

# **Project Report: Signal processing on Autonomous Car using radar and camera**



Name: Alec Mabhiza Chirawu  
Chinese Name: 亚历克上  
Student Number: M202161029

## **Digital Signal Processing , Information And Engineering Department, USTB**

Professor: 陈媛

*13 January 2023*

# Summary

The main goal of this project report is to automate major functions that are currently performed manually by car drivers. This report primarily focuses on the autonomous vehicle's detection and estimation development process. This report outlines the software requirements for self-driving autonomous vehicles, with high emphasis on Signal Processing(detection and estimation). Autonomous cars are regarded as safer and more efficient than manual drivers in this day and age.

The system is built with Python and the Tensorflow library, as well as OpenCV for image detection, Pykit for distance calculation, and radar for obstacle avoidance. The hardware components for the development are: a four-wheel-drive X5 frame that serves as the car's body, a Raspberry Pi 3 that serves as the car's brain, a dc to dc buck converter that serves as the car's vein for current supply, a motor driver that will carry instructions from the raspberry pi to the wheels, a 12V battery that serves as the car's heart, an LCD that displays the current network's IP address, an I2C chip and a camera and the radar.

When the car uses the camera for path recognition in Experiment 1, it is set on the path and estimates the path and wheel location. In Experiment 2, the automobile moves around using radar to avoid all obstructions.

Key findings include:

- In recent years, there has been a significant growth in the demand for human safety.
- With the help of a camera and radar, self-driving cars can avoid obstacles and go to their intended destination.
- The car is trained under situations similar to those it would encounter in the real world, and Machine Learning plays a key role throughout this time.
- In image processing benchmarks, the same strategy of compressing data also played a vital role.

The information presented in this project report has been gathered from my project and secondary sources.

The project report has been prepared for submission as the Information Signal Processing end of Course at University of Science and Technology in Beijing , Information and Communication Department.

## TABLE OF CONTENTS

<b>1. Introduction .....</b>	<b>1</b>
a) Methodology .....	1
b) Scope of the project report .....	1
c) Findings.....	2
<b>2. Detection and Estimation .....</b>	<b>3</b>
1. Experiment 1: Camera lane detection, estimation.....	3
a) Real scenarios.....	3
b) Pixel Summation .....	3
c) Lane .....	4
2. Experiment 2 : Radar detection and estimation .....	6
a) Estimation Linear models .....	7
b) Estimation Kalman filtering .....	7
<b>3. Experiments(Results) .....</b>	<b>9</b>
a) Experiment 1: Camera lane detection, estimation .....	9
b) Experiment 2 : Radar detection and estimation .....	10
<b>4. Comparisons .....</b>	<b>11</b>
<b>5. Conclusion.....</b>	<b>11</b>
<b>6. Recommendation .....</b>	<b>12</b>
<b>7. Reference List .....</b>	<b>13</b>

# 1.Introduction

I'm going to use the TensorFlow Library because it has a number of libraries that have a lot of mathematical equations in the background. The main equations for the radar and camera are Gaussian noise equations, Minimum Variance Unbiased Estimation (MVUE), and Best Linear Unbiased Estimators (BLUE) which is a combination of Gaussian and Markov theorems. We don't need to understand every detail of all the processing required for neural networks to build one using this amazing framework.

With only a brief discussion of hardware and associated software, the employment of a camera and radar to detect and estimate the distance between two points on a path.

## a) Methodology

Information for this report was sourced from my projects and various secondary sources, all listed in the Reference List. The Data from publications by the IEEE transactions on industry applications also proved valuable. This project report is a comprehensive review of the available research , which provides a narrow view of the topic.

## b) Scope of the project report

Wherever the term radar on its own is used, it references Radio Detection Ranging. Camera can detect the path but it's not efficient enough. Neither radar alone is not enough to calculate the distance between the car and an object and the right lane. This will be clearly shown and discussed in the findings below. There is also ongoing research on spatial domain methods like neural network - based methods and deep stacked auto system auto encoders to cover this problem. The main idea is to prevent cars from crashing into each other, objects and going off lane. To calculate and filter the noise signal processing gives great outputs on this aspect. At the end providing a good performance for the autonomous car as considered from the expected result

