MARMARA UNIVERSITY
INSTITUTE FOR GRADUATE STUDIES IN PURE AND APPLIED SCIENCES

AN AREA-EFFICIENT FPGA IMPLEMENTATION OF LANE AND OBSTACLE DETECTION FOR DRIVING ASSISTANCE SYSTEMS

FATİH TAT

MASTER THESIS
Department of Electrical and Electronics Engineering

Thesis Supervisor
Asst. Prof. Dr. Salih BAYAR

ISTANBUL, 2022

MARMARA UNIVERSITY
INSTITUTE FOR GRADUATE STUDIES IN PURE AND APPLIED SCIENCES

AN AREA-EFFICIENT FPGA IMPLEMENTATION OF LANE AND OBSTACLE DETECTION FOR DRIVING ASSISTANCE SYSTEMS

FATİH TAT

MASTER THESIS
Department of Electrical and Electronics Engineering

Thesis Supervisor
Asst. Prof. Dr. Salih BAYAR

ISTANBUL, 2022

ACKNOWLEDGMENT

I would first like to thank my thesis advisor Asst. Prof. Dr. Salih BAYAR of the Faculty of Engineering at Marmara University with my sincere wishes. He guided me with his knowledge from the first day of the master's thesis process. Whenever I ran into a problem, the door to his office was always open to me. His determination, discipline, diligence, motivation, and energy provided a significant contribution to the development of my thesis.

Finally, I sincerely thank my family for their support and encouragement throughout my education life and in this thesis process. I am proud and honored that I have achieved success with their support.

CONTENT

ÖZET

SÜRÜŞ DESTEK SİSTEMLERİ İÇİN ŞERİT VE ENGEL TESPİTİNİN ALAN VERİMLİ FPGA UYGULAMASI

Bu tez, sürücüler için daha güvenli ve konforlu bir sürüş sağlayacak düşük maliyetli Alanda Programlanabilir Kapı (FPGA) cihazları üzerinde bir sürücü yardım sisteminin Şerit ve Engel algılama alt sistemlerini geliştirmeyi amaçlamaktadır. Tasarlanan algoritmada başlıca hedefler daha hızlı, verimli, farklı hava koşullarında ve farklı yol koşullarında çalışabilen bir sistem oluşturmaktır. Son zamanlarda Sürücü asistan sistemi, Uyarlanabilir hız sabitleyici, Kilitlenme önleyici fren sistemi, Otomotiv navigasyon sistemi, Çarpışma önleme sistemi, Acil durum sürücü asistanı vb. gibi benzer sistemlerle ilgili birçok çalışma yapılmıştır. Bu tez çalışmasında kenar algılama algoritmalarını, ilgi alanı yöntemini (ROI) ve Hough dönüşümünü (HT) kullanarak şerit takip ve nesne belirlemeyi gerçekleştirmektedir. Önerilen sürücü destek sistemleri Xilinx Zynq-7000 FPGA ve MATLAB yazılım platformu üzerinde 7 farklı hava ve yol koşulu üzerinde yaklaşık 1400 görsel üzerinde test edilmiştir. FPGA üzerinde elde edilen sonuçların MATLAB ile elde edilen sonuçlara göre 1322 kata kadar daha hızlı olduğu gözlemlenmiştir. 1400 görsel için test edilen Sobel operatörünün 1332 görsel ve Prewitt operatörünün 1288 görsel üzerinde şerit belirlemede doğru çalıştığını göstermiştir. Sobel filtresinin sonuç doğruluğu olarak Prewitt filteresine göre şerit belirlemede %6.2 ve engel algılamada %3.3 daha iyi olduğu kanıtlanmıştır. Look up Table (LUT) kullanımı açısından tasarlanan algoritma 1906 LUT kullanarak %38.95 daha verimlidir. Bu sistem, gerçek zamanlı uygulamalar için kabul edilebilir olup kenar algılama algoritmaları, ilgi alanı yönetimi ve Hough Dönüşümü yöntemlerini uygulayarak saniyede 27 kare işleyebilir.

ABSTRACT

AN AREA-EFFICIENT FPGA IMPLEMENTATION OF LANE AND OBSTACLE DETECTION FOR DRIVING ASSISTANCE SYSTEMS

This thesis aims to develop the Lane and Obstacle detection subsystems of a driver assistance system on low-cost Field-Programmable Gate Array (FPGA) devices that will provide a safer and more comfortable ride for drivers. The performance parameters are system speed, computational efficiency, and working in different weather and road conditions. Recently, many studies have been carried out on systems such Driver assistant system, Adaptive cruise control, Anti-lock braking system, Automotive navigation system, Collision avoidance system, Emergency driver assistant etc. The presented thesis performs Lane and Obstacle detection by using edge detection algorithms, region of interest (ROI), and the Hough Transform (HT) method. Proposed driver support systems are implemented on Xilinx Zynq-7000 FPGA and MATLAB software platform on approximately 1400 images over seven different weather and road conditions. It has been observed that the results obtained on FPGA are up to 1322 times faster than the results obtained with MATLAB. The study showed that for 1400 frames, the Sobel operator works correctly at 1332 frames and the Prewitt operator at 1288 frames at lane detection. The Sobel filter has been proven to be 6.2% better than the Prewitt filter at lane detection and 3.3% better at Obstacle detection in result efficiency. The designed algorithm implemented in terms of Lookup Tables (LUTs) usage is 38.95% more efficient than using 1906 LUTs, with a low energy footprint. This system can process 27 frames per second by applying edge detection algorithms, ROI, and HT methods which are acceptable for real-time applications.

SYMBOLS

Th **:** Threshold Value

Gy : The gradient of y

Gx : The gradient of x

G : Gradient vector

d : Distance between the line and the origin

r : Angle of the vector from the origin to this nearest point

ER : Efficiency rate of the system

SF : Successfully detected frames

TF : Total number of frames.

ABBREVIATIONS

FPGA **:** Field-Programmable Gate Array

FF : Flip Flops

LUTs : Look-up Table

BRAMs : Block Rams

WHO : World Health Organization

ASIC **:** Application Specific Integrated Circuit

VLSI : Very Large Scale Integration

PC **:** Personal Computer

ROI **:** Region of Interest

RGB **:** The Red-Green-Blue

LoG **:** Laplacian of Gaussian

MSE **:** Mean Square Error

ADAS **:** Advanced Driver Assistance System

HT **:** Hough Transform

RANSAC **:** Random sample consensus

LIST OF FIGURES

LIST OF TABLES

1. INTRODUCTION

With the advancement of technology, technological devices that make our lives easier have started to take place in our lives, and one of them is the driver assistant systems, which have recently become popular. Driver assistance systems are the technological system developed for easy, safe, and self-driving [1]. The benefit of systems is that it provides a safe journey crucial for safety because a large part of the increased traffic accidents is caused by driver negligence. According to the data of the World Health Organization (WHO) published in 2015, 1.25 million traffic accidents occurred 2015. It has been reported that a total of 270,000 people lost their lives in these traffic accidents, and an average of more than 700 lives were lost daily [2]. The increase in accidents and human losses has caused scientists to question the cause of traffic accidents. Thus, the causes of traffic accidents have started to be investigated as a new research area. The main reason for this is that the number of accidents caused by driver mistakes such as sudden lane changes is quite high. In addition, a study has shown that 90% of traffic accidents are caused by driver [3].

The most common driver mistakes happen when vehicles traveling on their roads abruptly and unintentionally change lanes (e.g. falling asleep, losing control of the steering wheel, etc.). Sudden lane changes of vehicles may damage the driver as well as the vehicles in the opposite lane. For example, the United States transportation department reported that approximately 25,000 people lost their lives on American roads, as 42,643 people changed their lanes [4]. The statistical data clearly show the significance of the Lane detection system. Another factor that makes the driver assistance system essential is that the systems allow the detection of obstacles in front of the vehicle. This system helps drivers to prevent possible accidents by means of obstacles (e.g. vehicles, traffic safety equipment, barriers, bumps, and any unexpected objects) in case of sleep or distraction [5].

Driver assistance systems have been developed in response to the advancement of technology and the need for safe driving. Many products have been developed using the Application Specific Integrated Circuit (ASIC), Very Large Scale Integration (VLSI) [6]. These systems were first developed on a personal computer (PC) using image processing techniques. Although there are many studies in the literature, the problems encountered are the slow operation of the systems and their inoperability in different road conditions. Depending on the application [7-10], FPGAs are faster than traditional microprocessor-based systems (ASIC, VLSI). Thus, they are more suitable for the implementation of

algorithms. As microprocessor-based systems, FPGAs can be embedded in many different environments including vehicles. They also provide the advantages of performing intensive calculations in a short time compared to microprocessor-based systems. FPGA can perform many operations in parallel at the same time and provides these operations at high speeds [11-13].

The main objective of this work is to design a functional system under different weather conditions (e.g. sunny, rainy, cloudy, etc.) and in certain driving environments (e.g. heavy traffic, rough ground, faint stripes, etc.) with acceptable success rates. It is planned to implement Lane and Obstacle detection modules which are essential components of driver assistance systems. Within the scope of this thesis containing both modules are implemented on the Xilinx ZYNQ-7000 FPGA platform and MATLAB. Recently, Xilinx FPGA has been preferred in universities and industries [13]. At the same time, thanks to this FPGA hybrid system, embedded system application, and implementation can be realized. The proposed thesis is aimed to convert the image to grayscale and then detect the lanes and obstacles by using ROI and edge detection methods (e.g. ROI, Canny algorithm). And, lanes are colored by applying Hough transform. Sobel and Prewitt's operators are compared in terms of performance. The designed algorithm is implemented on FPGA and MATLAB, and the calculation time is compared. In addition, the use of LUTs is obtained on the FPGA as an important issue. It is planned to implement an algorithm on FPGA that completes quickly and requires low LUTs usage as a performance parameter. We aim to reduce LUT usage as much as possible.

