Computer Vision
Short Project. Design and implementation of the Hough transform to guide vehicles on a road
Report

Carlos Trapiello Fernández
Juan Gallostra Acín

December 2016

1 Introduction

Computer vision is a key technology for smart vehicles, which seems to be the tendency that the big companies are focus on develop nowadays. One of its main applications, no matter if you are searching the driver's assistance or the autonomous driving, is the identification and tracking of lane lines.

The perfect solution for the lane detection has not been found so far, one way to face this problem is using the Hough transform which allows to isolate features of a particular shape within an image. However Hough transform is instable in complex cases and hard to know whether the detected lines belong to the same lane lines.

The aim of our project is to implement a good performance lane detection system based on the Hough transform, trying to solve all the inherent difficulties that the problem have. The lane detection method presented in this paper processes an incoming live video stream frame by frame and extracts the position of lane markings.

Each frame is processed in different subsequent steps. First, information of the lane markings is amplified and extracted from the frame in a pre-processing stage. After that, Hough transform is applied in order to detect the lane markings as straight lines in the lane detection stage. Finally, the possible solutions obtained for the current frame are compared with the solution chosen in the previous frames of the video stream in the lane tracking stage.

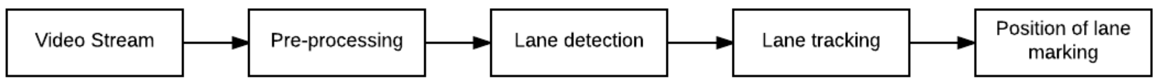


Figure 1.1: Flow of the lane detection method presented in this project.

2 Image pre-processing

The pre-processing stage have two main aims, to remove the inevitable noise during image acquisition and to extract from the frame information on the position of the lane markings. In order to fulfil this requirements we present the following procedure:

1. **Grayscaling**: The image is transformed to a grayscale space.
2. **ROI selection**: A Region of interest (ROI) is defined and only this region is further studied.
3. **De-noising**: Noise is removed applying a median filter.
4. **Binarization**: Selecting an optimal threshold the ROI is binarized to a black and white image.
5. **Edge detection**: A Sobel operator is used to detect the edges in the image.

Now we are going to explain in detail in which consist each of the steps defined.

2.1 Grayscaling

A raw image from a camera is provided in the RGB colour format. In this format each pixel is assigned three colour channels, one for red, one for green and one for blue. The values of the three channels are combined to yield the actual colour of the pixel.

In the RGB colour format the distinction of lane markings from their environment is really challenging. It is more effective to make use of the characteristic that lane markings usually are brighter than the road they are printed on. From the different methods existing to transform an image from RGB to the grayscale format, all of them based on weighted summation, we decided to directly use the rgb2gray Matlab function. This function converts RGB values to grayscale by forming a weighted sum of the three channels with the following weights

$$I = 0.2989 \cdot R + 0.5870 \cdot G + 0.1140 \cdot B$$

The resulting grayscaled image is showed in Figure 2.1 (b).

2.2 Region of interest

The figure 2.1 (a) shows an image taken from within the car. Only the lower part of the image actually have relevant information for the lane detection problem. The part shown in Figure 2.1 (c) will be called the region of interest (ROI).

The ROI contains all essential information for the subsequent lane detection steps, so from now on, the rest of the frame will be discarded. The size of the ROI is a limiting parameter that determines computational speed and effort of the lane detection. The smaller the ROI is chosen, the faster the lane detection performs.

As we will approximate the lane markings with a straight line assumption, this works well for the lanes within the ROI on straight roads and even in moderate bends. However for hard bends, this assumption do not hold, so for this case we decided to split horizontally the ROI in two regions as shown in Figure 2.2.

2.3 De-noising

As we said before, image noise is inevitable during the image acquisition. Aiming to enhance the image we use a median filtering. Median filtering is a non-linear filter, which has a good effect on impulse noise and image scanning noise, and also bring less image blur caused be linear filters.

We choose a $3x3$ filter template taking the balance of effectiveness and efficiency into consideration. The result of the de-noising process is shown in Figure 2.1 (d).

2.4 Binarization

Before the edge-extraction process is fundamental to remove all the possible edge disturbances that might appear in the image. Some weak sources of edges, like the signal post at the side of the road, varying colours on the street material, shadows ... can be deleted from the image by thresholding.