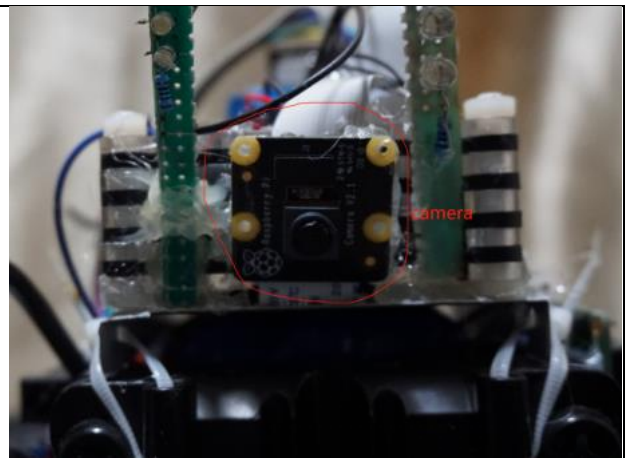
## c) Findings

Mainly signal processing is a branch of electrical engineering that studies, modifies, and synthesizes signals including sound, pictures, and scientific measurements. Signal processing techniques can be used to increase transmission, storage efficiency, and subjective quality of a measured signal, as well as to accentuate or detect components of interest.

In communication engineering signal processing has played a major role in advancing and improving many systems.

### Types of cars used for the experiments

The 4 wheel driver was high enough to support the camera for road detection and estimation in experiment 1, while the tank vehicle worked efficiently for radar detection in experiment 2 due to its top flatness, box shape, and 40 percent better turning angle than the 4 wheel driver. These are the two main pieces of equipment I used in the experiments.



## 2. Detection and Estimation

### 1. Experiment 1: Camera lane detection, estimation

#### a) Real scenarios

The path below is the key image that I used to conceptualize this system. The lane has a left, right, and straight path that spans over 80% of the world's highways. All of these angles must be calculated properly for the automobile to stay on track otherwise it will crash into other cars or go off track. From this image our first problem arises which is to estimate the camera angle and the environment. So Self-calibration falls within the more general problem of reconstruction [7]. That is, consider a set of 3-D points viewed by  $N_{cam}$  camera with calibration matrices  $P^i$ ,  $i = 1, \dots, N_{cam}$ . Let  $m_j^i = p^i w_j$  be the homogeneous coordinates of the projection of the  $j^{th}$  point onto the  $i^{th}$  camera. The reconstruction problem is then to find the set of camera matrices  $P^i$  and scene structure (world coordinates)  $w_j$  such that  $m_j^i = p^i w_j$ . Depending on the assumptions and restrictions placed on  $P^i$ , the scene may be recovered to varying degrees of reality. Then the coordinates of the matching point  $P_k$  in image  $k$  are calculated from  $P_0$  as  $P_k = H_k P_0$ . Given  $H_k$ , the desired parameters may be obtained algebraically. For this project I used a block chain algorithm to convert low resolution images (240 x 240 pixels) captured at a high frame rate (15 Hz).



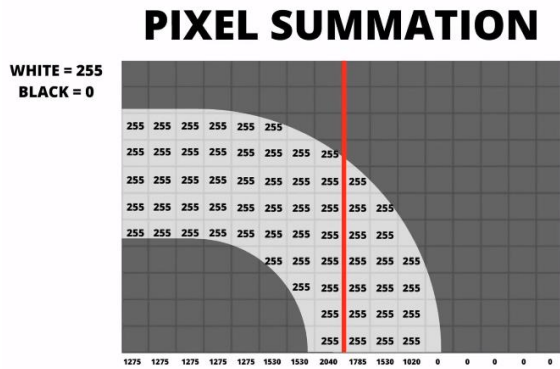
#### b) Pixel Summation

The camera is placed in the center of the front wheel else the data will be biased. The ideal filter for this assignment is the Gaussian filter. Row photographs taken by a moving object contain a lot of noise, which needs to be filtered first before all the implementations are applied to the image. We only need to use tensorflow to invoke

a Gaussian function that contains the following equation:

