 6 dB, 5 dB, 4 dB, and 3 dB, respectively. Representative frames are given in Figure 12 for all of the SNR levels. It is a success of the algorithm that it accurately detects the lanes 92% of the time, even for very noisy images having 5 dB SNR.

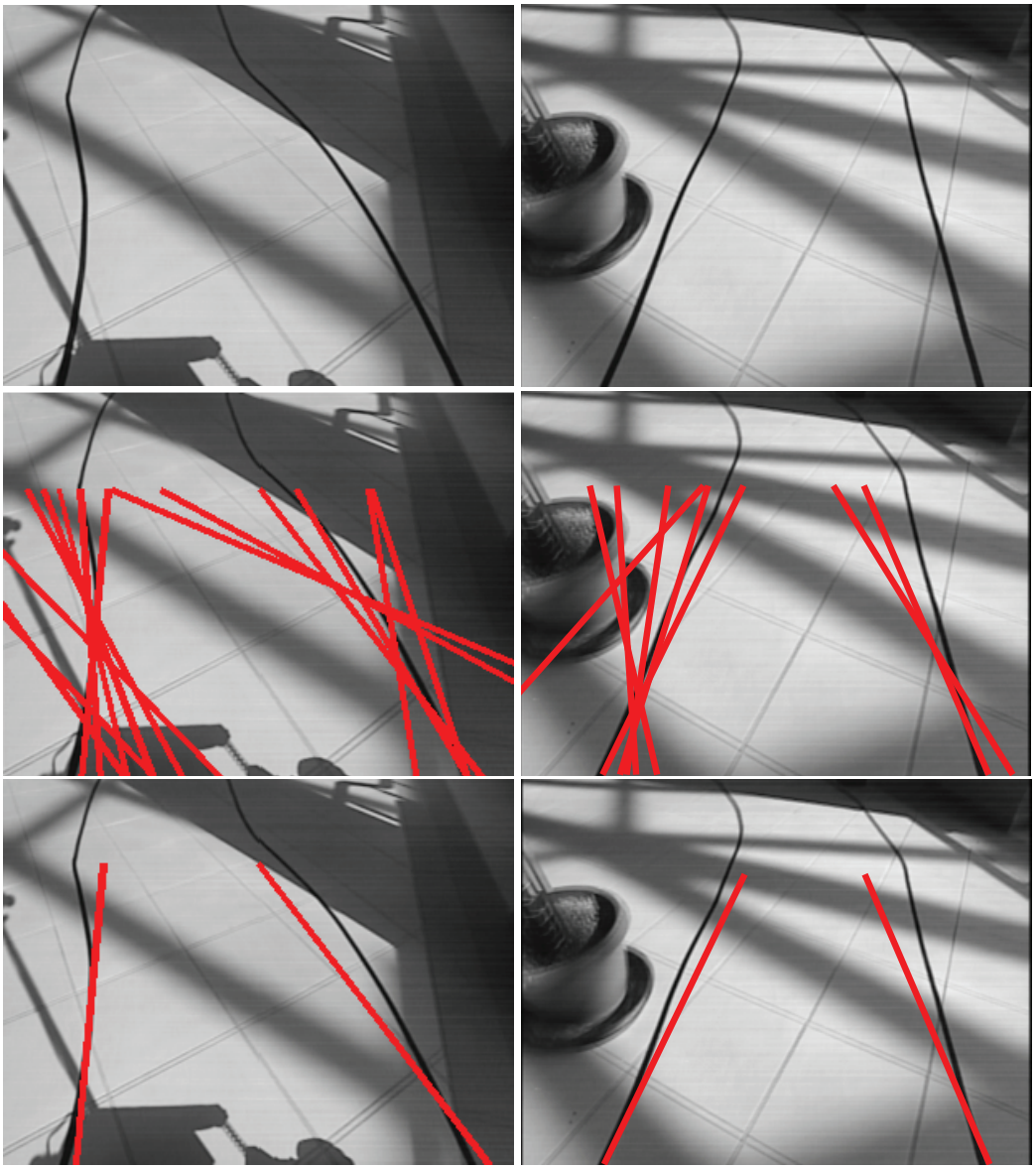

Figure 10. Row 1: The 2 original images captured in an over-lighted environment, row 2: detected Hough lines, row 3: detected lane marks.