The intensity of all pixels, whose value falls below a certain threshold, are set to zero. As lane markings are painted in white colour, they will not be affected negatively.

The main problem is the selection of the right threshold, we have to define it in such a way that we don't send to zero relevant information but also trying to remove all the possible edges disturbances.

For computing the threshold we have decided to use an adaptative threshold selection which adapts better to the different situations in the road due to the use of first order statistics approximations. What this method actually does is compute the local threshold for each pixel.

The binarized image is shown in figure 2.1 (e).

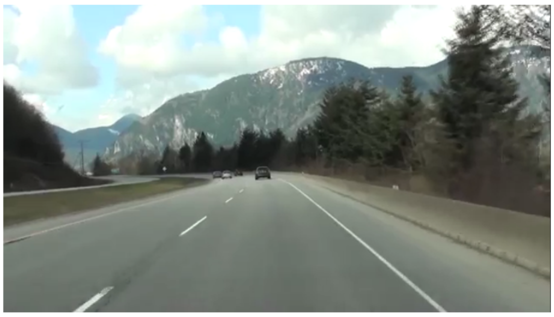
2.5 Edge detection

A Sobel filter is applied to the binarized image producing a new image where only the edges of the existing elements are present. More technically, each pixel in the new image describes the gradient of the original image at that position. As the image has been previously binarized there exists strong gradients that perfectly define the edges of the elements.

The gradient is calculated applying two discrete differentiation operators to the image, one for the horizontal and one for the vertical direction. The operators, which include smoothing and have the form of two 3x3 kernels, have the following form:

$$G_x = \begin{bmatrix} -1 & 0 & +1 \\ -2 & 0 & +2 \\ -1 & 0 & +1 \end{bmatrix} \qquad G_y = \begin{bmatrix} -1 & -2 & -1 \\ 0 & 0 & 0 \\ +1 & +2 & +1 \end{bmatrix}$$

The formulas show that 9 pixels are required to determine the gradients at a position. G_x and G_y are then combined to yield the overall gradient. The Figure 2.1 (f), illustrates the output of the Sobel filter.



(a) Original image

(b) Grayscaled image

(c) Region of interest

(d) De-noised ROI

(e) Binarized ROI

(f) Output of the Sobel filter

Figure 2.1: Original image and output of the different pre-processing stages.

(a) Region of interest

(b) Upper ROI

(c) Lower ROI

Figure 2.2: Split ROI into two sub-regions.

3 Lane detection

Once the pre-processing of the image has finished we are able to detect the lane markings. As we said before, depending if we are on a straight line or if we are on a bend, the straight line assumption works or not.

3.1 Selection of the best ROI

In order to deal with these errors, we decided to take a closer approach whenever we are detecting a curve, and split horizontally the ROI into another two sub-regions in such a way we try to modelize the curve as two straight lines.

So the first step in the lane detection stage is decide if for the current frame we are located on a curve of the road or on a straight part. To do that, we use the information extracted from the lanes detected in the previous frame.

We have implemented the following conditions:

- If the angle of the right lane detected in the previous frame is between a certain degrees interval dependant on the allocation of the camera we assume that in the current frame we are on a straight part of the road and take as ROI the one shown in Figure 2.1 (c).
- If not, we make the assumption that we are driving through a curve, and the original ROI is split in two sub-regions as shown in Figure 2.2.
- In order to initialize the study of the first frame, we always suppose that we start on a straight line (one ROI).

When we split the ROI in two, instead of one straight line, we will obtain two, so we have to implement an extra condition to be able to detect when the curve has finished and we are again on a straight part of the road.

- When we detect two lines if the angle difference among them is lower than 20 degrees we approximate it as a unique line so for the next frame we will be able to discern among the previous conditions. If the difference of angles is greater than 20 degrees for the next frame we assume we are still in the curve (2 ROI).

The selection of the parameter angles, as well as the limits of the regions of interest, depends on the collocation of the camera inside the car. For this particular case, we have suppose fixed camera inside the car as shown in the different figures.

3.2 Hough transform

Independently that we are analysing one ROI or the two sub-regions presented before, the procedure of lane detection through Hough transform is the same.