The rest of this thesis is organized as follows. Chapter 2 examines the related study. Chapter 3 describes the methodology for Lane and Obstacle detection algorithms on the image. Chapter 4 provides information on FPGA simulation and implementation of lane and Obstacle detection algorithms and Chapter 5 contains the conclusion.

2. RELATED WORK

Many studies have been carried out in the field of driver assistance systems in recent years. Studies have focused especially on Lane and Obstacle detection systems from driver assistant systems. These systems differ according to applied algorithms and implemented programs. The common aspect of the studies is to design systems that work fast and efficiently. In this direction, the Lane and Obstacle detection process on FPGA has started to gain importance and many studies have been carried out in this area [14-23].

In the literature review, it has been proven that many Lane detection algorithms are used on FPGA. In the study [14], Canny, Prewitt, Sobel, and Roberts operators are tested on FPGA and it is shown that the Canny algorithm is the most time-consuming (Roberts three times faster) operator. However, it has been shown that the Roberts algorithm determines the lanes outside the road, which shows that the success rate of the system is low. Although the results show that it requires a smaller area cost in the study [15], which is another Roberts operator application, it is seen that the number of training sets used is not sufficient. Another study [16] stated that it consumes more time and had failures due to noise while performing lane detection using the Hough transform (HT). In [17], a success rate between 87% and 98% is achieved in a day and night trials by applying a sequential algorithm such as the peak-finding and grouping, edge connecting, lane segment combination, and lane boundaries selection. However, it is seen that the lanes are prominent and do not require any extra work. It has been shown that it takes 7.8 ms to find the results of large-size images (1024x1024 pixels) to Lane detection by applying the Prewitt operator in the study [18]. This amount of time is unusually long for the Lane detection algorithm. The common problem in the experiments where the HT operator is applied is seen as high time-consuming as in [19-21]. Using the Sobel operator, Kalman filter, and HT are combined and good results are obtained in different road conditions with LUTs usage being 45% [22].

Experiments are carried out on various techniques in [23]. It can be seen from the results that FPGAs accomplish deployments at different times and with different LUTs. Thus, the importance of FPGA selection is emphasized in terms of LUTs usage and simulation execution times. The deficiencies seen in the literature are eliminated by the implemented algorithm and FPGA selection.

3. METHODOLOGY

In this study, Lane detection and Obstacle detection processes are carried out using seven different video data sets. The process is started by taking frames from each video. Frames are converted to grayscale to process and the received image dimensions are set to 512 x 512. The edge detection process is carried out by applying the edge detection filters (Sobel and Prewitt) separately by throwing the ready frames into Vivado. At the same time, edge detection is performed on MATLAB using the same algorithms to compare Vivado and MATLAB. Regions of interest on the images for which edge detection is performed are determined separately for Lane detection and Obstacle detection. The final boundary of the lanes is determined by applying HT in the selected regions. The Lane detection process is completed by drawing lanes over the input image. The object determination process is completed by making controls in the ROI selected for the Obstacle detection process. Figure 3.1 shows the proposed architecture of Lane and Obstacle detection.

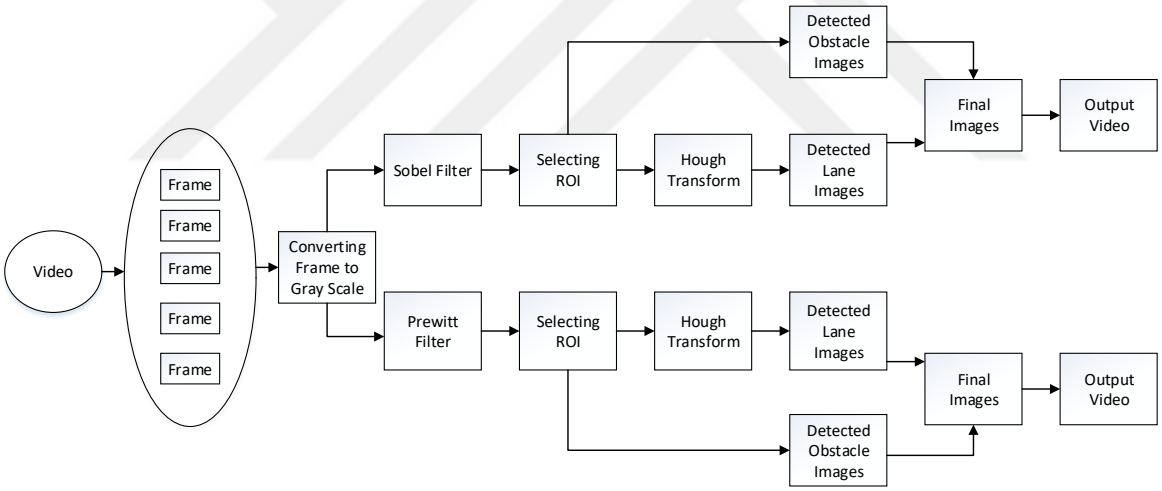


Figure 3.1 Proposed Architecture of Lane and Obstacle detection

The test architecture is depicted in figure 3.2.

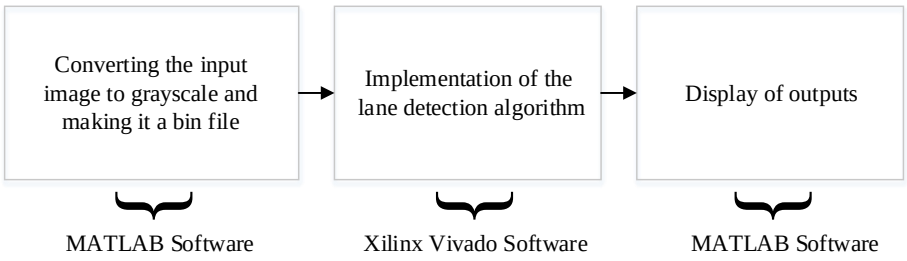
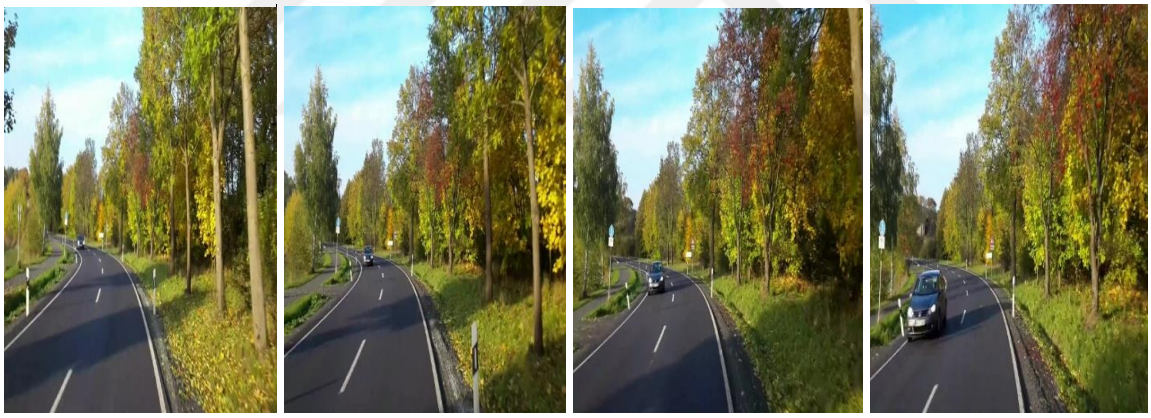


Figure 3.2 Test architecture

This section moves on to describe in greater detail the methodology of Obstacle and Lane detection methods used in this study.

3.1 Taking frames from video

The video is divided into frames as seen in Figure 3.3.



(a) (b) (c) (d)

Figure 3.3 Frames taking from video (a) Frame 1 (b) Frame 2 (c) Frame 3 (d) Frame 4

3.2 Converting Frame to Gray Scale Image

In the frames we take for image processing, each pixel takes 256 different values between 0-255 of 8 bits and this is called The Red-Green-Blue (RGB) color space. Images containing 0 (black) and 255 (white) values are called grayscale images. Grayscale image processing is preferred in image processing because RGB values contain more

information (lightness, chroma, and hue) for each pixel and create complex mathematics [24]. Grayscale images have a single intense value due to the equal intensity of red, green, and blue in RGB values, thus providing ease of mathematical processing. Therefore, the frames we obtained are converted to gray scale as seen Figure 3.4.

(a) (b) (c) (d)

Figure 3.4 (a)-(d) Frames Converted to gray scale

3.3 Edge Detection

There is a lot of work going on in image processing and edge detection is a method that works on highlighting edge lines. It has an important place in technological fields such as image processing, pattern recognition, and computer vision [25]. Edge detection just tries to find areas in the image with sharp changes in intensity or color, with high values representing sharp changes and low values representing shallow changes. The process of edge detection significantly reduces the amount of data and filters out unneeded information, while preserving the important structural properties of an image.

Edge detection algorithms have advantages such as speed, efficiency, and simplicity, as well as disadvantages such as low positioning accuracy and sensitivity to noise. The performance of edge detection techniques varies according to applications, even if they use the same operators. Edge detection methods can be divided into two groups Gradient and Laplacian. These reasons gave rise to many edge detection methods and they are first-order differential operators (gradient) Roberts operator, Sobel operator, and Prewitt operator, and Second order differential operators Laplacian operator and Laplacian of Gaussian (LoG) operator. The Gradient method detects the edges by looking for the maximum and minimum in the first derivative of the image. The Laplacian method searches for zero crossings in the second derivative of the image. In recent years, many algorithms such as neural networks and morphology-based edge detection algorithms

have been developed and continue to be developed [26-27]. Figure 3.5 shows image edge detection operators.

