Lane Detection and Trajectory Generation System

Manuel Díaz-Zapata[2], Jose Miguel Correa-Sandoval[1], Juan Perafán-Villota[1, γ], Víctor Romero-Cano[1,2]*

1. Departamento de Energética y Mecánica, Universidad Autónoma de Occidente, Santiago de Cali, Colombia
2. CHROMA, French Institute for Research in Computer Science and Automation (Inria), Grenoble, France

* The first author undertook this work while he was part of Universidad Autónoma de Occidente
γ. Email: jcperafan@uao.edu.co

Abstract

This paper presents the development of a perception system that enables an Ackermann-type autonomous vehicle to move through urban environments using control commands based on short-term trajectory planning.

We propose a lane detection and keeping system based on computer vision techniques that are computationally efficient. Also, a Kalman filter-based estimation module was added to gain robustness against illumination changes and shadows.

Additionally, the simulation and control of the Autónomo Uno robot gave good results following the steering commands to keep the position. In the simulation the controllers had some slight noise problems but the robot executed the given steering commands and it moved following the road. This behavior was also seen in the physical implementation.

Keywords: autonomous vehicle, computer vision, lane detection, lane keeping, Ackermann kinematic model

Related work, Motivation and Objective.

Since the late 20th century there have been studies about systems focused on lowering the amount of lives lost due to traffic accidents, improving road safety and increase the autonomy of the vehicle. These systems are known as Advance Driving Assistance Systems (ADAS). One of the main features of ADAS is lane detection and tracking, which it is a task where real-time processing is most crucial, since this and the steering actuation system are the ones controlling where the vehicle is headed with respect to the vehicle's position in the lane and its surroundings.

In the study presented by Naronte et. al [1], is shown that there have been different advances on the development of this techniques. On 2015 Son et. al [2] present different modules working together to do lane detection, some of these modules include color invariant lane detection when exposed to illumination changes, adaptive region of interest (ROI) based on the vanishing point, Canny edge detection, and lane grouping using minimum squares. Using these modules on a simulated environment, they got a fast and powerful result for real-time application under different weather and lighting conditions. But roads with cracks and blurred lane markings were this system's weak spots.

A year later, Madrid and Hurtik [3] presented a low-cost warning departure system for vehicles, which is efficient and affordable for most people. They use the Hough transform for line detection together with a fuzzy image representation, which gave better results than the standard

Hough transform with the Sobel operator. It also presented a reliable and accurate response to real-time processing. This system had difficulties during the night and when white vehicles enter the scene.

Mammeri et al. [4] on 2016 presented an internal computing system that finds the lane markings and passes them to the driver. For the line detection they used Probabilistic Progressive Hough Transform, ROIs through MSER blobs were also used, which are then refined using a three-stage algorithm. Using the MSER results, the lane colors (white and yellow) are identified on the HSV color-space and Kalman filtering is used to follow both lane lines. This system is limited at night due to the road lighting and traffic conditions.

On the other hand, there has been little interest in Colombia to develop these kinds of technologies due to the economic and infrastructure difficulties present in the country. This lack of interest can be seen on the low adoption and acceptance of using technologies associated with renewable energies in the transport sector such as electric autonomous vehicles. Nevertheless, studies and work done in this area in world powers give a hopeful future that could bring many benefits to the Colombian territory [5].

These benefits are the main motivation to create a perception system for vehicles that facilitates lane detection and trajectory generation on urban environments. Hoping that the creation of software for platforms that want to develop autonomous vehicles, will promote research and development on this area in a national level.

Methods.

This work was developed in three stages, first the Lane Detection system, second the Trajectory Generation system and finally the system was tested in the Autónomo Uno robot and at the same time in a simulation environment in Gazebo. For the development of the **Lane Detection system**, we mainly used videos [6] that were converted from mp4 format, to rosbag so they could be better integrated with ROS, since this environment allows easier integration with robot control hardware.

Perspective transform was used to create the bird's eye view (BEV) of the road, based on a fixed ROI in order to avoid the vanishing point effect. Besides, the BEV view provides better information of the road which it helps to create a more accurate lane model. Using this transformation, it is easier to find the center of the lane and the error for the car to steer, so it can stay in the center of the lane.

We predefined the ROI, since our own automatic ROI finder was too inefficient working at an average of 2 FPS on a 30 FPS video. Once the BEV is obtained, the following computer vision techniques were used to enhance the lane markings and filter unwanted data: grayscale conversion and vertical Haar-like feature filter [7].