Hough transform was proposed by Hough in 1962, however it was Duda and Hart who combined Hough transform and polar coordinates, so the problem when the straight line is vertical (infinite slope) is avoided. The generalised Hough transform formula is:

$$\rho = x \cdot cos\theta + y \cdot sin\theta \qquad \theta \in [-90, 90]$$

Where:

- (x, y) are the spatial coordinates.
- (ρ, θ) are the polar coordinates.

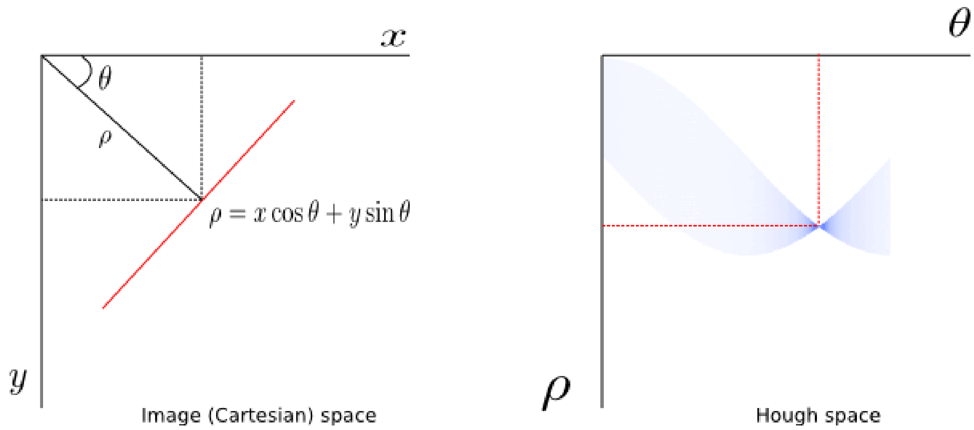


Figure 3.1: Visual representation of the Hough transform.

Applying the Hough transform to the pre-processed image, we are able to obtain a matrix that contains the information of the number of points detected in the Hough space for each combination of (ρ, θ).

Each point in the image space, represent a curve in the Hough space, also a curve defined by two points in the image scape, is represented as the intersection point of the two respective curves in the Hough space. So, wherever we detect a high density of curve intersection in the Hough transform it means that there is a straight line in the image space (all the points in the straight line will intersect in the same (ρ, θ) values).

As we are not interested in the horizontal lines we can remove (send to zero) all the detected points in the Hough space for angles greater than 70 and lower than -70, by doing this we eliminate the disturbances created by the detected horizontal lines.

At this point, we need to decide which of the detected density peaks we are going to use as lane markings candidates. As said before, only the high density peaks in the Hough space represent a straight line in the image space, so the straight lines selected are those from the detected ones that have a density of at least one fifth the maximum peak density. In case of detecting a huge number of peaks that fulfil that constraint only the 20 with the higher density will be chosen.

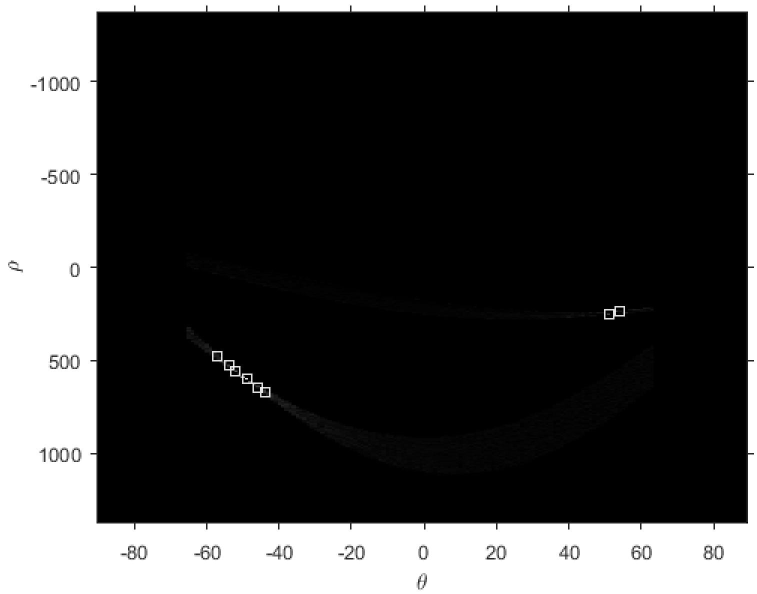
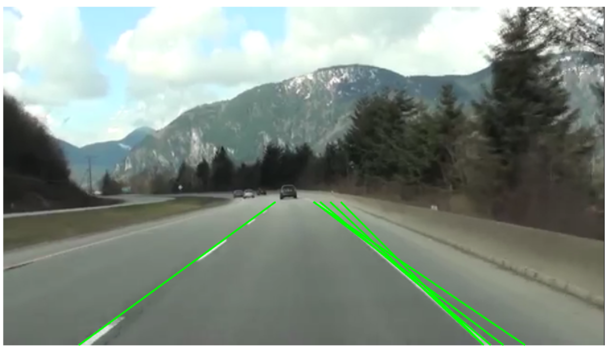


