17th International Conference on
AEROSPACE SCIENCES & AVIATION TECHNOLOGY,
ASAT - 17 – April 11 - 13, 2017, E-Mail: asat@mtc.edu.eg
Military Technical College, Kobry Elkobbah, Cairo, Egypt
Tel: +(202) 24025292 – 24036138, Fax: +(202) 22621908

Implementation of Vision-Based Trajectory Control for Autonomous Vehicles

A. Desoky[*], A. Bayoumy[†], G. Hassaan[‡]

Abstract: This paper demonstrates building, implementing, and developing a trajectory tracking control system based on computer vision for autonomous vehicles. The main goal of this system is to enforce the autonomous vehicle to be able to track road lane. This system includes a single digital camera, an embedded computer, and a microcontroller board. The digital camera is mounted at the top of the vehicle along its longitudinal axis. It captures a real-time sequence of images during vehicle motion. The captured images are then processed using Open-CV library for Python compiler over Linux operating system. These software packages are running on the embedded computer (Raspberry Pi 2) to obtain geometrical data of road lane. From this data, the observable errors can be determined. These errors are vehicle lateral offset and a heading error. Finally, a steering controller utilizes these errors in control law to compute the steering command. This command corrects offset and heading errors to ensure that the vehicle is in its way. The embedded computer then paths this command to Arduino microcontroller board to adjust the steering servomotor. The proposed implementation also demonstrates the integration between the embedded computer and microcontroller using Ethernet. During this work, a set of autonomous driving experiments is performed. Significant results are obtained that demonstrate the accuracy and robustness of the lane detection and control algorithms.

Keywords: Vision-based, Lane detection, Autonomous Vehicle, Raspberry Pi, Arduino.

1. Introduction

In past decades, the number of motor vehicles in the world is increasing gradually. The official investigation reports of traffic accidents point out those dangerous driving behaviors, such as drowsy and drunk driving, have taken a high percentage between all the accidents reasons [1]. So as to evade this kind of unexpected accidents, it is essential to develop an appropriate set of autonomous vehicle systems that can directly improve the safety of driving. However, several complex issues are involved with keeping an eye on drivers directly all the time to remove all possible threats. Actually, human behaviors are really hard to recognize, predict and handle by present available equipment. So, monitoring and warning system focusing on the vehicle behaviors is required while the vehicle is moving on the road.
Moreover automobile manufacturers have developed and are ongoing to develop systems for vehicles that decrease the driver's responsibility to monitor and control on the vehicle. In the field of transportation systems a large highlighting has been given to issue such as improving safety conditions, optimizing the utilization of transportation systems, reduce energy consumption and environment protection from pollution. The efforts in solving those

[*] MSc Student, Faculty of Engineering, Cairo university, Giza, Egypt, ahmeddesokyeg@gmail.com
[†] Assistant Professor, MSA University, Giza, Egypt, ambayoumy@msa.eun.eg
[‡] Emeritus Professor, Faculty of Engineering, Cairo university, Giza, Egypt, galalhassaan@ymail.com

problems have caused the interest towards a new field of research and application such as autonomous vehicle driving, in which new techniques are probed for the whole or partial automation of driving tasks. These tasks involve tracking the road and keeping within the correct lane, preserving a safe distance among vehicles, controlling the speed of the vehicle according to traffic conditions and road features, moving across lanes so as to pass vehicles and avoid obstacles.

Autonomous vehicle are a type of vehicle that performs the required tasks without human guidance. Researchers are done to create such vehicle that can cope with the several of environment, typically, on land, underwater, air borne, underground, or in space. From there, the vehicle would take over and drive to destination without human input. The autonomous vehicle is one in which a computer executes all the tasks that the human driver normally would. Finally, this would mean getting a vehicle, entering the objective into a computer, and enabling the system. From there, the vehicle would take over and drive to destination with no human input. The vehicle would be able to sense its environment and make steering and speed changes as necessary. So to develop an autonomous vehicle it will involve automated driving, navigating and monitoring systems.

This problem would require all of the automotive technologies such as lane detection to assist the vehicle to pass inside the lane, obstacle detection to locate other vehicles, pedestrian, animals, etc., collision avoidance to evade striking obstacles in the road, cruise control to keep a safe speed, and lateral control to maintain the vehicle's position on the roadway. Thus, sensors will be a major component to develop these technologies. The vehicle would be able to sense its environment to make steering and speed changes as necessary. Therefore, to develop the autonomous vehicle it will involve automated driving, navigating and monitoring systems.