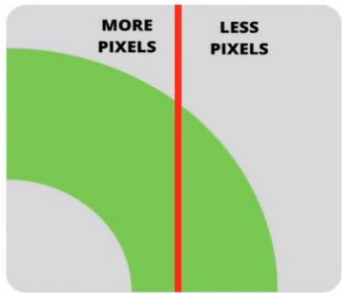
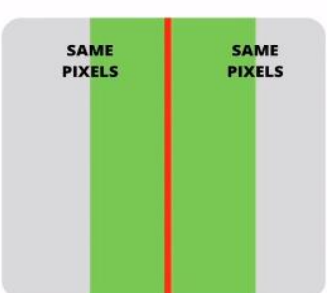
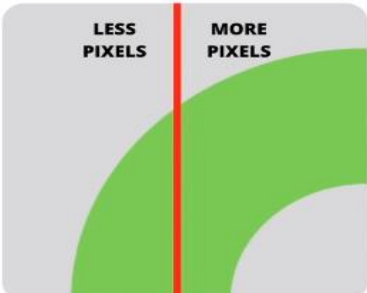
$$G(x, y) = \frac{1}{2\pi\delta^2} e^{-\frac{x^2+y^2}{2\delta^2}}$$

Pixel summation is used in the following way: white pixels are represented by 255, which we obtain from the color picker module, and utilis, where I did thresholding, turning everything to background and foreground color. The background is always black and has a value of 0. This makes it easier to apply Minimum Variance Unbiased Estimation (MVUE), and Best Linear Unbiased Estimators (BLUE).



**c) Lane**

As a result, I used The path is white, which is represented by 255, and any color is changed to black, which is represented by 0. We expect three outputs from this aspect: left curve, straight curve, and right curve. The advantage of utilizing this strategy is that it improves lane detection and angle estimate accuracy. The image taken by the camera has a lot of noise in it, so it's filtered before being compared to others in the dataset for detection. The data in the dataset is summarized in the table below. To obtain acceptable lines, a canny edge detector or any other edge detector can be used, however in my project, I utilize track detection to make training this approach easier. The pixel summation will let us figure out how many curves are in our path.

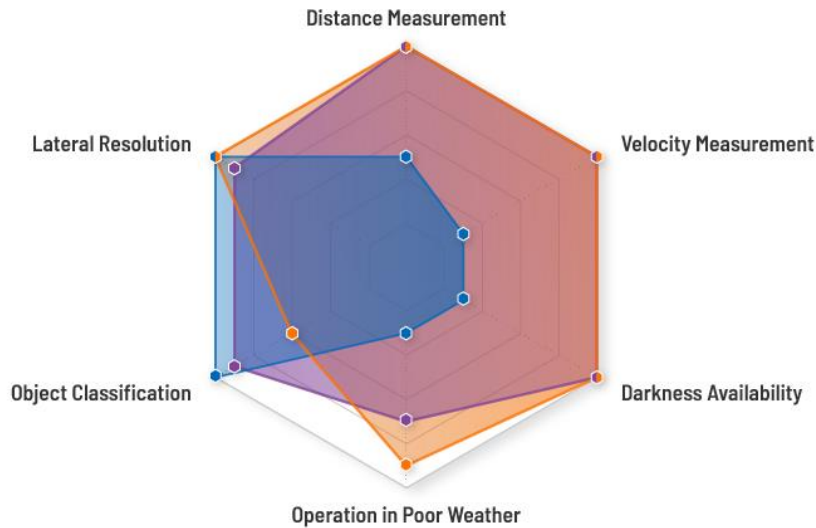
To summarize PIXEL SUMMATION , there are have three scenarios:		
 <p><b>LEFT CURVE</b></p>	 <p><b>STRAIGHT</b></p>	 <p><b>RIGHT CURVE</b></p>
<p>The automobile recognizes that the curve is heavy on the left side, detecting a turn to the left, but the angle of turn is evaluated by picture histogram and color filter so that the car does not turn too early or too late for this circumstance.</p>	<p>The equation for a straight line is simple: the values on the left side are equal to the values on the right side, resulting in an even line. The race car will accelerate straight in this</p>	<p>The automobile recognizes that the curve is heavy on the right side, detecting a turn to the right, but the angle of turn is evaluated by picture histogram and color filter so that the car does not turn too early or too late for this</p>

<p>The red line serves as a pixel separator, allowing the program to calculate the amount of pixels on each side and properly move the automobile. The camera's position is also indicated by the red line.</p> $y = mx + c$	<p>circumstance. The image should be centered so that the left and right sides are equal in size.</p>	<p>circumstance. The red line serves as a pixel separator, allowing the program to calculate the amount of pixels on each side and properly move the automobile. The camera's position is also indicated by the red line.</p>
--	---	---



## 2. Experiment 2 : Radar detection and estimation

Radar detection and distance estimation are very interesting in the sense that it's a blind car which can move without causing a collision with obstacles. This is the cycle of radar and the main and effective part is the object classification and Lateral resolution which makes the front of the car.