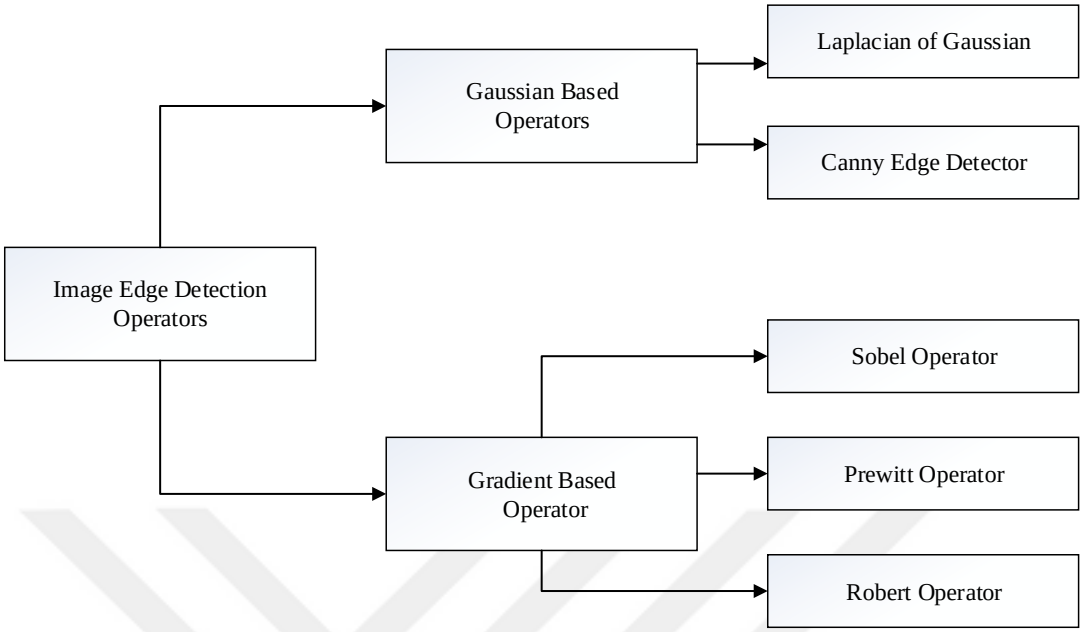


Figure 3.5 Image edge detection operators

Sobel and Prewitt operators are chosen as edge detection operators in this study. The reason for choosing these two operators is that they are easy-to-implement algorithms and give good results. Easy to implement means that results can be obtained by making fewer mathematical calculations. In this way, the circuit created in the hardware application will not need to be very large. Thus, the system can be implemented without the need for high computational costs and long computation times.

3.3.1 Sobel Operator

The Sobel edge detection method is introduced by Sobel in 1970 (Rafael C. Gonzalez (2004)) [28]. The Sobel filter is widely used in image processing and computer vision, especially in edge detection algorithms [29]. It is an approach to the derivative of an image as a working principle. The pixel value changes dramatically at the image's edge, and the gradient amplitude is considerable. We may utilize the gradient of an image to identify whether it contains the edge we seek and where it is located.

Sobel operator has advantages such as fast detection speed, noise smoothing, and suppression. As a working principle, it determines the edges in a certain direction and angle by convolving the image with the kernel masks. Generally used kernel masks consist of 3x3 matrices as seen in Figure 3.6. Many studies have been done with kernel masks of different sizes and different elements, but in this study, the kernel masks seen

in Figure 3.6 are preferred. These filters are usually performed in two directions, x and y directions, which are horizontal and vertical in edge detection applications.

$$\text{Kernel 1=} \begin{bmatrix} 1 & 0 & -1 \\ 2 & 0 & -2 \\ 1 & 0 & -1 \end{bmatrix} \qquad \text{Kernel 2=} \begin{bmatrix} 1 & 2 & -1 \\ 0 & 0 & 0 \\ 1 & 2 & 1 \end{bmatrix}$$

(a) (b)

Figure 3.6 (a) Horizontal kernel mask for Sobel operator (x-direction) (b) Vertical kernel mask for Sobel operator (y-direction)

Convolution multiplication to be made with kernel masks is done separately in the x-direction and y-direction. Let's call the matrix of the image taken from the video and converted to grayscale I. The convolution in the x-direction is called the gradient of x (Gx), the convolution in the y-direction is called the gradient of y (Gy) and this formulation is expressed as in Figure 3.7.

$$\text{Gx} = \begin{bmatrix} 1 & 0 & -1 \\ 2 & 0 & -2 \\ 1 & 0 & -1 \end{bmatrix} * \text{I} \qquad \text{Gy} = \begin{bmatrix} 1 & 2 & -1 \\ 0 & 0 & 0 \\ 1 & 2 & 1 \end{bmatrix} * \text{I}$$

(a) (b)

Figure 3.7 (a) Gradient of x-direction (b) Gradient of y-direction

If the input image has T rows and Y columns and the kernel has three rows and three columns, the output image will have size M -3 + 1 row and N-3 + 1 column. In this study, T = 512 and Y=512 because the images are converted to 512x512 size after converting to a grayscale image. So the size of Gx and Gy will be 510x510. The convolution is performed by sliding the kernel over the image starting at the top left corner, to move the kernel through all the positions where the kernel fits entirely within the boundaries of the image. Mathematically, we can formulate Gx and Gy as (1) and (2).

$$Gx(i,j) = \sum_{k=1}^{m} \sum_{z=1}^{n} \text{I}(i+k-1, j+z-1) Kernel1(k,z) \qquad (1)$$

$$Gy(i,j) = \sum_{k=1}^{m} \sum_{z=1}^{n} I(i+k-1,\ \ j+z-1)\ Kernel2(k,z) \quad (2)$$

Gradient vector (G) calculated as in Eq. (3).

$$G = \sqrt{(Gx^2 + Gy^2)} \quad (3)$$

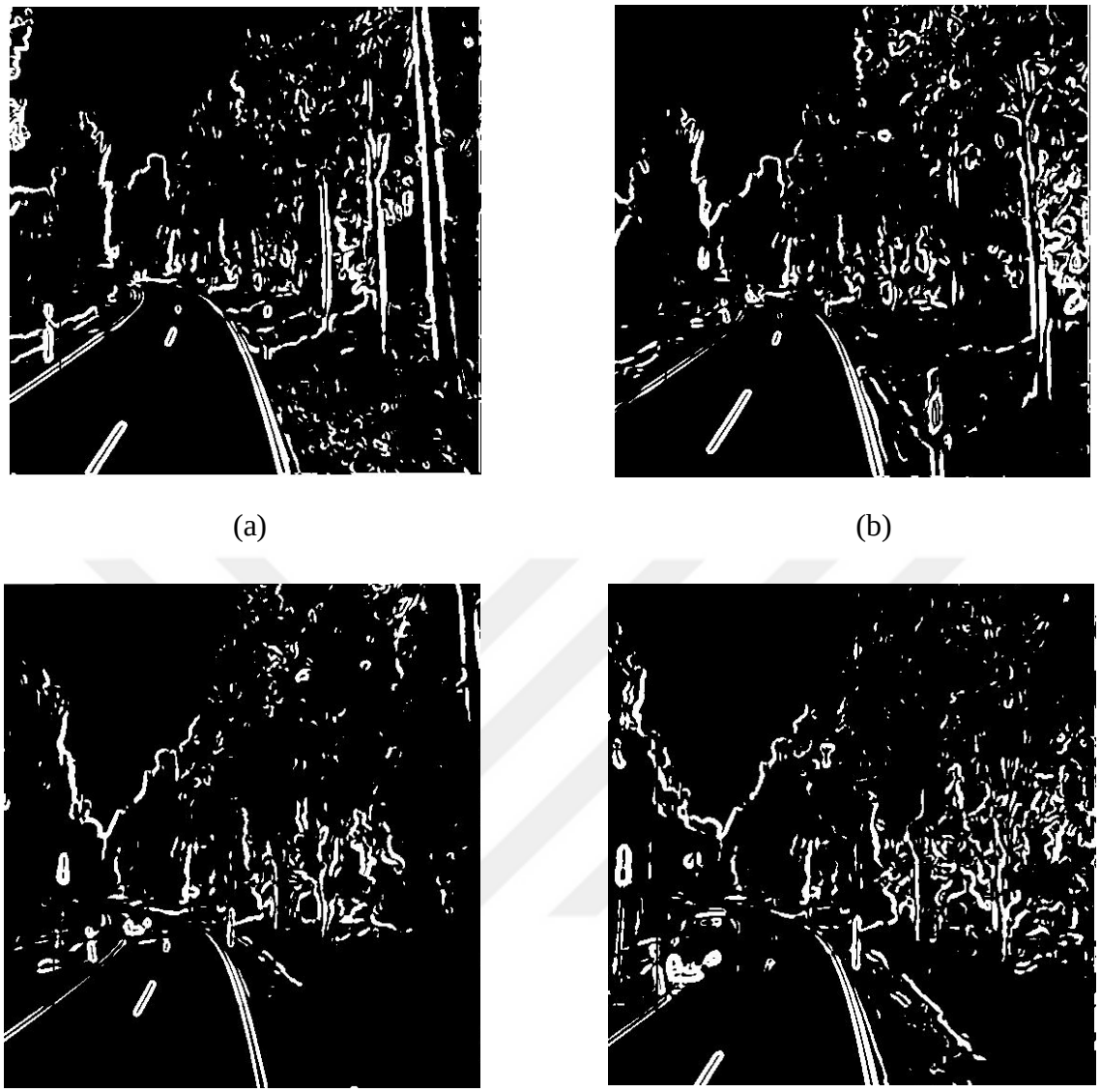
Eq. (3) introduces computational costs and long computation times that complicate the circuit. Therefore, the Eq. (4) is generally used to calculate Gradient [30].

$$G = |Gx| + |Gy| \quad (4)$$

Finally, the calculated gradient image (G) is compared with the determined threshold value (Th) as in Eq. (5).

$$G_{final}(x,y) = \begin{cases} G(x,y), \mid G(x,y) \geq Th \\ 0 \quad , \mid G(x,y) < Th \end{cases} , 0 < G(x,y) < 255 \quad (5)$$

To compare the Th, a value between 0 and 255 is determined and checks are made as can be seen in Eq. (5). If the gray value in the gradient matrix is greater than or equal to the specified Th value, it is taken as the edge point. If it is small, it is not an edge point and is set to 0. The defined edge points are connected to be the image edges detected by the Sobel operator. Sobel operator applied frames can be seen in Figure 3.8. It is seen that the lanes are well defined for consecutive frames.