In the last years, almost all vehicle manufacturers are busy developing some sort of self-driving vehicle that can maneuver in real world traffic [2]. Many projects on autonomous driving are in development in industry. In 1980, Mercedes-Benz was the first one to develop the autonomous road vehicle. Recently there are many researches until now about the autonomous driving systems used a lane-keeping vision system to detect the road lane and compute road information such as heading angle and offset displacement, which are then sent to the controller operating the forward steering wheel to ensure that the vehicles driving in the inside lane. The process of estimating the vehicle position and its heading direction inside the lane is essential for many autonomous vehicle applications. The developments of techniques of the navigation and control of the autonomous vehicles have become an active research topic. The autonomous navigation vehicle is the capability of moving on the required path without any intervention from human.

1.1 Related Work

We will investigate some of the major research projects in the field of automated driving, with a strong focus on sensing. Automotive companies carry out most of the research in the field of automated driving; hence, there is not much public information available.

In 2004, DARPA of the US Department of Defense offered a one million dollar as a prize to the first autonomous vehicle which able to travel a 150 mile through the Desert [2]. The vehicle was required to be fully autonomous without humans allowed to be in the vehicle during the competition. Stanley won the competition. Buoyed by this success, DARPA announced a new challenge, which was held in 2007 named the Urban Challenge. The autonomous vehicles those were able to drive cross-country. Those vehicles have to perform regular driving tasks such as turning and keeping headway, as well as special maneuvers such as parking, U-turns, etc. Fig. 1 shows the Boss vehicle that won the $2 million prize. It uses 11 LIDAR sensors (4 different types), 5 long-range dynamic RADARs (Continental ARS300), and 2 cameras.

Fig. 1. The won vehicle 'Boss' DARPA urban challenge in 2007[2]

There has been a valuable amount of research on road lane detection and tracking based on vision. Vision-based localization of the lane boundaries can be divided into two subtasks: lane detection and lane tracking. Most of the vision based lane detection systems in follow quite a similar work-flow:

1) Capturing the image from the camera,
2) Cropping the image to obtain appropriate Region of Interest (ROI),
3) Detecting the lines (the actual lane detection process),
4) Fitting straight line or curve models to the detected lane markings.

The image is converted into gray-scaled image, often followed by thresholding. It is a technique by which a gray-scaled image retains only the pixels that have gray-scale intensity higher than a threshold value. All the pixels with less intensity will be assigned a zero value. To detect the edges in the image that would give clues regarding where the lane markings can be, a Sobel filter or a Canny edge detector[3] , [4], is generally used.

In[5] the images are first converted into gray-scale using only the R and G channels of the color image and ignored the B channel relying on the good contrast of red and green channels with respect to the white and yellow lane markings. The gray-scale image is passed through a very low thresholded Sobel edge detection.

In [6] the images is filtered using optimized Gaussian filters. This filtered image is then is thresholded. Edge detection is a main tool in computer vision, and image processing, which attempt to identify points in the image by changing the image brightness sharply.

One of the supreme valuable controllers is a proportional, integral and derivative PID controller. A proportional, integral, and derivative controller PID controller [7, 8] as shown in Fig. 2 is a standard controller with loop feedback commonly used in control systems, A PID is the supreme commonly used the controller with feedback from the process. A PID controller computes the value of the error as the change between the desired set point and the measured feedback variable from the process. The controller endeavors to minimize the error by attuning the control inputs of the process.