Table 2. Success rates of the proposed algorithm at various SNR levels.

SNR	3 dB	4 dB	5 dB	6 dB	7 dB	8 dB
Success rate	59%	75%	92%	95%	97%	99%

The algorithm is finally tested with several real-life road images taken from the Carnegie Mellon Image Database [30]. Sample results for the real-life images are provided in Figure 13. A vehicle and intense shadows of the trees exist as disturbances in the 2 images. The algorithm successfully detects lane marks for both images.

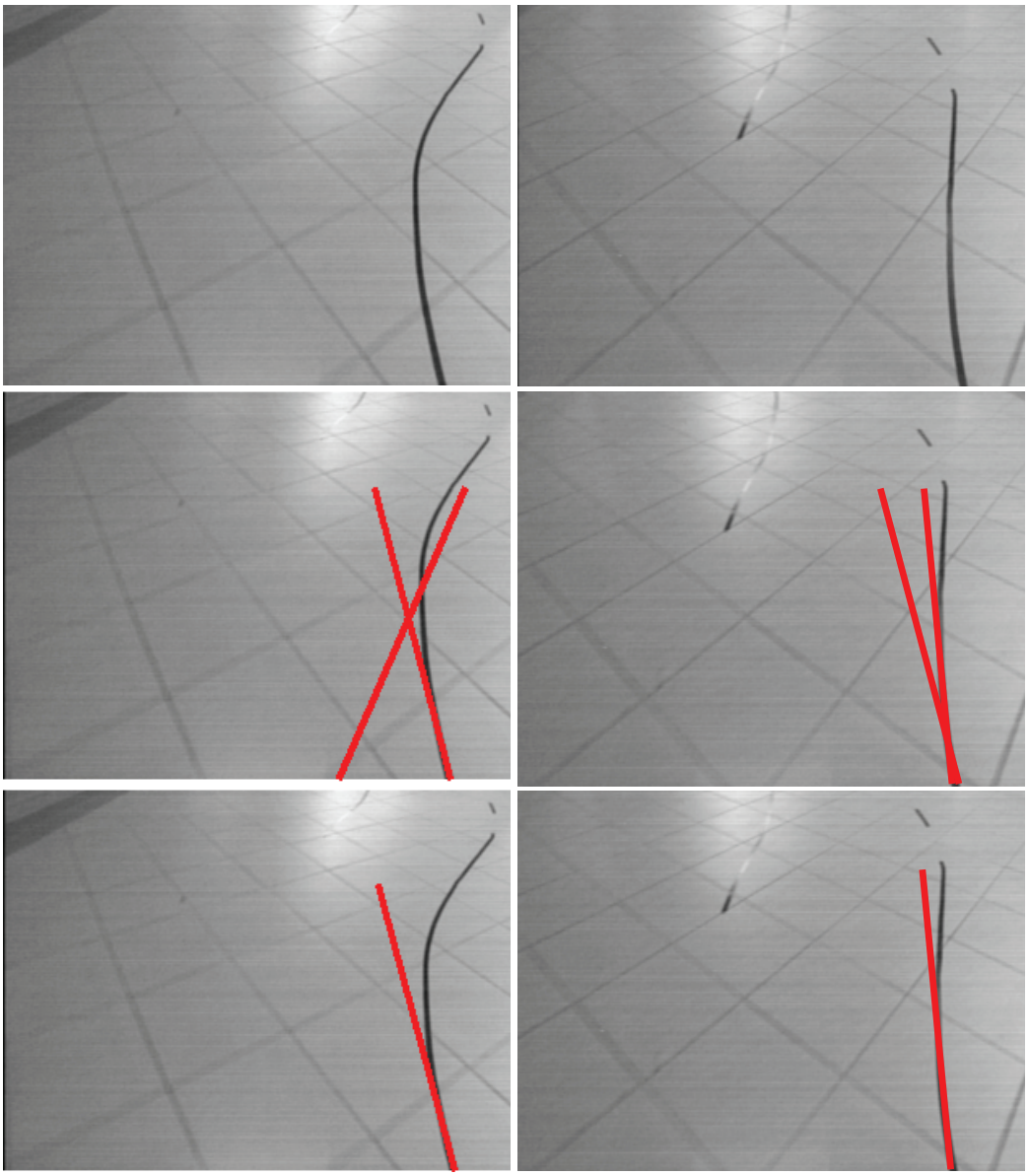

Figure 11. Row 1: The 2 original images for the presence of only right lane mark, row 2: detected Hough lines, row 3: detected lane marks.

Experimental results under various disturbance effects show that the proposed lane detection system is robust and successfully keeps the mobile robot platform inside the lane marks.