(a) (b)

(c) (d)

Figure 3.8 Frames applied Sobel operator (a) Frame 1 with Sobel operator applied (b) Frame 2 with Sobel operator applied (c) Frame 3 with Sobel operator applied (d) Frame 4 with Sobel operator applied

3.3.2 Prewitt Operator

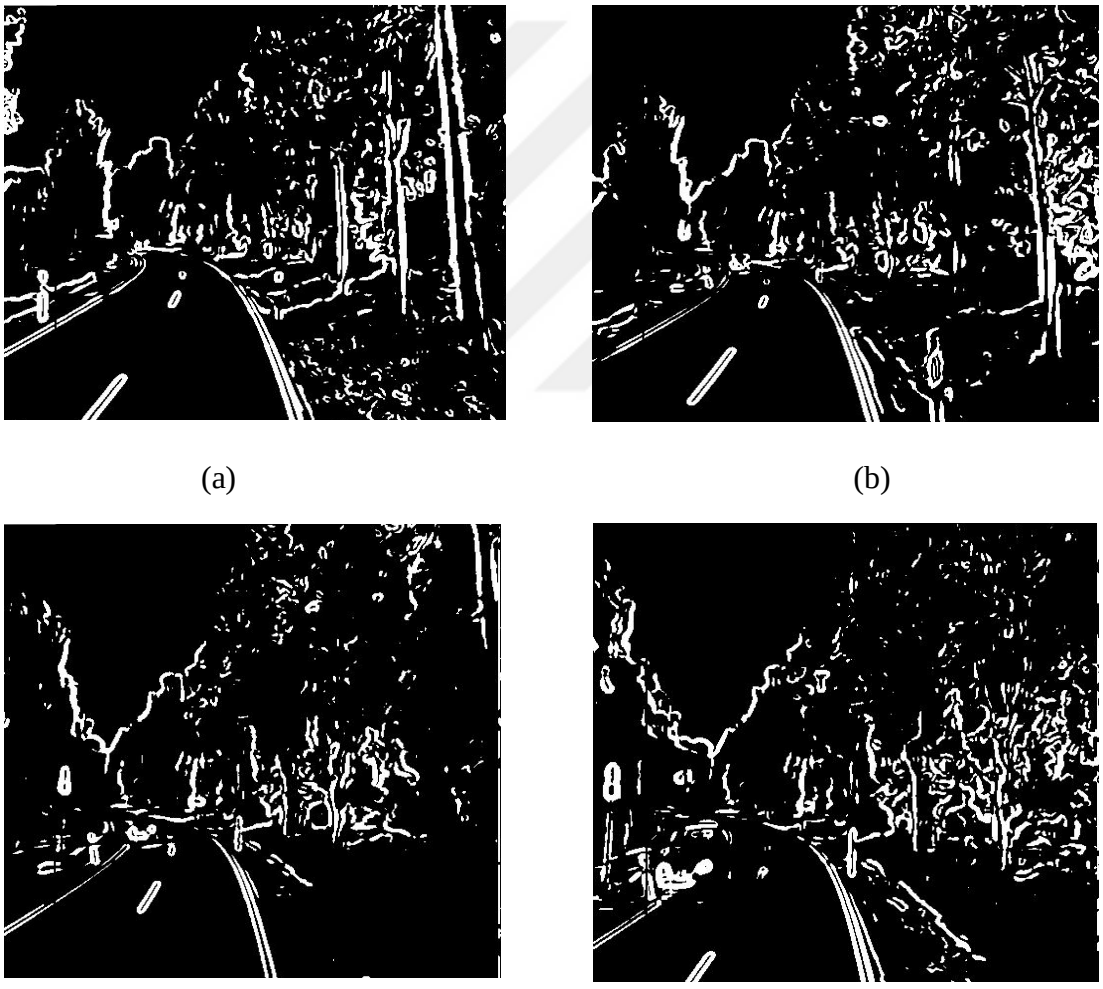
The Prewitt edge detection algorithm is developed by Prewitt in 1970 [31]. Prewitt detection is somewhat simpler to implement computationally than Sobel detection, but tends to produce noisier results because of changed kernel mask in Figure 3.9.

Kernel 1 = $\begin{bmatrix} 1 & 0 & -1 \\ 1 & 0 & -1 \\ 1 & 0 & -1 \end{bmatrix}$ Kernel 2 = $\begin{bmatrix} -1 & -1 & -1 \\ 0 & 0 & 0 \\ 1 & 1 & 1 \end{bmatrix}$

(a) (b)

Figure 3.9 (a) Horizontal kernel mask for Prewitt operator (x-direction) (b) Vertical kernel mask for Prewitt operator (y-direction)

Also this operator does not place any emphasis on pixels that are closer to the center of the masks. Prewitt operator applied frames can be seen in Figure 3.10.



(a) (b)

(c) (d)

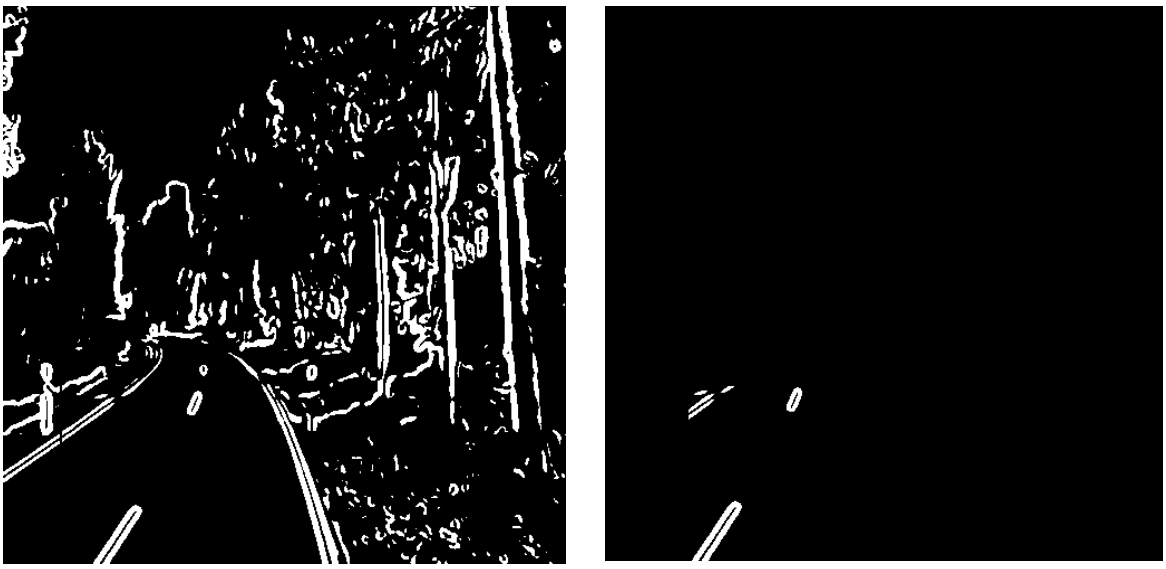
Figure 3.10 Frames applied Prewitt operator(a) Frame 1 with Prewitt operator applied (b) Frame 2 with Prewitt operator applied (c) Frame 3 with Prewitt operator applied (d) Frame 4 with Prewitt operator applied

3.4 Region of Interest

The separation of the region to be examined on an image is called ROI in image processing. ROI has not been preferred much in image processing, but it has started to be preferred in recent years and has been widely used especially in the field of medical image processing. In this area, there are studies such as mean square error (MSE) and threshold to determine the ROI [32-33]. ROI produce quick and accurate result by avoiding unnecessary image examination [34-35].

However, in this thesis, ROI is used to select the same regions for each image taken from a video. The selected regions are used for two purposes: Lane and Obstacle detection. Three different ROIs are determined from the image taken, namely the left lane, the right lane, and the region indicating the front of the vehicle. For the determined ROI regions to work efficiently and to perform the algorithm, the camera placed in the vehicle must be located in the middle of the vehicle. The purpose of placing the camera in the middle region of the vehicle is to create two ROI regions covering the lower left and lower right corners of each image taken. These two ROI regions formed will cover the lanes and will make it easier to perform Lane detection without examining redundant areas. The third ROI will show the front of the vehicle and facilitate Obstacle detection.

Although this type of work was not common in advanced driver assistance systems (ADAS), there seem to be numerous studies in medical image processing. Using this technique in this thesis has been a good advancement in terms of ADAS. Figure 3.11 shows three different ROI regions taken over a frame.



(a) (b)

(c) (d)

Figure 3.11 (a) Edge detection operator applied image (b) ROI applied to detect left lane (c) ROI applied to detect right lane (d) ROI applied to detect obstacle

3.5 Hough Transform

The Hough transform is a method used in image analysis, computer vision, and digital image processing to describe the specification of an image [36]. The HT started to be used in 1962 with the patent issued by Paul Hough [37]. In 1972, it is developed and continued to be used by Richard Duda and Peter Hart in a study called "generalized HT " [38]. It started to become popular in 1981 with the article "Generalizing the HT to detect arbitrary shapes" by Dana H. Ballard.

HT is used in many fields, but especially in image processing to extract lines, circles, and ellipses. In this study, it is used to detect straight lines, which is the simplest use of HT. It can express straight lines as Eq. (6) in the image space and is drawn graphically for each (x, y) point.

$$y=mx+b \tag{6}$$

In the HT, the value of m and b are more important than point coordinates (xi,yi) to indicate straight lines. However, two new polar coordinates, d, and r have been defined because it causes unlimited m and b values in vertical lines. As can be seen in Figure 3.12, the d value is the distance between the line and the origin, and the r-value is the angle of the vector from the origin to this nearest point.