After, the image is divided into two parts, one for each of the lane markings. Then, each image's histogram gets equalized and a median filter is applied along with pixel-wise gamma correction using γ=20. These steps are done in order to enhance the brighter colors present on the lane markings, so it is easier to apply a threshold to binarize the image. Furthermore, image binary thresholding was performed on the value range from 240 to 250, where pixel values go from 0 or black to 255 or white, greater values were not selected due to included noise.

The Lane model is created by passing a set of sliding windows vertically through each of the lane markings and finding a set of centroids that describe each marking. The parameters used for the

sliding windows are their height (W_h), width (W_w), number of windows (N_w) and horizontal starting point of search (S_h). The W_w, N_w and S_h are defined by us, but the height of each window is found dividing the image height (h) by N_w. The centroids for each marking were found by averaging the values of the x coordinates for the with pixels in the window's area. The search process is repeated until N_w windows have been placed.

Kalman Filtering [8] was used to filter the changes on the x coordinate of each centroid or to predict the position if no white pixels are found due to broken lines. Also, Kalman filtering is not applied on the y coordinate since it increases by a fixed known value W_h on each iteration. Finally, the center line was found by computing the difference between centroids of each lane model.

The **Trajectory Generation system** is based on the middle lane provided by the lane detection system. First two points of the middle line are selected for the steering vector, then the steering angle is calculated as shown in eq. 1. pt_1 is the closet point to the car and pt_2 the sixth point of the middle line, also the index x and y indicate the coordinates of the points.

$$\theta = arcsin\left(\frac{pt_{1x} - pt_{2x}}{\sqrt{(pt_{2y} - pt_{1y})^2 + (pt_{2x} - pt_{1x})^2}}\right) \quad (1)$$

The steering angle is used along with the selected points to create the steering vector. This vector describes the short-term trajectory that the vehicle needs to follow in order to stay in the middle of its lane. Furthermore, the steering angle is used to compute the inverse kinematics of mobile robots, thus the Ackermann-type vehicle is able to move the steering wheels to perform the trajectory.

Finally, the system was integrated with the physical platform Autónomo Uno robot, also with the simulated environment using ROS along with Gazebo. The inverse kinematics of mobile robots [9] were used to control the Ackermann-type vehicle, in addition, a Proportional—Integral (PI) controller system was added to control the response given by the vehicle while it follows the steering vector provided by the lane detection system.

One of the main inputs to our simulation system is the robot's heading, which is provided by the vision-based trajectory planning module. This heading value is translated into steer commands for each of the front wheels using the instantaneous center of rotation's (ICR) distance and the ICR's angle, those are provided by the inverse kinematics. The ICR distance is computed as shown in the eq 2.

$$ICR_{distance} = \frac{length}{\tan(\theta)} \quad (2)$$

Results.

The results of the **Lane Detection system** and **Trajectory Generation system** can be observed on figure 1. Where on figure 1a the ROI we defined can be seen delimited in red. Then, on figure 1b, we can see the BEV view and the lane markings after the filtering. The final process to enhance the lane was performed and the result can be seen on figure on 1c. After, the sliding windows step is done and the centroid-based lane model is created, see figure 1d, where the

centroids in red are those detected by the sliding windows, but the ones in blue or green are the ones predicted by the Kalman filter since the observation was not confident enough.

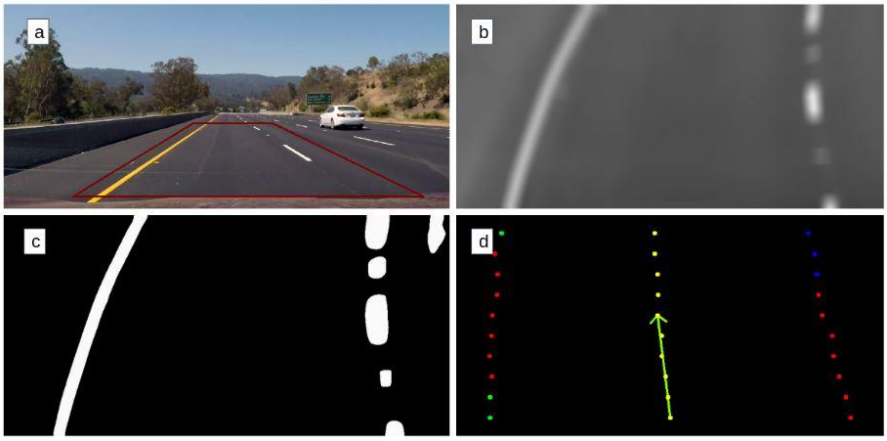

Fig. 1 a. Manual ROI defined, **b.** Filtered perspective transform of a road curve, **c.** Binary image after thresholding, **d.** Lane centroids and steering vector