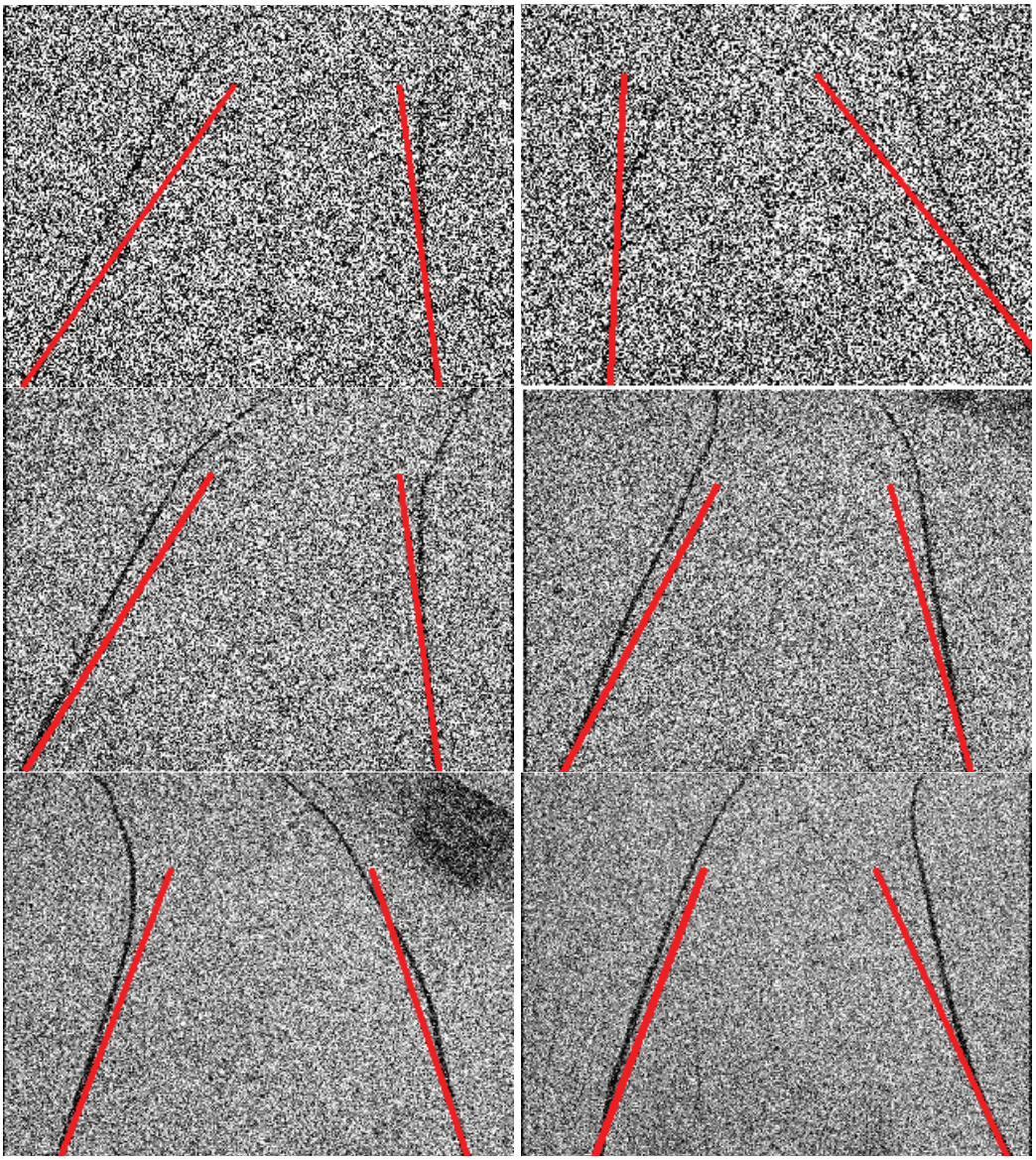

Figure 12. Representative images with additive Gaussian white noise. Row 1: SNR = 3 dB and 4 dB from left to right, row 2: SNR = 5 dB and 6 dB from left to right, row 3: SNR = 7 dB and 8 dB from left to right.

6. Conclusions

In this study, a mobile robot platform and a real-time lane detection algorithm were developed. The algorithm was implemented on a DSP platform and optimized in terms of computational speed to make it suitable for real-time applications. After several optimization schemes, the proposed lane detection algorithm was able to process 30 frames/s, which should be satisfactory for a vehicle travelling at 120 km/h. The developed algorithm was tested under various conditions and the results proved the algorithm to be robust against several visual disturbances.