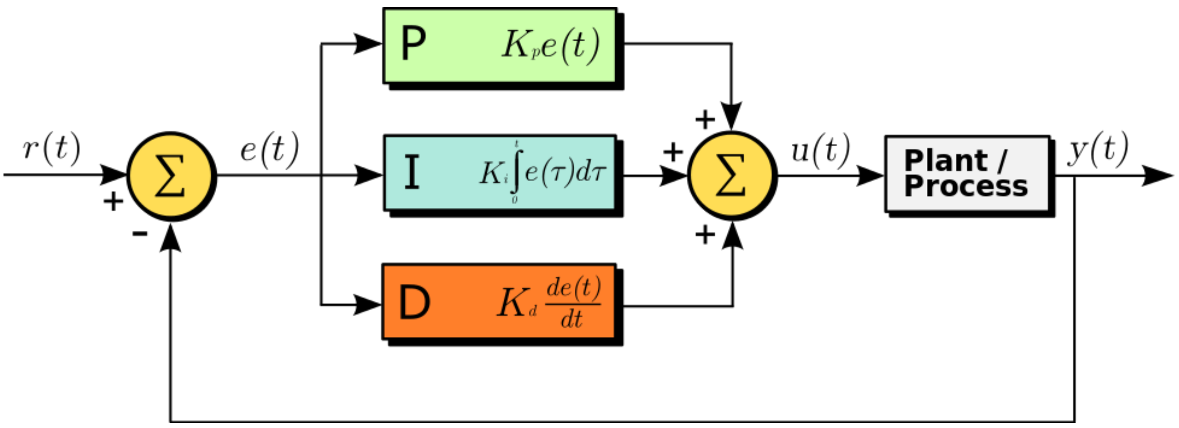


Fig. 2. A block diagram of a PID controller in a feedback loop

The textbook version of the PID controller is

$$u(t) = ke(t) + k_i \int_0^t e(\tau)d\tau + k_i \frac{de}{dt} \quad (1)$$

In [9] it presents an experimental results of an autonomous vehicle. This vehicle is able to detect lane and tracking it by using digital camera. The system uses the computer vision techniques in real time are collected by a single digital camera. Further processing and analysis for the images captured by the digital camera are accomplished to recognize the lane lines. In [10] an intelligent vehicle control system is designed and embedded in a Digital Signal Processing (DSP) platform (eZdspTMF2812).whereas the system is tested and evaluated on a university campus. The installation platform is golf car to carry the overall system, including steering wheel serve motor, throttle driving circuit brake actuator, and sensors. The technology of digital image processing is used to enable the autonomous driving system for achieving lane keeping multi-mode.

1.2 Contribution

This paper has four contributions:

1) Prepare an autonomous vehicle with camera, steering servo motor and DC motors to actuate steering wheel and the driving wheels. All necessary electronic components, control unit and power supply are applied. This is for the purpose of practical implementing of the vision system.
2) Improving the vehicle steering control system by detecting the driving lane using computer system.
3) Implement the Python with Open CV to develop and identify the road lanes with higher accuracy and minimal computational time.
4) Building the stand-alone real-time autonomous vehicles using a portable computer to detect and track lane line.

2. Hardware Implementation

2.1 System Overview

In (Fig. 3) the proposed hardware subsystem consists of three main subsystems namely; computer vision system, digital control system, and vehicle system.

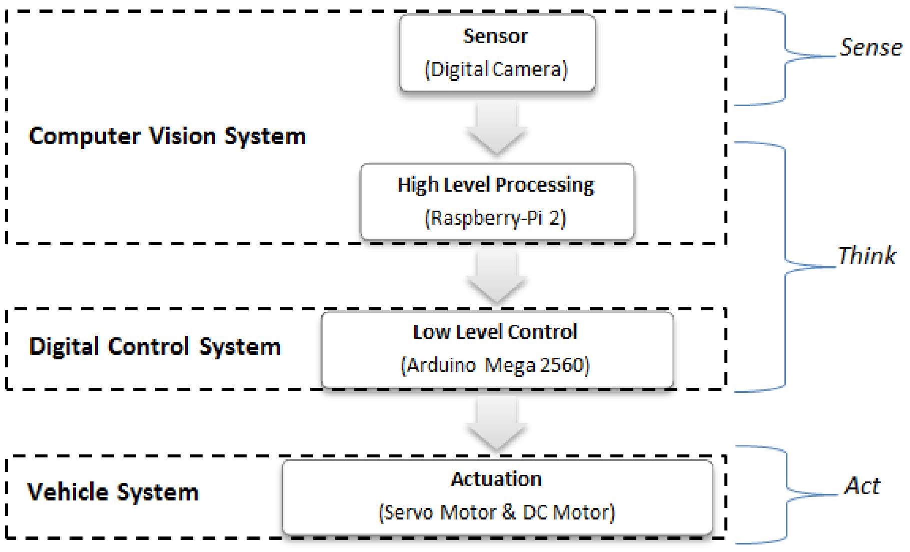


Fig. 3. Overview of the vehicle's hardware

The computer vision system consists of digital camera and embedded computer , the digital camera performs the necessary image processing. Further processing is carried out to