A radar contains an Antenna which is the connecting link between some guiding system and the free space which is the path. It does only have two functionalities which are sending radiate electromagnetic energy and receiving electromagnetic energy.

The radar rotates 100% cycle clockwise(CW). The first thing about the radar is the power density(PD) at a range R from the radar is :

$$PD_D = \frac{P_{CW} G_t L_1}{4\pi R^2} \text{ w/cm}^2$$

The term  $L_1 (< 1)$  is the one-way atmospheric trans atmospheric transmission factor.

$$L_1 = e^{-\alpha R_x}$$

$R_x$  is the path length in cm's and  $\alpha$  is the one way extinction coefficient

### a) Estimation Linear models

The antenna radiates electromagnetic energy which is represented in red and then it receives electromagnetic energy which is represented by green. The yellow rectangle shows the clear path. The Antenna receives the electromagnetic energy and then compares the values with those in the database and compares the distance . In the picture below it received the signals from short electromagnetic energy proving that the distance from the obstacle and the antenna is too close, so the car needs to move to the open space. Using Best Linear Unbiased Estimator which is under linear model with the formula:

$$x = H\theta = w$$

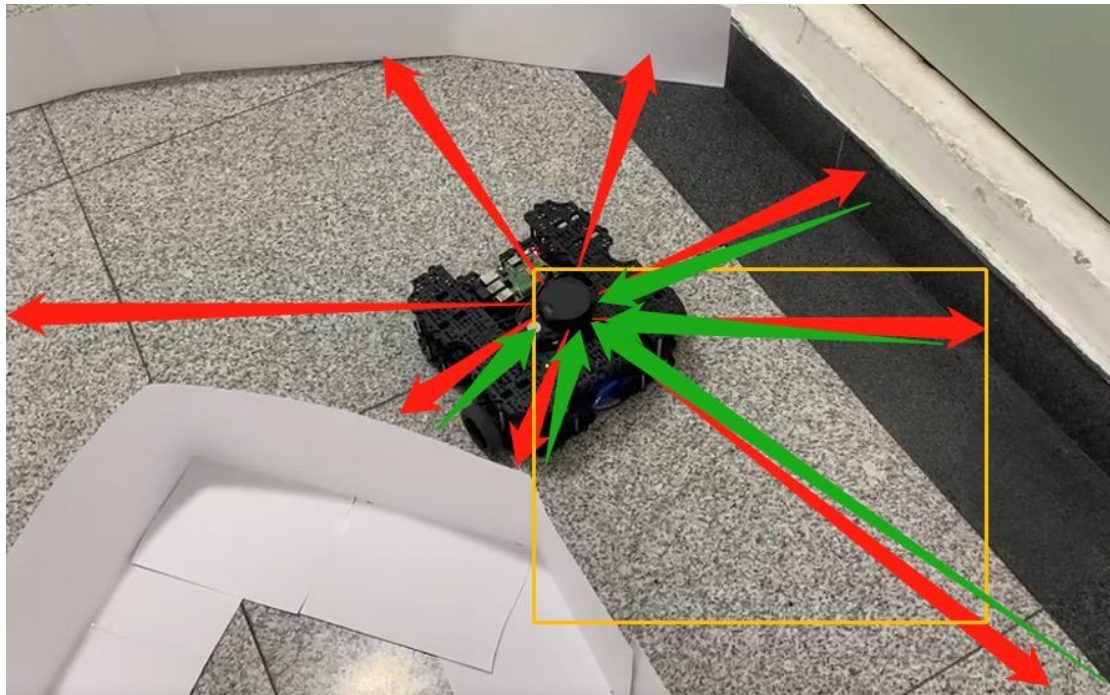
Where

$\theta$  = the vector parameter to be estimated

$x$  = received signal from which to estimate  $\theta$

$H$  = (known) observation matrix

$w$  = noise of statistical characterization  $N(0, \alpha^2 I)$



### b) Estimation Kalman filtering

Kalman filtering is known as linear quadratic estimation(LQE), this algorithm uses a measurement observed over time which includes statistical noise and other inaccuracies and produces estimates of unknown variables that are more correct than Gaussian filtering which is based on a single measurement alone. It estimates joint probability distribution over the variables for each time frame.