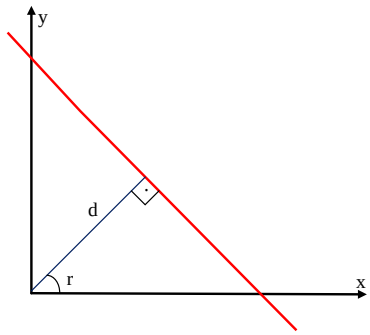


Figure 3.12 Lane coordinates with parameters

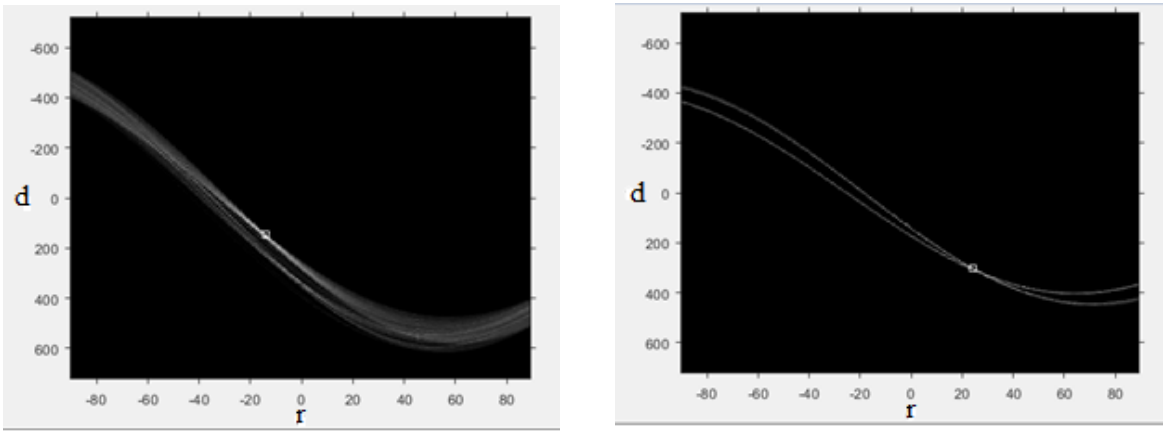
By adding 2 new parameters, Eq. (6) can be written as Eq. (7).

$$y = (-\frac{\cos r}{\sin r})x + (\frac{d}{\sin r}) \quad (7)$$

Mathematically, we can formulate d as (8).

$$d = y\sin(r) + x\cos(r) \quad (8)$$

Lanes can be drawn by finding the two peaks and slopes of the lines obtained using Eq. 8, which is one of the HT equations. In Figure 3.13, the d and r values of the peak points found with the HT appear for the left lane and the right lane.



(a) (b)

Figure 3.13 (a) Hough transform peaks of a sample left ROI (b) Hough transform peaks of a sample right ROI

3.6 Showing Detected Lanes and Obstacle

The peaks between the two lanes determined by the HT and the slope found allowed the determination of the lane. By using these parameters, the Lane detection process has been completed as seen in Figure 3.14.

(a) (b)

Figure 3.14 (a)-(b) Detected lane Images

Obstacle detection is completed by checking the intensity of the ROI region, which is taken as the Obstacle detection region. Obstacle detection ROI is adjusted to cover only the front of the vehicle. Thus, this selected region covers lanes and obstacles by applying edge detection algorithm. Detected lanes and objects are highlighted in white and take the value one. Lanes are separated using Hough transform and obstacles are only remaining in the selected region. By controlling the value of the remaining region mathematically, it is seen that the objects are determined when the intensity is above 0.3. The obstacle is determined by performing the control mechanism.

3.7 Hardware Implementation Sobel and Prewitt Operator

Many studies have been done on the implementation of the Sobel operator on FPGA. Studies vary according to the use of FPGA resources, the size of the images used, and the computation time of the algorithm. In addition, the studies differ according to the algorithms developed, the design elements used, and the design styles. The most common positive effect in the studies on FPGA is parallel processing. In the study [39], the author

applied the Sobel filter in parallel and stated that it achieved approximately 153 times faster than sequentially running designs. But, 43 images, which is a low number for the training set, were tested and it is seen that the use of resources is high. Another study [40] used pipelining and increased operating speed but the memory usage was high. RAM was used as memory instead of registers, and it was seen that the use of system resources was reduced, but it was seen that the system was running slower [41].

In this thesis, the hardware realization consists of three parts as Sobel buffer memory, gradient computation module, and edge map module, as seen in Figure 3.15.

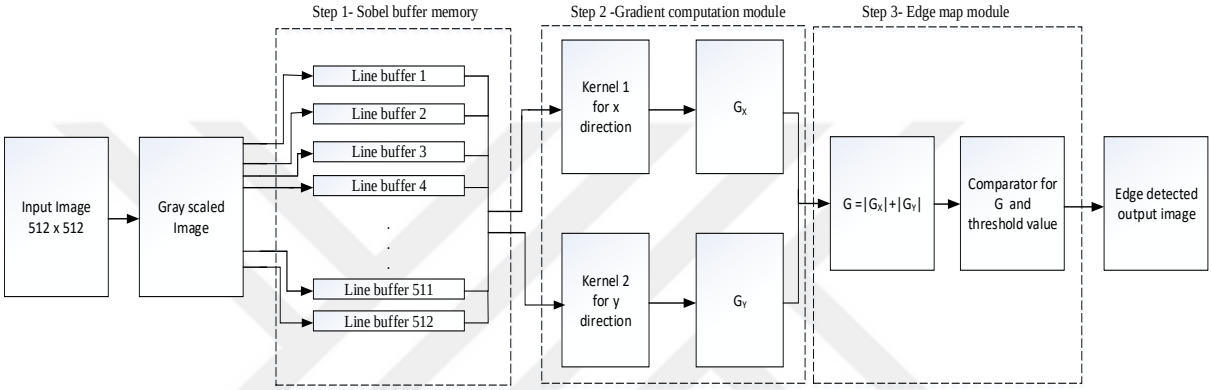


Figure 3.15 The proposed edge- detection hardware architecture using Sobel and Prewitt operator

Since the input image is grayscale, each pixel takes eight bits and must be stored in memory as it cannot be processed in one calculation. The Sobel and Prewitt operators apply a 3x3 kernel mask and nine pixels participate in the calculation in each operation. Step 1 in Figure 3.15 shows that the row of each image is taken as a buffer for efficient utilization and improvement. The reason for using a buffer is that the same input line will be used many times. The same input lines use many times doing convolution with buffer lines. Each convolution operation is performed on two dimensional three buffers for applying kernel mask.

Implementation of the Sobel and Prewitt algorithms is done with step 2 gradient computation as seen in Figure 3.15. To determine the horizontal and vertical lines, two gradient computations were implemented in parallel, as in Figure 3.16 and Figure 3.17.

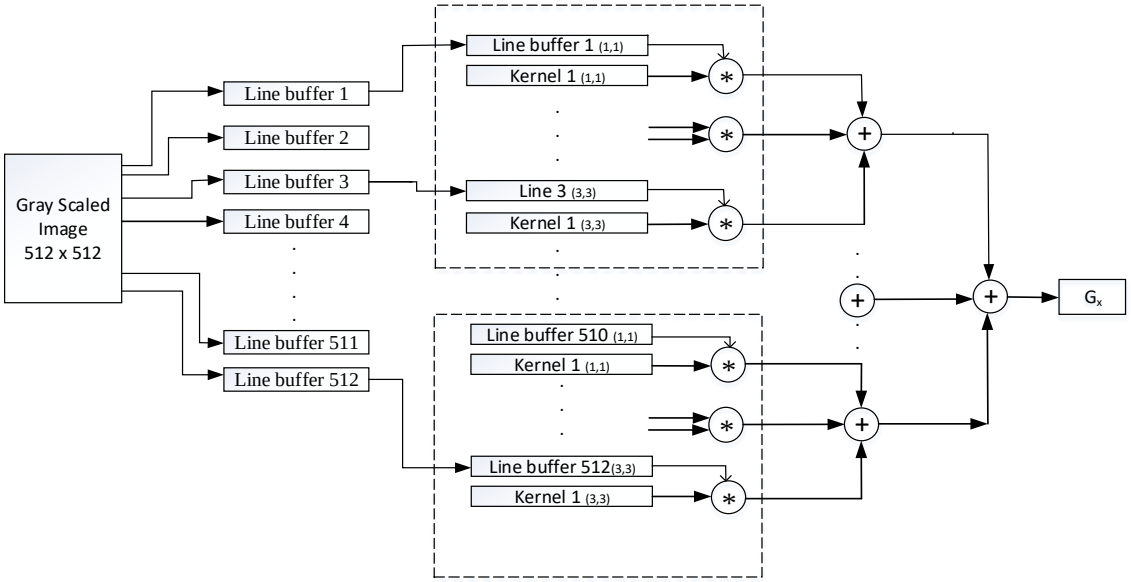


Figure 3.16 Convolution process by applying the Kernel 1 (x direction-horizontal lines) mask to the input gray scaled image

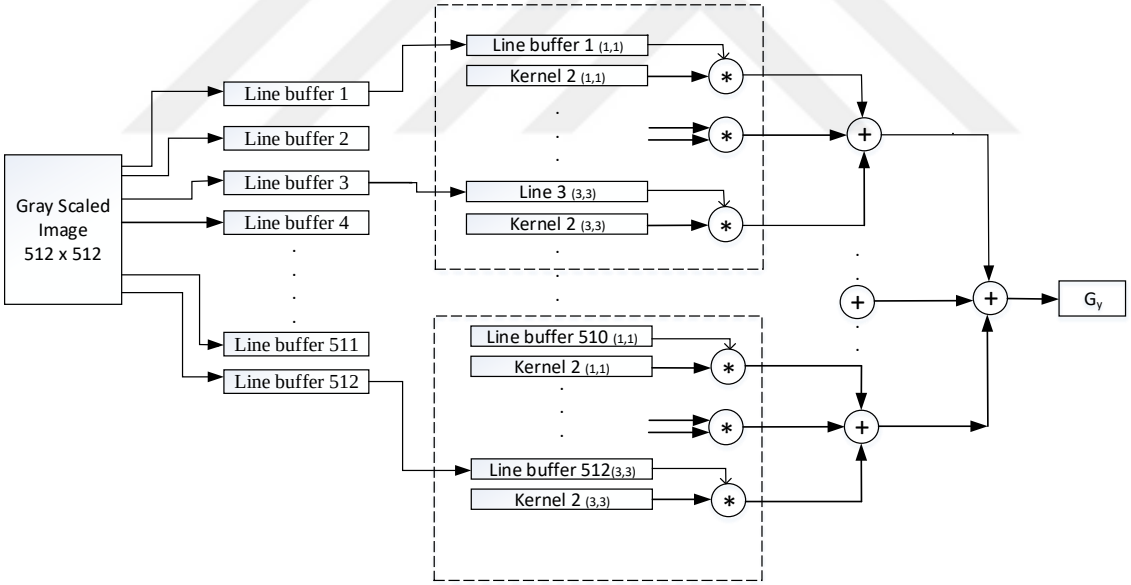


Figure 3.17 Convolution process by applying the Kernel 2 (y direction-vertical lines) mask to the input gray scaled image

The absolute value of the horizontal and vertical gradient results of the edge filter gives the gradient value as shown in step 3 in Figure 3.15. In Figure 3.18, the gradient value obtained is compared with the determined threshold value. If the specified threshold

value is lower than or equal to the gradient value, it is considered as edge and takes the value 255, and if it is greater, it takes the value 0.