calculate the heading angle error and lateral offset distance and send the command message to the digital control system. Digital Camera captures the image of the real scene of the environment then sends it to the USB port of the Raspberry Pi board.
The embedded computer (Raspberry-Pi 2) is responsible for calculating the desired steering angle from computer vision system and based on the difference between the desired and measured steering wheel angles, the digital control system corrects the direction of the vehicle.
The vehicle system is the experimental platform as shown (Fig.4) consists of a miniature vehicle in scale 1:16. It is equipped with brushed dc motor for driving the rear wheels and servomotor for two front steering wheels. It is still conveniently small for driving indoors in order to use the vehicle indoors without risking crashes. The experimental platform features are represented in (Table 1), also vehicle servo and DC motors features are represented in (Table 2 and Table 1)

2.2 System Architecture

As shown in (Fig.5) system architecture block diagram that demonstrates the relation between the hardware components. The sensor (digital camera) is fixed on front and top along the central line of the experimental vehicle and acquires snap shots (frames) of the real scenes. The acquired frame is transmitted by USB cable to the Raspberry Pi 2 board to be analyzed by using the Open-CV library and Python. After analyzing the image, the required heading angle is defined then could be sent to the Arduino board via Ethernet cable. The Arduino board sends the command signals to the steering servomotor through the relay-board.

Table 1. Experimental Platform features

Parameters		Value	Units
1	Brand	Kyosho	
2	Vehicle mass (loaded)	2.5	[Kg]
3	Vehicle width	24	[cm]
4	Vehicle length	38	[cm]
5	Wheel base (L)	26	[cm]
6	Wheel track (b)	20	[cm]
7	Wheel radius (r)	45	[mm]
8	Tire width (w)	30	[mm]
9	Max Wheel Steering angle	±25	[degree]
10	Vehicle speed (v)	0.2	[m/s]

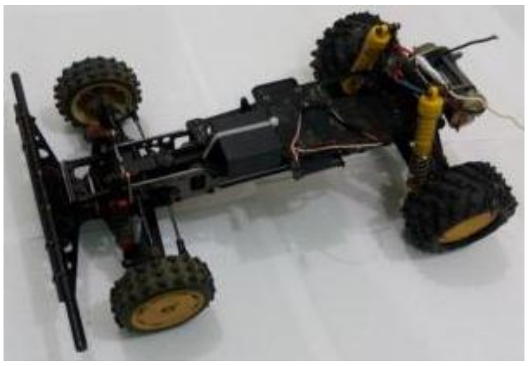


Fig.4. Experimental platform

Table 2. Vehicle Servo and DC motors features

Servo Motor				
1	Brand		Futaba	
2	Model		FP-S129	
3	Operating Volt		4.8 or 6	[Volt]
4	Max. Torque		3.5	[Kg.cm]
5	Max angle		±60	[degree]
6	Weight		60	[gram]
7	Dimension		45.5×23×43.5	[mm]
DC Motor				
1	Brand		MABUCHI	
2	Model		RS-540RH-6035	
3	Type		Carbon-brush Motor	
4	Voltage	Operating Range	3.6 ~ 7.2	[Volt]
		Nominal	7.2	[Volt]
5	No Load	Max. Speed	9500	[rpm]
		Current	1.15	[Amp]
6	Max. Efficiency	Max. Speed	7700	[rpm]
		Current	4.91	[Amp]
		Torque	125	[g.cm]
		Output	9.88	[Watt]
7	Stall	Torque	660	[g.cm]
		Current	21	[Amp]