The center line model can be observed on figure 1d, along with the steering vector. The behavior of this steering vector, can be tuned by choosing a different point as pt_2. We found that selecting pt_2 too close to the car, makes the system too sensitive to variations in the centroid's x coordinate. Also, by choosing pt_2 closest to the top side of the image reduces its sensitivity. The sixth point was chosen because it gave stability and good angle representation in both cases: straight and curve roads.

The Ackermann Kinematics model defined in [9] is computed locally in the Autónomo Uno robot, using the two main micro controllers installed. The first one is the Raspberry Pi 3 B+, that allows communication with a main computer in order to receive control commands. The second is an Arduino DUE, that controls two different types of motors used for traction and direction respectively. Also, each motor has its own built-in PID controller that manages its respective behavior.

On the other hand, the Ackermann Kinematics model needed for the Gazebo simulation environment are computed on the main computer. Also, the dynamics involved in the environment requires a controller module, see figure 2 (left). Here the red and pink signals are respectively the front wheels response to the steer angle.

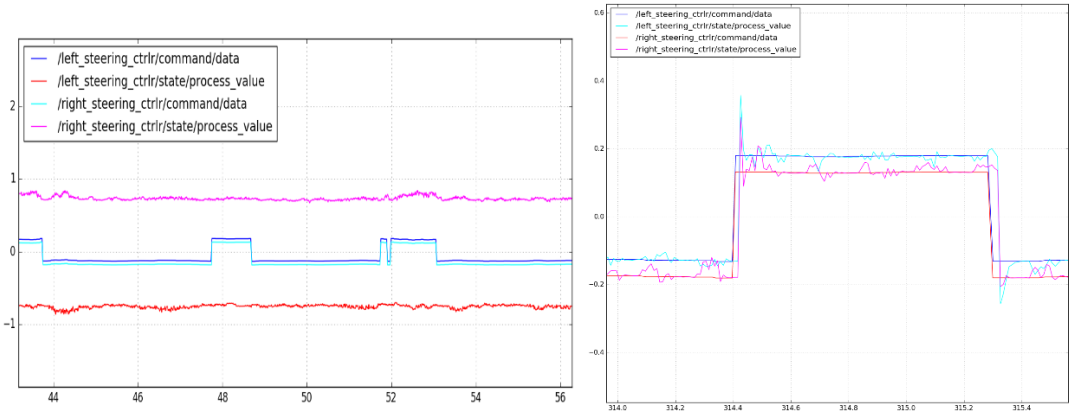

Fig. 2. **Left,** Initial response state of the system without controller. **Right,** Ackermann system response with PI controller

Therefore, a PI controller was created using the Ziegler-Nichols tuning method [10] for the steering and velocity of the wheels. Even though the signals with the controller have noise, figure 2 (right), the system manages to follow the set-point of the steering angle. This behavior was seen on the simulation environment and in the physical platform. On figure 3, the system's response is shown for three principal cases, turn left, turn right and go straight.

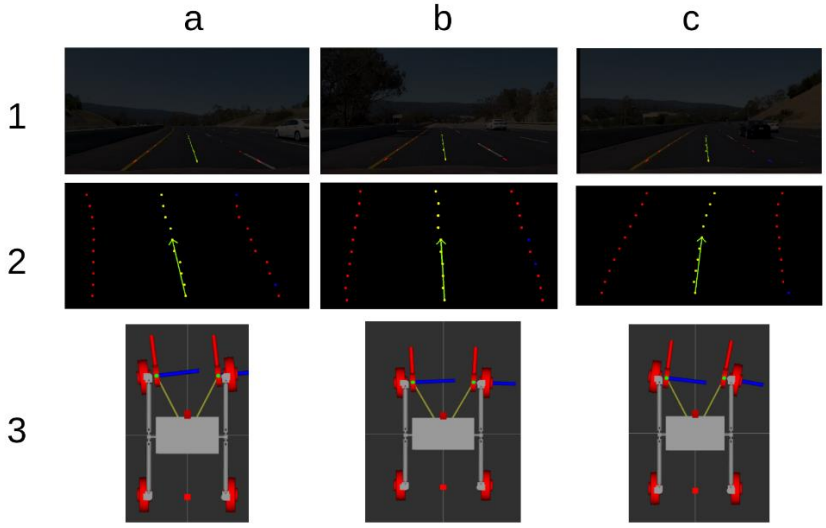

Fig. 3. Column a. System behavior for left curve road, Column b. System behavior for straight road, Column c. System behavior for right curve road

Discussion and Conclusion.