Using this kalman filtering the car can calculate the distance from the antenna to the surroundings very fast so that the car can keep its speed as well as avoiding obstacles. Using Tensorflow , we just call the kalman function which states that:

- recursive filter for estimating internal state of linear dynamical system from a series of noisy measurements

$$\begin{aligned}\mathbf{X}_k &= \mathbf{F}_k \mathbf{X}_k + \mathbf{W}_k \\ \mathbf{Y}_k &= \mathbf{H} \mathbf{X}_k + \mathbf{V}_k \\ \mathbf{W}_k &\sim \mathcal{N}(\mathbf{0}, \mathbf{Q}_k), \quad E[\mathbf{W}_k \mathbf{W}_j^T] = \mathbf{Q}_k \delta_{k-j} \\ \mathbf{V}_k &\sim \mathcal{N}(\mathbf{0}, \mathbf{R}_k), \quad E[\mathbf{V}_k \mathbf{V}_j^T] = \mathbf{R}_k \delta_{k-j},\end{aligned}$$

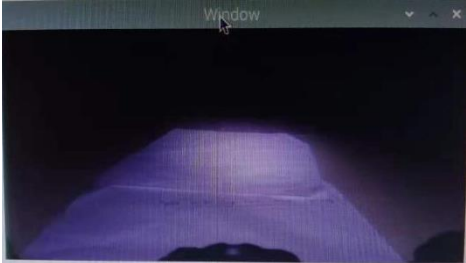


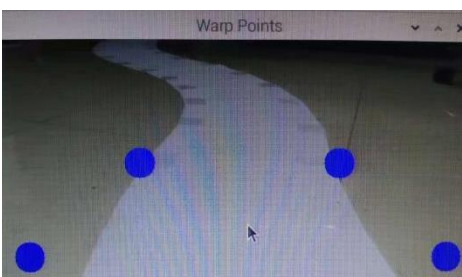
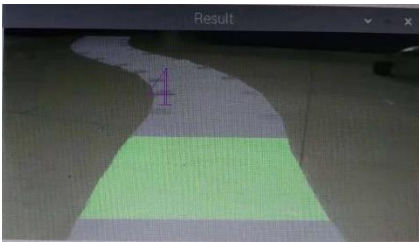
- than can recursively estimate/predict and update the state covariances as:

$$\begin{aligned}\mathbf{P}_k^- &= \mathbf{F}_{k-1} \mathbf{P}_{k-1}^+ \mathbf{F}_{k-1}^T + \mathbf{Q}_{k-1} \\ \mathbf{K}_k &= \mathbf{P}_k^- \mathbf{H}_k^T (\mathbf{H}_k \mathbf{P}_k^- \mathbf{H}_k^T + \mathbf{R}_k)^{-1} = \mathbf{P}_k^+ \mathbf{H}_k^T \mathbf{R}_k^{-1} \\ \mathbf{x}_k^- &= \mathbf{F}_{k-1} \mathbf{x}_{k-1}^+ \\ \mathbf{x}_k^+ &= \mathbf{x}_k^- + \mathbf{K}_k (\mathbf{y}_k - \mathbf{H}_k \mathbf{x}_k^-) \\ \mathbf{P}_k^+ &= (\mathbf{I} - \mathbf{K}_k \mathbf{H}_k) \mathbf{P}_k^-\end{aligned}$$