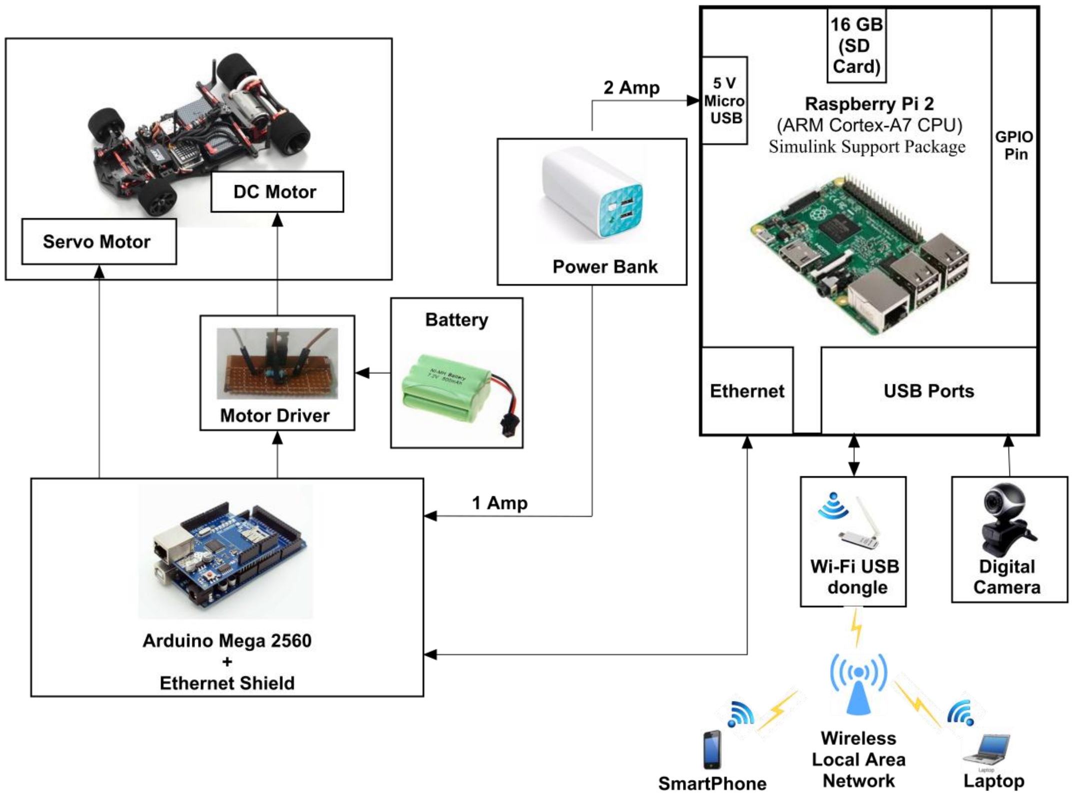


Fig.5. System Architecture Block Diagram

In order to investigate the applicability of the proposed system, the experimental vehicle (platform) is prepared with the appropriate systems as shown in Fig. 6.

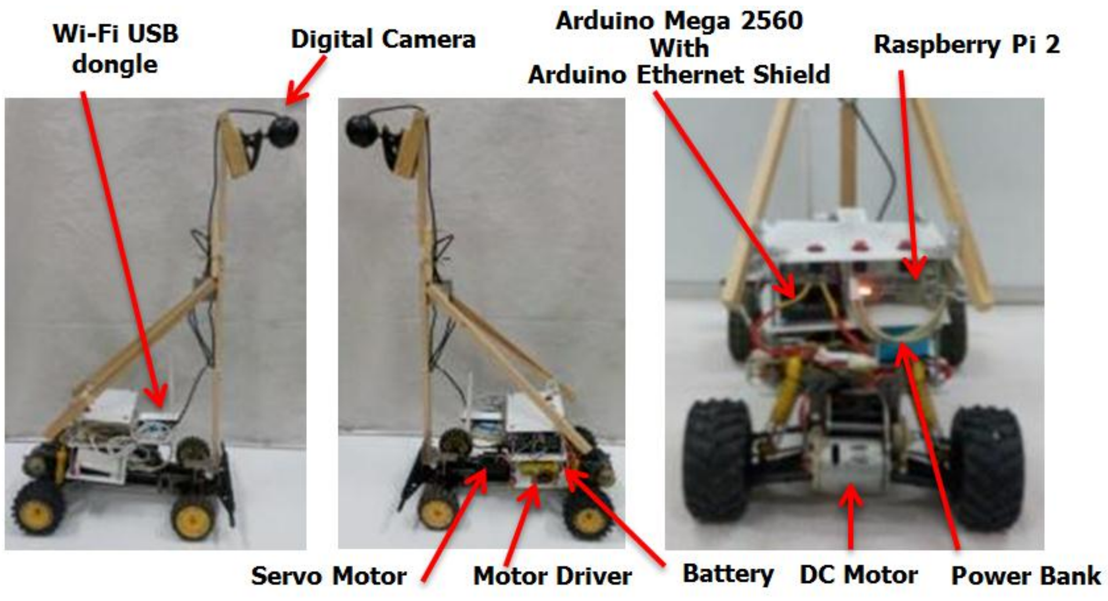


Fig. 6. Vehicle with System Components

The main features of the hardware subsystems are explained in brief as following:

1) Raspberry

Raspberry Pi [11] is a low-cost portable computer operated in Linux operating System. It is an embedded computer has an ARM processor is clocked in at 900MHZ, and 1GB of RAM. The most advantage thing regarding ARM processor is it consumes low power via Micro USB connection to power itself. It supports Ethernet cable to access the network, USB cable to interface with peripheral devices adding (camera/Wi-Fi module/dongle).