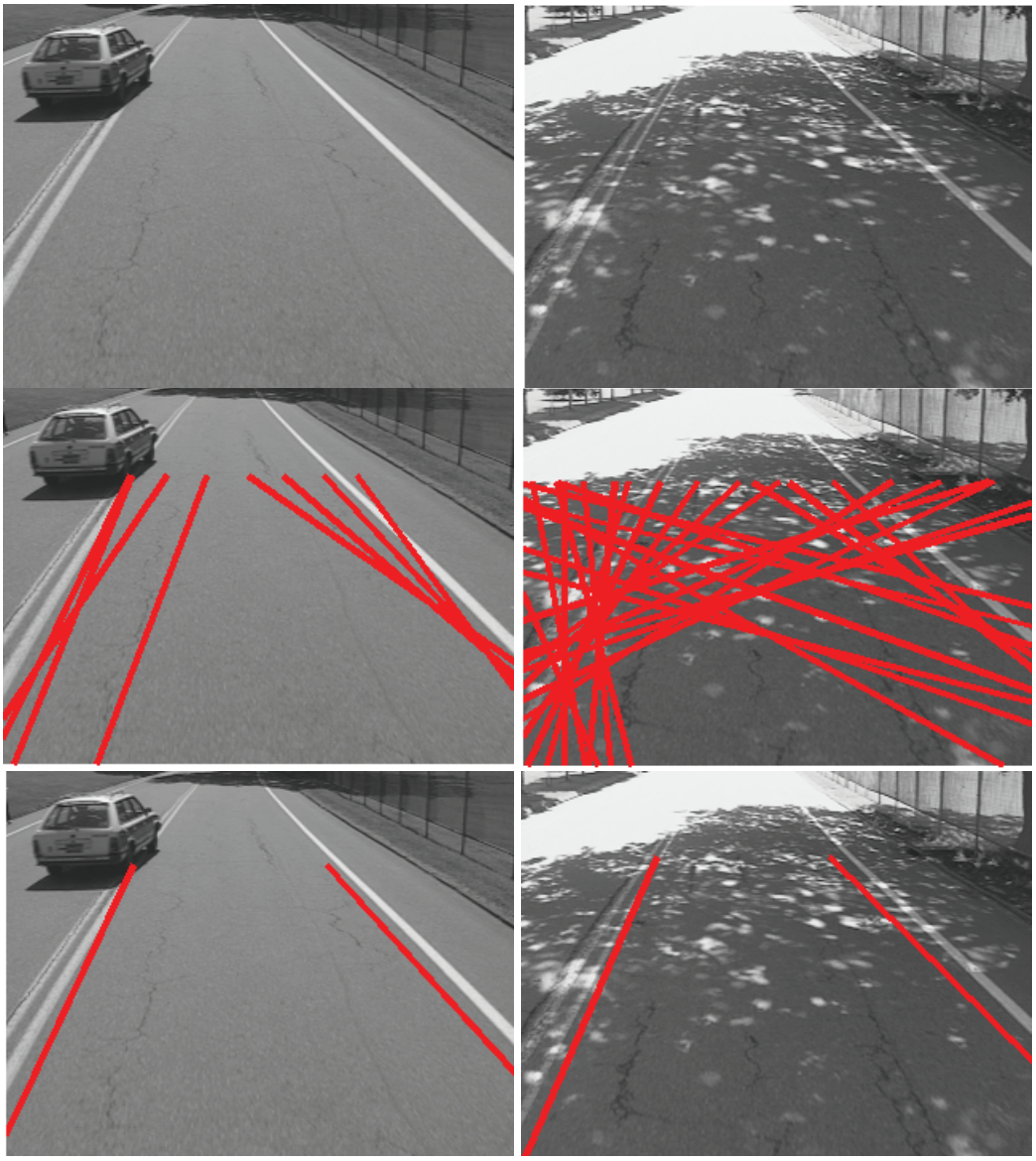

Figure 13. Row 1: The 2 original real-world images, row 2: detected Hough lines, row 3: detected lane marks.

This study differs from most of the lane detection algorithms presented in the literature because the proposed image processing algorithm was implemented in a DSP-based platform rather than on a PC. The study is a good example of the development and optimization of an image processing algorithm with DSP boards using low-level programming languages.

One of the limitations of the developed system is the frame rate of the camera used, which limits the number of frames the proposed method processes per second. An analog NTSC format camera is used in the study, which could deliver only 30 frames/s. Using a digital camera with higher frame rates, the number of frames processed per second could have possibly been increased. In addition, the mobile robot platform could be driven at a maximum speed of 4 km/h because of the DC motor's technical properties. A more powerful platform could have been built for testing at higher speeds.

References

[1] Wikipedia, Autonomous car, http://en.wikipedia.org/wiki/Autonomous_car, last accessed 22 November 2011.

[2] M. Ekinci, W.J.F. Gibbs, B.T. Thomas, "Knowledge-based navigation for autonomous road vehicles", Turkish Journal of Electrical Engineering & Computer Sciences, Vol. 8, pp. 1–29, 2000.

[3] A. Saudi, J. Teo, M.H.A. Hijazi, J. Sulaiman, "Fast lane detection with randomized Hough transform", International Symposium on Information Technology, Vol. 4, pp. 1–5, 2008.