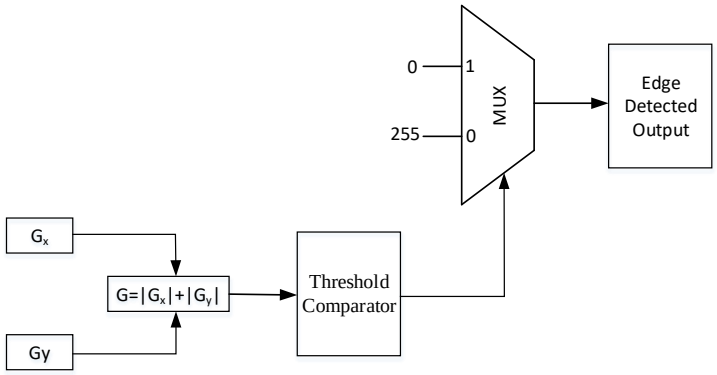


Figure 3.18 Finding edge points by comparing G with Threshold value

4. RESULTS AND DISCUSSION

In this study, videos with of seven different types are used and in the selection of videos, the camera required for the operation of this system is paid attention to be in the middle of the front of the vehicle. While selecting the data sets, the working efficiency of the designed algorithm is checked by selecting different types of videos (uncertain lanes, rainy weather, dark road, curvy roads, etc.). 200 frames are taken as input from each of these dataset videos, and the Lane and Obstacle detection algorithm is studied on a total of 1400 images.

Various experiments are carried out on the designed algorithm. And this algorithm has been tested on MATLAB as well as the hardware version Vivado 2017.4. These tests were carried out on a laptop equipped with an Intel Core i5-3230M CPU running at 2.60GHz 6GB DDR3 RAM. The Hardware implementation is carried out using the Xilinx Zynq-7000 FPGA.

4.1 Lane and Obstacle detection for seven different training sets

One of the tests of the studies is to show how the designed algorithm works in different road conditions. In this direction, seven different data sets have been selected as can be seen in Figure 4.1, including dashed road lines, arrival-departure and curvy road, travel in rainy weather, road with straight lanes, road with different color lanes, travel in foggy weather, road with more edge lines.

(a)

(b)

(c)

(d)

(e)

(f)

(g)

Figure 4.1 (a) dashed road lanes (b) arrival-departure and curvy road (c) rainy weather and night ride (d) road with straight lanes (e) different color lanes (f) travel in foggy weather (g) road with more edge lines

When the dashed, straight, different color, foggy weather, and curve lanes roads are examined, it is seen that the Sobel algorithm works very efficiently (over 95%) and there is almost no margin of error. In a different study on straight lanes using a Sobel filter and HT, the efficiency is reported to be 92.3% [42]. In a study using the Random sample consensus (RANSAC) method, an HT -based study is carried out and 90% efficiency is achieved by working on different color lanes [43].

It is observed that the efficiency decreased in rainy weather and night trials, but the Lane detection process is realized with 94% efficiency. The reason for this is that the raindrops reflected on the screen take the form of a line with the acceleration of the vehicle, and the algorithm perceives them as lanes. In another system, the Canny algorithm is used as an edge detection operator, and HT, lane boundary, and Hyperbola Fitting are used for Lane detection on curved roads in different weather conditions [44]. And the efficiency of the system has been observed to be in the range of 80%-90% on winding roads. In the implemented system, 96% efficiency is achieved on arrival-departure and curvy roads.

On the other hand, it has been observed that the efficiency of the road with more side lanes has decreased more than desired. Because of the effect of snowfall, the lane and roadsides are white, causing it to detect multiple lanes in some cases. As a result, as seen in Table 4.1, the efficiency of the algorithm is found to be high by trying several different training sets. The total efficiency of this driver assistant system by taking 1400 frames over seven different data sets has been calculated as 96.79% for the Sobel operator and 90.79% for the Prewitt operator. Where ER is the efficiency rate of the system, SF is the number of successfully detected frames and TF is the total number of frames.

$$ER = \frac{SF}{TF} x100 \quad (9)$$

Table 4.1 Comparison Lane Detection of Sobel and Prewitt Operator

Applied Operator	Total Frame received	# of correct results	# of wrong results	Efficiency Rate (%)
Sobel Operator	1400	1355	45	96.79
Prewitt Operator	1400	1271	129	90.79

The Obstacle detection system is made by comparing the intensity of the region determined by the ROI. The thesis study is planned to cover the front of the vehicle and achieved 95.14% for Sobel operator and 92% for Prewitt operator efficiency, as can be seen in Table 4.2. Many Obstacle detection algorithms have been developed for driver systems and efficiency between 90-95% has been achieved [45-47]. The designed thesis tried to make edge detection simpler using ROI and was successful in identifying obstacles in front of the vehicle.

Table 4.2 Comparison Obstacle detection results

Applied Operator	Total Frame received	# of correct results	# of wrong results	Efficiency Rate (%)
Sobel Operator	1400	1332	68	95.14
Prewitt Operator	1400	1288	112	92

Figure 4.2 shows the Lane and Obstacle detection outputs.

(a)

(b)

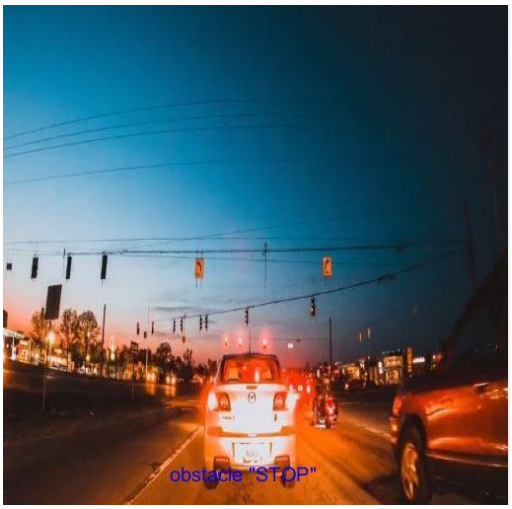
(c)

(d)

(e)

(f)

(g)



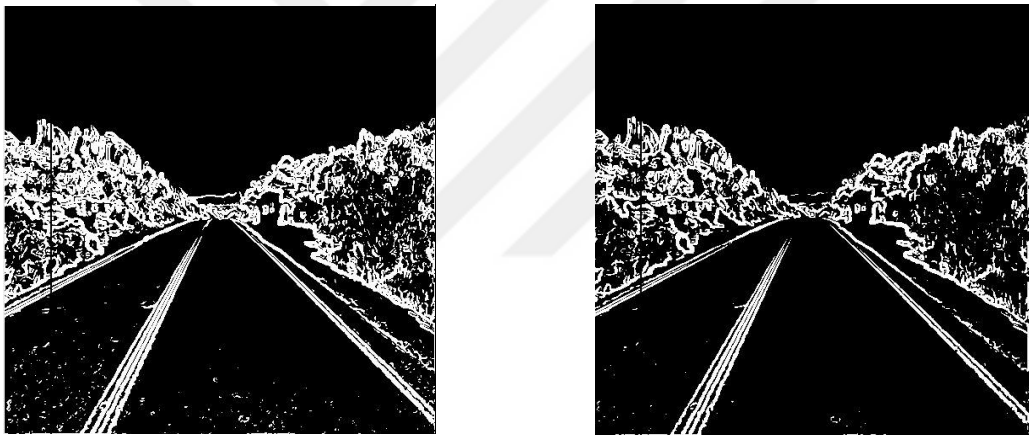
(h)

Figure 4.2 (a)-(g) Lane detection outputs (h) Obstacle detected image

4.2 Comparison of Prewitt and Sobel filter

One of the additional works of this thesis is the comparison of Sobel and Prewitt operators on the same input images. Both operators are derivative masks and are used for edge detection. These two operators are used to detect vertical and horizontal edges, but the Sobel operator is also efficient at detecting diagonal edges. In addition, while the Sobel operator emphasizes the pixels closer to the center of the mask, Prewitt is weak in this regard. Another feature that distinguishes these two filters is that the coefficient of the applied masks can be changed in the Sobel filter, but the mask is kept constant in the Prewitt algorithm. Also, the Prewitt mask is easier to apply and produces noisy results [48]. As can be seen in Table 4.2, the Sobel filter produces better results in this study.

In Figure 4.3, output images of two different operators applied on the same input image are shown.



(a) (b)

Figure 4.3 (a) Prewitt operator applied image (b) Sobel operator applied image

As shown Figure 4.3, the lanes in the Sobel operator are clear and continuous. However, while the Prewitt operator produces good results, there are small discontinuities in the lanes that make lane detection more difficult.