2) Arduino Board

It is a programmable circuit board which can get inputs from varieties of sensors and switches, and can control others physical devices. Program is uploaded in the board via IDE in a computer using USB cable. There are different types of Arduino board available in the market and Arduino Mega 2560 is selected.

3) Arduino Ethernet Shield

Arduino itself cannot connect to network it needs Ethernet shield. The shield should be mounted over the Arduino board for the internet connection using Ethernet library.

4) Digital Camera

The sensor (digital camera) is fixed on front and top along the central line of the experimental vehicle and acquires snap shots (frames) of the real scenes. The acquired frame is transmitted by USB cable to the Raspberry Pi 2 board. The camera features are represented in (Table 3).

5) DC Motor Driver Circuit:

DC motor requires high current about more than 1 Ampere. Arduino board (Microcontroller) cannot supply this amount of current. So, it is not permissible to directly connect motor to the output of any of the Arduino PWM port, which might get damaged.

Table 3. Camera features

1	Type	USB Camera	
2	CMOS chip type	Color CMOS image sensor	
3	Video format	24 bit RGB	
4	Interface	USB 2.0	
5	Frame rate	320x240 up to 30	[frames/sec]
6	Definition	0.3	[Megapixel]
7	Average Half Vertical FoV (α)	10.6	[degree]
8	Average Half Horizontal FoV (β)	14.2	[degree]
9	Camera Inclination Angle(γ)	50	[degree]

The MOSFET is a kind of transistor and stands to (metal oxide semiconductor field effect transistor). It can be applied to the electronic circuits to switch or amplify electronic signals. It has four terminals; body, source gate, and drain terminals; the body is connected internally to the source terminal. So, the three terminals show in (Fig.7) in the electrical diagram. The major benefit of the MOSFET is very low current (less than 1mA) required to turn on, while the load is supplied by higher current (10:50A or more).The MOSFET is controlled by applying voltage at gate terminal. It can be used for high speed switching, very low voltage applications in power supplies, and power motor controls and bridge circuits. The typical applications are (power supplies/ converters/ power motor controls/ bridge circuits) [12] .

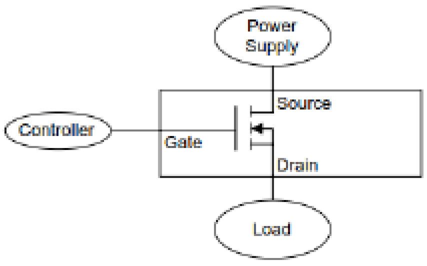


Fig.7. N-channel power MOSFET

There is a need of a circuit that can act as a bridge between the Arduino PWM port and the DC motor. Here MOSFET is used as a switch to turn a motor on or off depending upon the applied voltage at the gate. Its circuit is shown in (Fig.8, Fig.9, and Fig.7).

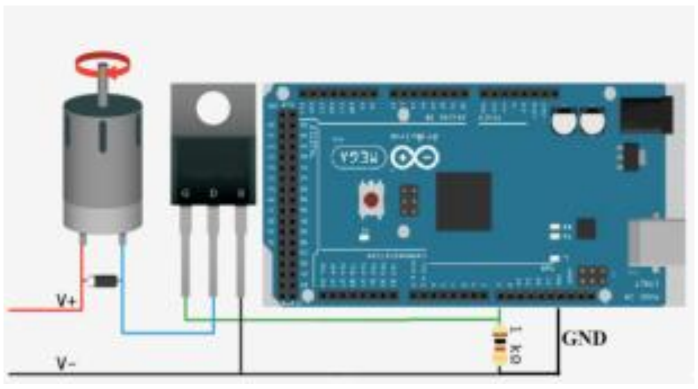


Fig.8. DC Motor driver circuit block diagram

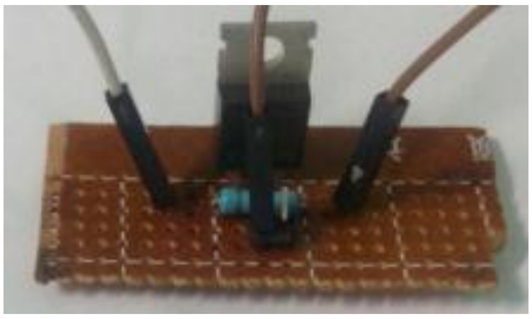


Fig.9. DC Motor driver circuit

6) Power supply system:

The first power item is power bank which is used to provide the required power to Raspberry Pi 2 and Arduino board. The second power item is NiMH battery that is used to provide the required power to the driving DC motor.

3. Software Implementation

Both boards Raspberry-Pi 2 and Arduino Mega 2560 require software as follow:

1) Raspberry-Pi 2:

The image processing algorithm builds around ARM processor of Raspberry Pi 2 which is the heart of the system. It uses Simulink hardware support package supporting the Raspberry-Pi 2 as Linux system, then using Python language with Open CV and the others library.

2) Arduino Mega 2560 board:

It uses Arduino programming language as integrated development environment (IDE).

3.1 Microprocessor Software Implementation

3.1.1 Installation of Operating System on Raspberry Pi

Raspberry Pi is a small computer; hence operating system (OS) should be installed. As the Raspberry doesn't have hard drive, OS is installed in the external memory. For that, memory card (SD card) is used for the installation of operating system and all the required software and supporting files are stored in the same SD card. The proposed work implement Simulink support package on Raspberry Pi 2 model B. This package support enables development software for algorithms which can run in Raspberry Pi.

Moreover, it allows controlling peripheral devices that connected via its GPIO interfaces such as serial, I2C and SPI by using command functions. Simulink support package allows users to create Simulink models in various fields such as audio, image or embedded system using general and support package toolboxes. Furthermore, Raspberry Pi can be used to create applications in standalone mood by deploying Simulink model onto it. MATLAB/Simulink support package can be set up for Raspberry Pi according to [13].

3.1.2 The Proposed Lane Detection Algorithm:

The aim of the image processing is to obtain a processed image for each original captured image; this processed image can give information about the position of the vehicle such as its orientation and the lateral distance with respect to the centerline of the road lane at a particular look-ahead distance.

There are two main stages are implemented the first stage is the pre-processing stage and then the second stage is the lane detection stage. The main aim of pre-processing stage is to get a gray image and remove its noise. The major aim of lane detection is to detect the lane lines in front of the vehicle to obtain the lateral offset distance and the lane lines angles with respect

to the vehicle. The autonomous driving system is equipped with a vision system is based on the real-time data of images sequences taken from a vehicle driving on the road.
The vision system includes a digital camera and an embedded platform. Whereas the digital camera captures the image and the embedded platform performs image processing. (Fig.10) demonstrate block diagram for the image processing function.

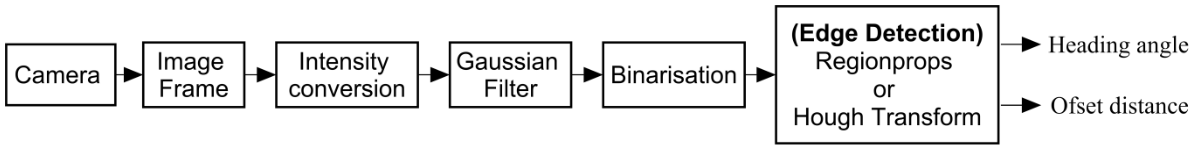


Fig.10. Image processing algorithm block diagram.

The captured image passes through the following steps as shown in Fig.10 and Fig. 11 whereas the original image is converted into a suitable image format to apply into regionprops method as follow:

1) Intensity conversion: The captured image is converted from RGB format to grey-scale.

2) Smoothing: The grey-scale image is smoothed to remove fine edges. The filter used is a standard 5×5 Gaussian.

3) Binarisation: Otsu's threshold method is used to convert the image pixels from grey to binary as clear black/white image. The result is an image that has the road markings highlighted against the black background of the road. Edge features to the sides of the road are also highlighted but this does not pose a problem.

4) Regionprops [14] calculate a variety of image features besides quantities from a black and white image. Regionprops produce some specific properties of the detected lines on the image such as centroid, line orientation angle, line length, and the line two terminal points. One of these specific properties is the centroid. The centroid is also the center of mass. It is the "middle" of the lane line. This would be the (x, y) locations of where the middle of each lane line is located.