[4] A. Borkar, M. Hayes, M.T. Smith, "Polar randomized Hough transform for lane detection using loose constraints of parallel lines", IEEE International Conference on Acoustics, pp. 1037–1040, 2011.

[5] L. Qing, Z. Nanning, C. Hong, "Lane boundary detection using an adaptive randomized Hough transform", 5th World Congress on Intelligent Control and Automation, Vol. 5, pp. 4084–4088, 2004.

[6] S. Lee, H. Son, K. Min, "Implementation of lane detection system using optimized Hough transform circuit", IEEE Pacific Conference on Circuits and Systems, pp. 406–409, 2010.

[7] J. Wang, Z. Liang, Y. Xi, "Lane detection based on random Hough transform on region of interesting", IEEE International Conference on Information and Automation, Vol. 4, pp. 1735–1740, 2010.

[8] Y. Wang, E.K. Teoh, D. Shen, "Lane detection and tracking using B-snake", Image and Vision Computing, Vol. 22, pp. 269–280, 2004.

[9] S.P. Adhikari, H. Kim, "Dynamic programming and curve fitting based road boundary detection", 9th WSEAS International Conference on Computational Intelligence, pp. 236–240, 2010.

[10] D.J. Kang, M.H. Jung, "Road lane segmentation using dynamic programming for active safety vehicles", Journal of Pattern Reorganization Letters, Vol. 24, pp. 3177–3185, 2003.

[11] A. Borkar, M. Hayes, M.T. Smith, "Robust lane detectıon and tracking with RANSAC and Kalman filter", IEEE International Conference on Image Processing, Vol. 7, pp. 3261–3264, 2009.