### 3. Experiments(Results)

#### a) Experiment 1: Camera lane detection, estimation

**The Output of this project is submissive.**

<p>The outputs of detection and estimation</p> 	<p>The path is captured as a white lane by the camera. The primary purpose of the window screen is to ensure that the car is in the correct position.</p>
<p>Threshold</p> 	<p>Because the surrounding region has so many colors, it's difficult to deal with them all at once, thus the best option is to convert the RGB colors to a Gray image and apply a Gaussian filter which just comprises black and white (2 colors) and is simple to utilize. The white color represents the Lane that the autonomous vehicle must follow.</p>
<p>Histogram</p> 	<p>The most efficient technique for the car to detect the midway point of the track is to use the Histogram. The algorithm, laneModule.py, and utilis.py module determine the width of the white lane on threshold and always keep track of the middle path of the path. The left and right distances of the lane will be calculated with the help of a matrix. The car detects the lane and uses the color code to determine which path is the primary one to take.</p>
<p>Wheel Points</p> 	<p>Using a warping technique, the wheel points are estimated as dotted blue on the screen. Because I made all of the modules reusable, the warp aids in wheel alignment adjustments based on the car's kind and size.</p>
<p>The final screen output</p> 	<p>I set the middle section of the curve to 0 and negative values as left curve and positive values as right curve for the algorithm to estimate if the curve is traveling left or right. The main aim is that the angle of the curve must remain at 0 , that means the car takes turns providing good results. The car will follow the side with the most white values, as determined by the algorithm.</p>

## **b) Experiment 2 : Radar detection and estimation**

- Data input: Copy radar scan, of any scan type short, medium or long.
- Temporal alignment: Compute the elapsed time between radar scans and use it to sample the ego motion module for the transformation matrix representing the change in position between radar scans.
- Egomotion compensation: Use the transformation matrix to rotate and translate the tracked radar object's state mean, error covariance estimate, and other properties.
- Spatial alignment: Move the tracking origin from the vehicle coordinate system to the sensor coordinate system to allow proper prediction and data association.
- Prediction: Estimate the new position and velocity of the tracked objects based on elapsed time between radar scans, as well as the estimation search region for track-to-detection data association.
- Association: Probabilistically match incoming radar detections with current radar tracks.
- Update: Use association scores to cluster detections to each track. From clusters update state estimates for position, velocity, acceleration, boundary, statistics and motion type (moving, stationary, stopped). Confirm tracks above certain quality metrics. Remove tracks below certain quality metrics.

## 4. Comparisons

Despite the fact that Kalman filters can be seen as a special case of Gaussian processes they differ in the way the models need to be thought about physical state-based versus covariance function that describe the underlying process.

Radar proved to be the best under the Kalman filter since the estimate variables of interest can't be measured directly. Kalman filter proved so effective since the value provided by the radar were continuously changing. The great advantage of using Kalman filter are numbered including that Kalman is light on memory in the sense that it doesn't keep any history other than the previous state which proved to be effective for the automobile. Due to the fact that it doesn't keep history the speed is fast and that makes it suited for real time scenario as proven by the experiment.

Gaussian filtering, on the other hand, is more effective in image processing where noise and details must be removed. Gaussian reduces salt and pepper noise, which is commonly found in photos, and it is quite good at smoothing images. Due to the fact that lot of process needs to be done , Gaussian turns to be slow as compared to Kalman filtering in radar.

## 5. Conclusion

We've seen in this research that camera detection and distance estimation, as well as radar detection, function well. Image processing has the disadvantage of being able to focus on small areas, making it difficult to detect sharp turns and obstacles in the path. This problem is well solved by radar, which can detect obstacles all the way around but not potholes, which can be detected using the camera image processing method. All of the projects yielded positive results, and they were able to move from point A to point B under specific environmental conditions.

In image processing, Gaussian theory is superior for removing noise signals, but Kalman theory is better for noise filtering on radar because it utilizes a joint probability distribution that is fast when the radar is rotating 100 percent clockwise. Gaussian filtering performed well on images recorded by the camera and then compared to the dataset to determine the turning angle. Gaussian employs single value calculation, which takes a lengthy time if applied in radar.

## 6. Recommendation

The information collected for this project report provides a broad overview of key changes in the image processing and radar detection and distance estimation methods. Further analysis would be possible if camera and radar are combined together and the same test are taken. The reliance on secondary data has resulted in some patchy data. For example, it is not possible to identify for any data loss during signal processing by the following categories:

Too many obstacles around the cars

Plane site

Therefore increasing the number of experiments in different conditions would enable a more thorough analysis to be made.

## 7. Reference List

L. Zheng, M. Lops, Y. C. Eldar, and X. Wang, "Radar and communication coexistence: An overview: A review of recent methods," *IEEE Signal Process. Mag.*, vol. 36, no. 5, pp. 85–99, 2019.