4.3 Hardware Results

Another goal of this thesis is to evaluate the performance of Sobel algorithms on MATLAB and FPGA. The Sobel operator has been tested in MATLAB 2021 and Vivado 2017.4. The Vivado implemented Sobel and Prewitt operators are executed on the Xilinx Zynq-7000 FPGA. When the same algorithm is constructed in Vivado and MATLAB, the FPGA is seen to be 1322 times faster than MATLAB. Table 4.3 illustrates the average computational durations for 1400 frames.

Table 4.3 Average simulation times of the operator on MATLAB and Vivado for 1400 frames

Applied Operator	Simulation time (second)	
	Sobel	Prewitt
MATLAB 2021	3.439537	3.548262
Vivado 2017.4 Xilinx Zynq-7000	0,0026	0.0026

Another significant issue is managing to keep the LUTs usage on the FPGA as low as reasonably achievable. Accordingly, the use of LUTs in the study is recorded as 3.58%(LUTs= 53200, used 1906), which can be seen in Table 4.4. In a study that performed edge detection using the Sobel and Prewitt operators, it is seen that the LUTs usage is 9% (LUTs= 33216, used 2989) [49]. In another study, three different FPGAs are used and these are SPARTAN-3 DEVICE XC3S400, SPARTAN-6 DEVICE XC6SLX25, VIRTEX-5 DEVICE XC5VLX50 [23]. The use of LUTs shown as SPARTAN-3 44%(LUTs= 7168, used 3166), SPARTAN-6 11%(LUTs= 15032, used 1680), VIRTEX-5 6%(LUTs= 28800, used 1927), respectively, and it is stated that the best performance is obtained from SPARTAN-6 in the use of low LUT number. In the study (50), many algorithms from edge detection operators are implemented on Zynq-based FPGA. The LUT usage of the Prewitt operator is specified as 3714 and the LUT usage of the Sobel operator is 3717. Improved, combined, vertical, and horizontal Sobel filters were tested in the study (51) using Zynq 7000 series FPGA. It has been observed that 3122 LUT is used in the combined Sobel filter application, which is the closest study to this thesis study. If we compare the implemented study with this study [51] in terms of LUTs usage, it is seen that it is approximately 38.95% more efficient

Table 4.4 Hardware utilization to design algorithm

Slice Logic Utilization	Used Utilization		Available	Utilization (%)	
	Sobel	Prewitt		Sobel	Prewitt
Number used as Flip Flops (FF)	253	253	106400	0.24	0.24
Look-up Table (LUTs)	1906	1906	53200	3.58	3.58
Block Rams(BRAMs)	0.5	0.5	140	0.36	0.36

5. CONCLUSION

The primary goal of this thesis is to create an FPGA- based Lane and Obstacle detection system using a novel algorithm approach that requires fast and low LUT usage. The designed algorithm is implemented on MATLAB and FPGA by collecting 1400 frames across seven different training sets. When selecting these training sets, special care is taken to select videos from the literature review that are difficult to detect lanes.

Sobel and Prewitt are compared in terms of performance and it is clearly shown that the Sobel filter is better at Lane and Obstacle detection. In this way, 96.79% efficiency in Lane detection and 95.14% efficiency in Obstacle detection is obtained using the Sobel operator. With the implementation of the Prewitt operator, 90.79% success is achieved in Lane detection and 92% success in Obstacle detection. Sobel is 6.2% better at Lane detection and 3.3% better at Obstacle detection. It has also been discovered that the Prewitt algorithm involves a noisy result, making the detection of ambiguous lanes challenging. When comparing the hardware implementations of these two algorithms, it is clear that they have roughly the same speed and use of LUTs.

Edge detection algorithms (Sobel and Prewitt) are also implemented on MATLAB and FPGA with the same testing image. In this comparison, the FPGA is shown to be 1322 times faster when carrying out the task on the same laptop. The fact that the FPGA can perform parallel calculations provides this working speed advantage. The use of LUTs usage, which is another reason for comparison, is found to be 38.95% more efficient than other systems compared in the literature.

It has been discovered that by performing the designed algorithm on the low-cost ZYNQ-7000 FPGA, it is a fast-running, low LUT usage, and high-efficiency system. It has also been evidenced that the Sobel operator is superior to the Prewitt operator in the edge detection algorithm.

Future work could be done by improving the edge detection filters used and experimenting with different filters. In addition, the designed algorithm can be tested in real-time.

REFERENCES

[1] P. Y. Hsiao, C. W. Yeh, S. S. Huang, and L. C. Fu, "A portable vision based real-time lane departure warning system: Day and night," *IEEE Trans. Veh. Technol*., vol. 58, no. 4, pp. 2089–2094, 2009.

[2] H. Jeppsson, M. Östling, N. Lubbe, "Real life safety benefits of increasing brake deceleration in car-to-pedestrian accidents: Simulation of Vacuum Emergency Braking" *Accid. Anal. Prev*., vol. 111, pp. 311–320, 2018.

[3] American Association of State Highway and Transportation Officials; Highway Safety Manual, AASHTO, USA, 2010.

[4] U.S. Department of Transportation, https://www7.transportation.gov/testimony/reauthorization-motor-vehicle-safety-programs-national-highway-traffic-safety, 2005.

[5] M. S. Darms, P. E. Rybski, C. Baker, C. Urmson, "Obstacle Detection and Tracking for the Urban Challenge", *IEEE Transactıons On Intellıgent Transportatıon Systems*, Vol. 10, No. 3, Sep 2009.

[6] G. Stein, E. Rushinek, G. Hayun, A. Shashua, "A Computer Vision System on a Chip: a case study from the automotive domain," *IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 2005.

[7] Y.J. Li, W. Zhang, N.J. Wu, "A novel architecture of vision chip for fast traffic lane detection and FPGA implementation*", IEEE 8th International Conference*, pp. 917–920, 2009.

[8] A. Aurobindo, S.S. Kakatkar, N. Mehendale, "A Review on Applications of FPGAs," *Society for Applied Microwave Electronics Engineering & Research*, 2022.

[9] S. Du, Z. Dong, Y. Li, T. Ikenaga, "Straight-Line Detection within 1 Millisecond per Frame for Ultra-High-Speed Industrial Automation," *IEEE transactions on industrial informatics*, pp. 1-11, 2022.

[10] A. Gabrielli, F. Alfonsi, F. D. Corso, "Simulated HT Model Optimized for Straight-Line Recognition Using Frontier FPGA Devices," *Electronics*, pp.1-9, 2022.

[11] H. Xiao, S. Xiao, G. Ma, C. Li, "Image Sobel edge extraction algorithm accelerated by OpenCL," *The Journal of Supercomputing*, 2022.

[12] A. Xiangjing, E. Shang, J. Song, J. Li, H. He, "Real-time lane departure warning system based on a single FPGA," *EURASIP Journal on Image and Video Processing*, 2013.

[13] P. Promrit, W. Suntiamorntut, "Design and Development of Lane Detection Based on FPGA," *International Joint Conference on Computer Science and Software Engineering*, 2017.

[14] W. Phueakjeen, N. Jindapetch, L. KuburaT, N. Suvanvorn, "A Study of the Edge Detection for Road Lane," *The 8th Electrical Engineering/Electronics, Computer, Telecommunications and Information Technology (ECTI)*, pp. 995-998, 2011.

[15] W. Kayankit, W. Suntiamomtat, "Hardware/Software Co-design for Line Detection Algorithm on FPGA," *6th International Conference on Electrical Engineer/Electronics, Computer, Telecommunications and Information Technology*, Vol. 01, pp. 604-606, 2009.

[16] Kaszubiak, M. Tornow, R.W. Kuhn, B. Michaelis, C. Knoepple, "Real-Time Vehicle and Lane Detection with Embedded Hardware," *IEEE Intelligent Vehicles Symposium*, pp. 619-624, 2005.

[17] M.J. Jeng, C.Y. Guo, B.C. Shiau, L.B. Chang, "Lane Detection System Based on Software and Hardware Codesign," *Proceedings of the 4th International Conference on Autonomous Robots and Agents*, pp. 319-323, 2009.

[18] X. Lu, L. Song, S. Shen, K. He, S. Yu, N. Ling, "Parallel Hough Transform-based straight line detection and its FPGA implementation in embedded vision," *Sensors 13*, pp. 9223–9247, 2013.

[19] R. Marzotto, P. Zoratti, D. Bagni, A. Colombari, V. Murino, "A real-time versatile roadway path extraction and tracking on an FPGA platform," *Comput. Vis. Image Understanding*, pp. 1164–1179, 2010.

[20] X. An, E. Shang, J. Song, J. Li, H. He, "Real-time lane departure warning system based on a single FPGA," *EURASIP J. Image Video Process*, 2013.

[21] J. Wang, Y. Wu, Z. Liang, Y. Xi, "Lane detection based on random hough transform on region of interesting," *Information and Automation (ICIA)*, pp. 1735–1740, 2010.

[22] I. E. Hajjouji, A. E. Mourabit, Z. Asrih, S. Mars, B. Bernoussi, "FPGA Based Real-Time Lane Detection and Tracking Implementation," *International Conference on Electrical and Information Technologies*, 2016.

[23] G. ChapIe, R. D. Daruwala, "Design of Sobel Operator based Image Edge Detection Algorithm on FPGA," *International Conference on Communication and Signal Processing*, vol.14, pp. 788-792, 2014.

[24] C. Saravanan, "Color Image to Grayscale Image Conversion," *Second International Conference on Computer Engineering and Applications*, 2010.

[25] C. Zhang, J. Fang, "Edge Detection Based on Improved Sobel Operator," *International Conference on Computer Engineering and Information Systems*, vol. 52, pp. 129–132, 2016.

[26] H. Lu, H. Aili, "An Improved Edge Detection Algorithm Based on Morphological Operators and Gradient," *J. Journal of Computational and Theoretical Nanoscience*, vol. 12, pp. 1121-1125, 2015.

[27] L. Wang, Y. Shen, H. Liu, Z. Guo, "An accurate and efficient multi-category edge detection method," *J. Cognitive Systems Research*, vol. 58, pp. 160-172, 2019.