[12] T. Suttorp, T. Bucher, "Learning of Kalman filter parameters for lane detection", IEEE Intelligent Vehicles Symposium, pp. 552–557, 2006.

[13] N. Apostoloff, A. Zelinsky, "Robot vision based lane tracking using multiple cues and particle filtering", IEEE Intelligent Vehicles Symposium, pp. 558–563, 2003.

[14] H.M. Mandalia, D.D.M. Salvucci, "Using support vector machines for lane-change detection", Proceedings of the Human Factors and Ergonomics Society Annual Meeting, Vol. 49, pp. 1965–1969, 2005.

[15] M.D. Jochem, D.A. Pomerleau, E.C. Thorpe, MANIAC: A Next Generation Neurally Based Autonomous Road Follower, The Robotics Institute, Carnegie Mellon University, Pittsburg, PA, USA.

[16] K. Kluge, S. Lakshmanan, "A deformable-template approach to lane detection", Proceedings of the 95th Symposium on Intelligent Vehicles, pp. 54–59, 1995.

[17] T.Y. Sun, S.J. Tsai, V. Chan, "HSI color model based lane-marking detection", IEEE Intelligent Transportation Systems Conference, pp. 1168–1172, 2006.

[18] D.H. Ballard, "Generalizing the Hough transform to detect arbitrary shapes", Pattern Recognition, Vol. 13, pp. 111–122, 1981.

[19] D. Ioannou, W. Huda, A.F. Laine, "Circle recognition through a 2D Hough transform and radius histogramming", Journal of Image and Vision Computing, Vol. 17, pp. 15–26, 1999.

[20] T.C. Chen, K.L. Chung, "An efficient randomized algorithm for detecting circles", Journal of Computer Vision and Image Understanding, Vol. 83, pp. 172–191, 2001.

[21] S.Y. Guo, X.F. Zhang, F. Zhang, "Adaptive randomized Hough transform for circle detection using moving window", International Conference on Machine Learning and Cybernetics, pp. 3380–3385, 2006.

[22] A. Goneid, S. El-Gindi, A. Sewisy, "A method for the Hough transform detection of circles and ellipses using a 1-dimensional array", IEEE International Conference on Cybernetics and Simulation, Vol. 4, pp. 3154–3157, 1997.

[23] A.C. Gürbüz, "Line detection with adaptive random samples", Turkish Journal of Electrical Engineering & Computer Sciences, Vol. 19, pp. 21–32, 2011.

[24] G. Küçükyıldız, H. Ocak, "Development and optimization of DSP based real time lane detection algorithm on a mobile robot platform", IEEE Signal Processing and Communications Applications Conference, pp. 1–4, 2012 (article in Turkish with abstract in English).

[25] M. Bertozzi, A. Broggi, A. Fascioli, "Stereo inverse perspective mapping: theory and applications", Journal of Image and Vision Computing, Vol. 16, pp. 585–590, 2007.

[26] H.A. Malot, H.H. Bultoff, J.J. Little, S. Bohrer, "Inverse perspective mapping simplifies optical flow computation and obstacle detection", Journal of Biological Cybernetics, Vol. 64, pp. 177–185, 1992.

[27] G.Y. Jiang, T.Y. Choi, S.K. Hong, J.W. Bae, B.S. Song, "Lane and obstacle detection based on fast inverse perspective mapping algorithm", IEEE Transactions on Cybernetics, Vol. 4, pp. 2969–2974, 2000.

[28] E. İnce, "Measuring traffic flow and classifying vehicle types: a surveillance video based approach", Turkish Journal of Electrical Engineering & Computer Sciences, Vol. 19, pp. 607–620, 2011.

[29] Gonzalez RC, Woods RE, Digital Image Processing, 3rd ed., Upper Saddle River, NJ, USA, Prentice Hall, pp. 142–147, 2007.

[30] Vision and Autonomous Systems Center, Carnegie Mellon University, Sample Database, http://vasc.ri.cmu.edu/idb/html/road/index.html, last accessed 30 October 2012.