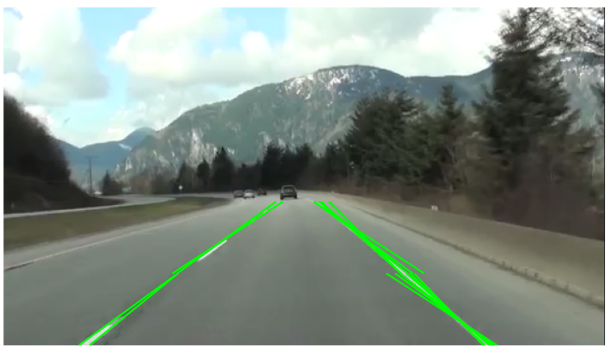
Figure 3.2: Visual representation of the peaks obtained after applying the Hough transform.

Each frame we generate the set of lines that the Hough transform, with the parameters previously defined, recognizes as straight lines in the pre-processed image. As shown in Figure 3.3 this is not accurate because there are several factors such as noise, shadows or other objects in the image with sharp edges that lead to detect other straight lines apart from the actual lane mark that should be found with the straight line assumption.

Once we have generated the whole set of candidates, in order to fit the lane marking, we will use the information of previous frames line selection, and the assumption that the lane detected in the present frame must look similar to the lane detected in the previous frame. The selection of the optimal line is processed in the lane tracking stage.



(a) Candidate lines when having one ROI.



(b) Candidate lines when having two ROIs.

Figure 3.3: Candidate lines.

4 Lane tracking

As previously explained, the main idea behind the lane mark tracking system is to help in the decision of selecting which line should be chosen as the line that best represents the lane mark for a given frame k.

This is done by keeping a record of which was the line selected as the lane mark on frame $k-1$ and assuming this line is actually the real lane mark. This information is then used to predict where will be the lane mark of frame k. Once the prediction of the lane mark position of frame k is computed then it is checked which is the line among all the obtained candidate lines of frame k that best fits the prediction. The set of candidate lines is the set of lines obtained after performing the Hough transform on the frame.

This line that best ressembles the predicted lane mark is then selected as the actual lane mark of frame k and will be used to predict the lane mark position of frame $k+1$. Since, as already stated, we are using the straight lane mark assumption it has been decided to predict the position of the lane mark on frame k with the line that was selected as the lane mark on frame $k-1$.

Once knowing which is the predicted lane mark of frame k the best fit for that same frame is calculated by

$$bl(k) = \min_{l \in L_k} \{(x_l^U - x_{bl(k-1)}^U)^2 + (x_l^D - x_{bl(k-1)}^D)^2\}$$

Where:

- $bl(k)$ and L_k are respectively, the best fit line and the set of candidate lines for frame k
- $bl(k-1)$ is the best fit line for frame $k-1$ and $x_{bl(k-1)}^U$ and $x_{bl(k-1)}^D$ are the x coordinate values of the points of $bl(k-1)$ located at the top (U) and bottom (D) of the ROI
- x_l^U and x_l^D are similarly the x coordinate values of the points of $l \in L_k$ located at the top (U) and bottom (D) of the ROI

The line that satisfies the above condition will be selected then as the line which best represents the lane track.