P. Dwivedi, "CarND-Advanced Lane Finder-P4," GitHub, 9 March 2017. [Online]. Available: <https://github.com/priya-dwivedi/CarND/tree/master/CarNDAdvanced%20Lane%20Finder-P4>. [Accessed 11 June 2020].

A. Mimouna, A. B. Khalifa, I. Alouani, N. E. B. Amara, A. Rivenq and A. Taleb-Ahmed, "Entropy-Based Ultra-Wide Band Radar Signals Segmentation for Multi Obstacle Detection," in *IEEE Sensors Journal*, vol. 21, no. 6, pp. 8142-8149, 15 March 2021, doi: 10.1109/JSEN.2021.3050054.

P. M. McCormick, S. D. Blunt, and J. G. Metcalf, "Simultaneous radar and communications emissions from a common aperture, part I: Theory," in *Proc. IEEE RadarConf*, May 2017, pp. 1685–1690.

Ma, Dingyou, et al. "Joint Radar-Communications Strategies for Autonomous Vehicles." *Arxiv.Org*, 26 May 2020, <https://arxiv.org/pdf/1909.01729.pdf>.

Daniel Casado Herraiez, Javier Diaz Dorronsoro, PhD, Andoni Medina, MSc " Self-Driving Autonomous System Overview", San Sebastian - Donostia, June 2020.

H. Kong, H. C. Akakin and S. E. Sarma, "A Generalized Laplacian of Gaussian Filter for Blob Detection and Its Applications," *IEEE Transactions on Cybernetics*, 2013.

Bjom, Hugosson. "Introduction of Autonomous Vehicles in the Swedish Traffic System : Effects and Changes Due to the New Self-Driving Car Technology." *DIVA*, Klara Hager, 4 June 2015, <http://www.diva-portal.org/smash/record.jsf?pid=diva2%3A816899&dswid=118>. Klara Hager

"Self-Driving Car: A Step to the Future of Mobility." *BracU IR*, <http://dspace.bracu.ac.bd/xmlui/handle/10361/11027>. Accessed 13 Apr. 2021.  
Subramanyam, Rakshith. "Chatropolis - A Self Driving Car Test Bed | ASU Digital Repository." *ASU Digital Repository*, 1 Apr. 2018, <https://repository.asu.edu/items/49419>.

Wiseman, Yair. "Autonomous Vehicles." *Encyclopedia of Information Science and Technology, Fifth Edition*, IGI Global, 2021, pp. 1–11, <http://dx.doi.org/10.4018/978-1-7998-3479-3.ch001>.



Youssef, Fenjiro, and Benbrahim Houada. “International Journal of Intelligent Systems and Applications(IJISA).” *MECS Press-Hong Kong*, Fenjiro Youssef, 8 Oct. 2020, <http://www.mecs-press.org/ijisa/ijisa-v12-n5/v12n5-2.html>.

Frost Sullivan, “Global autonomous driving market outlook, 2018 (frost sullivan reports, march 2018),”  
<https://info.microsoft.com/rs/157-GQE-382/images/K24A-2018%20Frost%20%26%20Sullivan%20-%20Global%20Autonomous%20Driving%20Outlook.pdf>, [Online; accessed Jun. 22, 2019].

Y. Sun, P. Babu, and D. P. Palomar, “Majorization minimization algorithms in signal processing, communications, and machine learning,” *IEEE Transactions on Signal Processing*, vol. 65, no. 3, pp. 794–816, 2017.

Z. Hu, Z. Zheng, T. Wang, L. Song, and X. Li, “Roadside unit caching: Auction-based storage allocation for multiple content providers,” *IEEE Transactions on Wireless Communications*, vol. 16, no. 10, pp. 6321–6334, 2017.

E. Ndashimye, N. I. Sarkar, and S. K. Ray, “A novel network selection mechanism for vehicle-to-infrastructure communication,” in *Proceedings of IEEE 14th Intl. Conf. on Pervasive Intelligence and Computing, 2nd Intl. Conf. on Big Data Intelligence and Computing and Cyber Science and Technology Congress (DASC/PiCom/DataCom/CyberSciTech)*, 8-12 Aug. 2016 (Auckland, New Zealand), pp. 483–488.

A. Ndikumana, N. H. Tran, and C. S. Hong, “Deep learning based caching for self-driving cars in multi-access edge computing,” *arXiv preprint arXiv:1810.01548*, 3 Oct. 2018.