[28] M. Rishnan, M. Radha, "Edge Detectıon Technıques for Image Segmentatıon," *International Journal of Computer Science & Information Technology*, vol. 3, pp. 259-267, 2011.

[29] M. Wen, C. Zhong, "Application of Sobel Algorithm in Edge Detection of Images," *China High-tech Enterprise*, pp.57-62, 2008.

[30] D. Bailey (2011). Design for embedded image processing on FPGAs.

[31] J. Canny, "A computational approach to edge detection," *IEEE Transaction on Pattern Analysis and Machine Intelligence*, vol.8, pp.679-714, 1986.

[32] S. Hirano, S. Tsumoto, "Rough representation of a region of interest in medical images," *International Journal of Approximate Reasoning*, Vol.40, pp. 23-34, April, 2005.

[33] J. B. Fasquel, V. Agnus, L. Soler, "A Hierarchical Topological Knowledge Based Image Segmentation Approach Optimizing the Use of Contextual Regions of Interest: Illustration for Medical Image analysis," *ICIP*, pp.777-780, 2006.

[34] J. B. Fasquel, V. Agnus, J. Morea, “An interactive medical image segmentation system based on the optimal management of regions of interest using topological medical knowledge,” *Computer Methods and Programs in Biomedicine*, No.82, pp.216-230, August, 2006.

[35] J. B. Fasquel, M. Bruynooghe, “New hybrid optoelectronic method for fast and unsupervised object detection,” *Optical Engineering*, Vol. 42, No. 11, pp. 3352-3364, November, 2003.

[36] L. Shapiro, G. Stockman, "Computer Vision," *Prentice-Hall*, 2001.

[37] R. Duda, P. Hart, "Use of the Hough Transformation to Detect Lines and Curves in Pictures," *Comm. ACM*, Vol. 15, pp. 11–15, 1972.

[38] P.V.C. Hough, “Machine Analysis of Bubble Chamber Pictures,” *Int. Conf. High Energy Accelerators and Instrumentation*, 1959.

[39] R. L. Rosas, A. de Luca, F. B. Santillan, “SIMD architecture for image segmentation using sobel operators implemented in FPGA technology,” *International Conference on Electrical and Electronics Engineering (CIE ’05)*, pp. 77–80, September 2005.

[40] A. Abbasi, M. U. Abbasi, “A novel FPGA-based architecture for Sobel edge detection operator,” *International Journal of Electronics*, vol. 94, no. 9, pp. 889–896, 2007.

[41] C. Moore, H. Devos, D. Stroobandt, “Optimizing the FPGA memory design for a Sobel edge detector,” *20th Annual workshop on Circuits, Systems and Signal Processing*, 2009.

[42] S. Malmir, M. Shalchian, “Design and FPGA implementation of dual-stage lane detection, based on Hough transform and localized stripe features*,” Microprocessors and Microsystems*, pp.12-22, 2018.

[43] M. Aly, “Real time detection of lane markers in urban streets,” *Intelligent Vehicles Symposium*, pp. 7–12, 2008.

[44] A. Assidiq, O. Khalifa, M. R. Islam, S. Khan, “Real Time Lane Detection for Autonomous Vehicles,” *International Conference on Computer and Communication Engineering*, pp.82-88, 2008.

[45] B. Heisele, W. Ritter, “Obstacle detection based on color blob flow,” *Intelligent Vehicles Symposium*, pp. 282–286, 1995.

[46] R. Okada, Y. Taniguchi, K. Furukawa, K. Onoguchi, “Obstacle detection using projective invariant and vanishing lines,” *IEEE Int. Conf. Computer Vision*, 2003.

[47] R. Labayrade, D. Aubert, J.P. Tarel, “Real Time Obstacle Detection in Stereovision on Non Flat Road Geometry Through "V-disparity" Representation,” *IEEE Intelligent Vehicle Symposium*, 2002.

[48] A. Jose, D. M. Dixon, N. Joseph, S. George, “Performance Study of Edge Detection Operators,” *International Conference on Embedded Systems*, vol.14, pp. 7-11, 2014

[49] A. J. Humaidi, S. Hasan, M. Fadhel, “FPGA-Based Lane-Detection Architecture for autonomous vehicles: a real-time design and development,” *The Asian International Journal of Life Sciences*, vol.16, pp. 223-237, 2018.

[50] C. S. Patel, N. Solanki, N. Tailor, “Analysis of Edge detection using Zynq based SoC FPGA,” *Turkish Online Journal of Qualitative Inquiry*, vol. 12, pp. 1070-1080, 2021.

[51] B. Li, J. Chen, X. Zhang, X. Xu, Y. Wei, D. Kong, “A Design of Zynq-based Medical Image Edge Detection Accelerator,” *International Conference on Biomedical Signal and Image Processing*, pp. 59-64, 2021.

Appendix A

Table A.1 Lane detection results for seven different training sets

Training Sets	# of frames used	# of correct results of Sobel - Prewitt	# of wrong results of Sobel - Prewitt	Efficiency Rate (%) of Sobel	Efficiency Rate (%) of Prewitt
Dashed road lanes	200	193-182	7-18	96.5	91
Arrival-departure and curvy road	200	195-189	5-11	97.5	94.5
Rainy weather and night ride	200	188-164	12-36	94	82
Straight lanes	200	200-191	0-9	100	95.5
Different color lanes	200	196-189	4-11	98	94.5
Foggy weather	200	195-187	5-13	97.5	93.5
More edge lines.	200	188-169	12-31	93.5	84.5

RESUME

Fatih TAT
Çankaya/Ankara, Turkey

EDUCATION

MSc	**Marmara University,** Turkey Electrical – Electronics Engineering (GPA: 3.79)	2018-
BSc	**Antalya Science University**, Turkey (GPA:3.24) Bachelor of Electrical – Electronics Engineering	2013-2018

WORK EXPERIENCE

Research Engineer, Bilkent University NANOTAM (Ankara/Turkey) — Feb 2020 – Working

Design, fabrication and characterization of electro optic devices.
Cleanroom engineer with hands-on manufacturing working
on fiber optic navigation systems and components for navigation systems.

Project Assistant, Marmara University TUBITAK (Ankara/Turkey) — Sep 2018- Aug 2019

Complete simulation work of A Novel MEMS Based
Phase Modulator Device Design Project with using COMSOL.

Internship, Serban Engineering, Vodafone Telecommunication A. Ş (Antalya/Turkey) — Jul 2016- Sep 2016

The control program of fault monitoring system was learned (U2000).
Construction stages of base stations, working principles,
signal calculations of base stations were shown on the projects.

Internship, Imecar Electronic / Antalya (Antalya/Turkey) — Jul 2017- Aug 2017

How to transform gasoline powered car to electrical powered cars, battery packing for electrical cars and charging of electrical cars were worked for different projects.

PUBLICATIONS

Journal Publications

J1. M.İ. Beyaz, F. Tat, R. Özbek, K.Y. Özkaya, "Hybrid Magnetic-Piezoelectric Energy Harvester for Power Generation around Waistline During Gait." J. Electr. Eng. Technol.

vol.15, pp. 227–233, 2020.

Conference Proceedings and Presentations

C1. F. Tat, S. Saglam, S. Bayar "FPGA Implementation of CNN Algorithm for Detecting Malaria Diseased Blood Cells" International Symposium on Advanced Electrical and Communication Technologies (ISAECT) conference, 2019.

C2. E. Demirpolat, F. Tat, G.G. Aktaş, E.E. Altinisik, O.K Cayir, "Design and Simulation of Band stop Filter using Sonnet Software" Computational Methods and Telecommunication in Electrical Engineering and Finance, 2018.

ACHIEVEMENTS AND CERTIFICATES

Graduation, Antalya Science University June 2018

TUBITAK Efficeny Challenge Electrical Vehicle Jan 2017 - Sep 2017

The car we built was completed, participated in the race and the competition was completed.

COMSOL Multiphysics Intensive Training Oct 2018- Oct 2018

Finite Element Analysis Theory, Basic Modelling Settings, Network Mesh Definitions, Processing of CAD Data, Working with Conjugate Variables, Solver Settings, Modelling with Free Equations, Applications Specific to the Field of Application.

PROJECTS AND COMPETİTİONS

An FPGA Implementation of Real-Time Lane and Obstacle Detection for Driving Assistance System **Feb 2109 -Working**

Cancer Image Classification using CNN Algorithms on MATLAB **March 2019 - April 2019**

Implementation of Designed 32 Bit Processor using VHDL **Feb 2019 - March 2019**

Implementation of Algorithms on Cores Using VHDL With Vivado **Jan 2019-Feb2019**

Making Turtlebot Follow the Desired Paths Using ROS Program **Now 2019 - Dec 2019**

A Novel MEMS Based Phase Modulator Device Design **Sep 2018- March 2019**

TUBITAK Project, *Senior Project* **October 2017 - June 2018**

Electromagnetic / Piezoelectric Based Hybrid Energy Harvesting Belt for Charging Portable Electronic Devices.(Supported by TÜBİTAK 2209-B Program)

TUBITAK Efficiency Challenge Electrical Vehicle **2016-2017**

Design of the car, battery management system, battery packing and electrical connection of the car were performed.

COMPUTER SKILLS

Excellent: Python, NI Multisim, MATLAB, COMSOL, U2000

Good: C, Proteus, Keil, Vivado, VHDL, Verilog, FPGA, Eagle, PCB Design, Analog Circuit Design

Familiar with: OCaml, Prolog

LANGUAGES

Turkish (Native), **English** (Fluent), **Russian** (A2), **Germany** (B1)

ACTIVITIES

Represent of Antalya Bilim University, Chambers of Antalya Electrical Engineer (EMO) 2015-2018

Represent of Antalya Bilim University Electrical and Electronics Departments 2016-2018

Represent of Antalya Bilim University Electrical and Electronics Head Boy 2016-2018

Active worker of Tubitak Efficiency Challenge Electrical Vehicle Club 2016-2018

Member of Institute of Electrical and Electronical Engineers (IEEE) 2014-2018