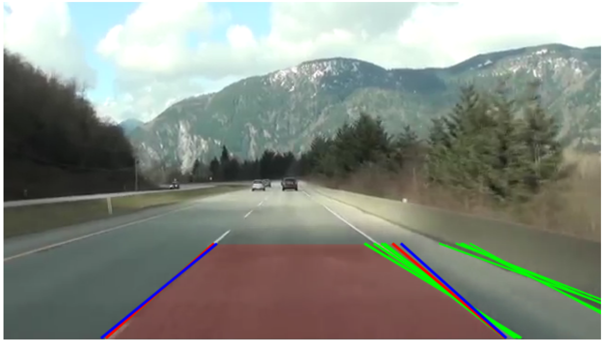
It has to be pointed out here that, since there are two lane marks to be detected (the left lane mark and the right lane mark) and the Hough transforms returns the set of all detected lines in the frame without differentiating them there is a previous step that has to be done before applying the lane tracking algorithm. It consits of filtering, from the set of lines, the lines that correspond to left lane mark candidates and the ones that correspond to right lane candidates. The filtering method that we have applied is, for each candidate line $l_i(k) \in L_k \;\; i = 1, 2, ..., |L_k|$

$$\begin{cases} x_{l_i(k)}^U > x_{l_i(k)}^D \rightarrow l_i(k) \in \;\; left \;\; lines \\ x_{l_i(k)}^U < x_{l_i(k)}^D \rightarrow l_i(k) \in \;\; right \;\; lines \end{cases}$$

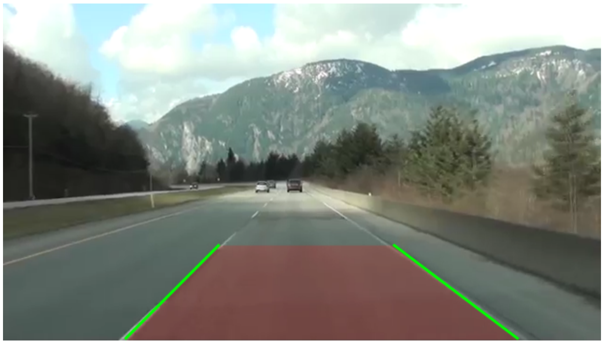
Where the notation used is the same as the one used in the expression that computes the best fit line.

Note that by filtering the lines into left lane mark candidates and right lane mark candidates in this manner all the vertical lines are discarded.

(a) Candidate lines (green) and best fit (red)



(b) Candidate lines (green), best fit (red) and previous selected line (blue)



(c) Final selected lines for a given frame (green)

Figure 4.1: Line selection with the implemented line tracking.

5 Results

Although we are able to detect the lane markings of the road on a huge range of situations, we are still hard dependent of environmental factors that difficult the detection and lead us to negative results.

We have found that the right selection of the threshold is one of the most limiting factors. Despite its power, Hough transform is only able to detect right the lane markings if their edges are well defined. For those frames that a bad pre-processing of the image makes us lose information of the lane markings, Hough transform will introduce failures. For a better performance of this method, it is recommended to implement a variable threshold which is able to modify its value for the different environmental situations we can face, such a shadow zones, night driving...

The other main problem encountered, is that whenever we lose the actual lane marking for another line detected in the frame, as we use the assumption for the tracking that we look for the line that fits best the one selected in the previous frame, we induce a serial selection error which will not be fixed until the inverse situation occurs.

In order to address this problem, we could have used other assumptions like using the mean line detected in the previous frames... However all the assumptions have its own pros and cons, that's why we finally choose the previous mentioned assumption as the used one.

We conclude that lane detection only using Hough transform is a poor method by itself, and it should be combined with other techniques like particle filters in order to obtain a robust performance on the road.

References

[1] Matlab documentation. Online, December 2016. https://es.mathworks.com.

[2] Jan Botsch. Real-time lane detection and tracking on high performance computing devices. Bachelor's thesis, Technische Universitat Munchen, March 2015.

[3] Joan Aranda López and Albert Sanfeliu Cortés. Lecture notes. Online, Decembre 2016. Lecture notes from the course on Computer Vision of the Master's degree in Automatics and Robotics at UPC.

[4] He Mao and Mei Xie. Lane detection based on hough transform and endpoints classification. Technical report, University of Electronic Science And Technology of China, 02 2012.