We were able to create a lane keeping algorithm for autonomous vehicles based on traditional computer vision techniques, that is robust to illumination changes and manages to detect and track lanes under different conditions such as: lane markings with two different colors (yellow and white), lane markings exposed to direct sunlight, shadows and transitions between asphalts of different color. Also, the system can react to the to the two main lane types, straight and curved. Furthermore, the system controls the vehicle's heading, based on a short-term trajectory

generation and the Ackermann Kinematic model so the vehicle stays in the center of the lane, which is an added capability compared to Lane Departure warning systems presented in [2], [3] and [4].

We also highlight the use of sliding windows versus the Hough Line transform and its variants used in [2], [3] and [4] since we are able to model with a fixed number of sliding windows each lane marking by representing them as series of points. The main problem with the use of the Hough Line Transform is the detection of only straight lines. Thus, to detect curved features the system needs to either detect small segments of the curved line and piece them together to create the lane model, or it needs to do a rough approximation of a curve with a straight line, which could not be an optimal model.

For future work, we recommend to do a comparative analysis between our method and the ones presented on previous work, focused on the computational cost and real-time performance. Also, given the current trend on the Computer Vision field, it would be interesting to see the Deep Learning-based approaches to both the perception and decision modules.

References

[1] Narote, S.P., Bhujbal, P.N., Narote, A.S., Dhane, D.M.: A review of recent advances in lane detection and departure warning system. Pattern Recognition 73, 216–234 (2018). DOI 10.1016/j.patcog.2017.08.014.
[2] Son, J., Yoo, H., Kim, S., Sohn, K.: Real-time illumination invariant lane detection for lane departure warning system. Expert Systems with Applications 42 (4), 1816–1824 (2015). DOI 10.1016/j.eswa.2014.10.024. URL http://dx.doi.org/10.1016/j.eswa.2014.10.024
[3] Madrid, N., Hurtik, P.: Lane departure warning for mobile devices based on a fuzzy representation of images. Fuzzy Sets and Systems 291, 144–159 (2016). DOI 10.1016/j.fss.2015.09.009. URL http://dx.doi.org/10.1016/j.fss.2015.09.009
[4] Mammeri, A., Boukerche, A., Tang, Z.: A real-time lane marking localization, tracking and communication system. Computer Communications 73, 132–143 (2016). DOI 10.1016/j.comcom.2015.08.010. URL http://dx.doi.org/10.1016/j.comcom.2015.08.010
[5] Ackerman, E.: Study: Intelligent Cars Could Boost Highway Capacity by 273% (2012). URL https://spectrum.ieee.org/automaton/robotics/artificial-intelligence/intelligent-cars-could-boost-highway-capacity-by-273
[6] Udacity: Lane finding project for self-driving car nd. https://github.com/udacity/CarND-LaneLines-P1 (2017)
[7] Viola, P., Jones, M.: Rapid object detection using a boosted cascade of simple features. Proceedings of the 2001 IEEE Computer Society Conference on Computer Vision and Pattern Recognition. CVPR 2001 1, I–511–I–518 (2001). DOI 10.1109/CVPR.2001.990517.URL http://ieeexplore.ieee.org/document/990517/
[8] Kalman, R.E.: A New Approach to Linear Filtering and Prediction Problems. Transactions of the ASME–Journal of Basic Engineering 82 (Series D), 35–45 (1960). URL https://www.cs.unc.edu/ welch/kalman/media/pdf/Kalman1960.pd
[9] Saavedra Ruiz, M.A., Pinto Vargas, A.M.: Desarrollo de un sistema de aterrizaje autónomo para un vehículo aéreo no tripulado sobre un vehículo terrestre. Universidad Autónoma de Occidente pp. 43–47 (2019). URL http://hdl.handle.net/10614/10754
[10] Ogata, K.: Modern control engineering. IEEE (1972). DOI 10.1109/TAC.1972.1100013. URL http://ieeexplore.ieee.org/document/1100013