Therefore, from the centroid, orientation angle, and length of each lane line, the final represented line can be plotted over the original lane lines.
Regionprops method [14] depends on contours that is as a curve joining all the continuous points (along the boundary), having same color or intensity. The contours are a useful tool for shape analysis and object detection and recognition.
Contour features is used to find the different features of contours, like area, perimeter, centroid, bounding box, major axis and minor axis lengths, and orientation that is the angle at which object is directed.
The contour features can use a function called image moments used to calculate some features like center of mass and area of the object.

3.1.3 Algorithm Using Raspberry Pi Simulink Support Package

The Simulink library browser is a collection of high level blocks that we can use to create a block diagram demonstration of the system to be easy to design. From a different perspective, these blocks allow us to access or generate, apply algorithms and visualize or save the processed data or information, which flows through the system.

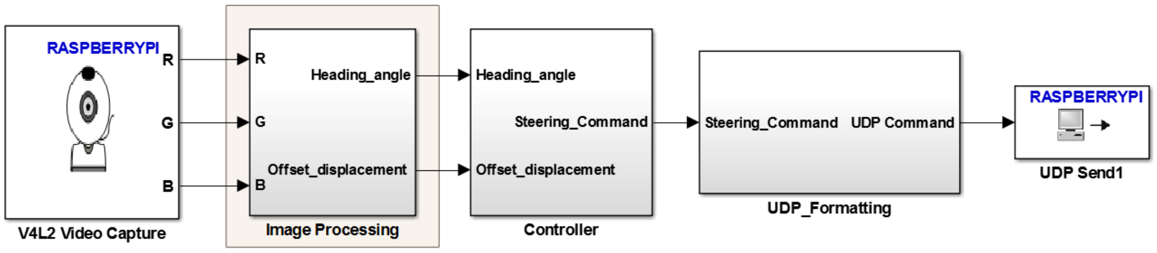


Fig. 11. Line detection Simulink

Fig. 11 shows line detection Simulink model using Hough Transform or regionprops (contour properties / blob analysis).
Using contour properties (Blob Analysis) yield same result as Hough transform method at running Simulink model in normal run mode.

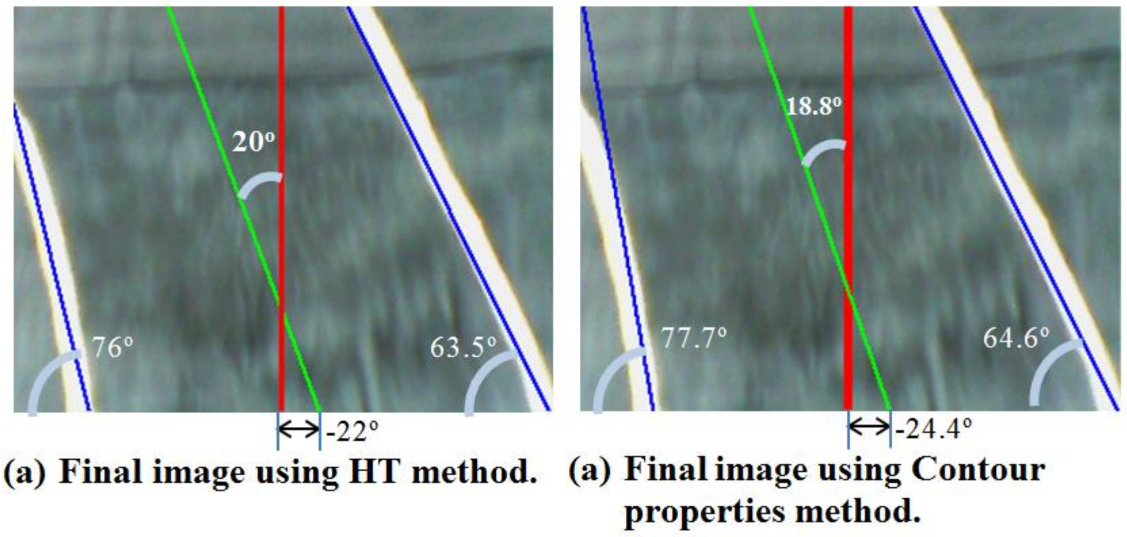
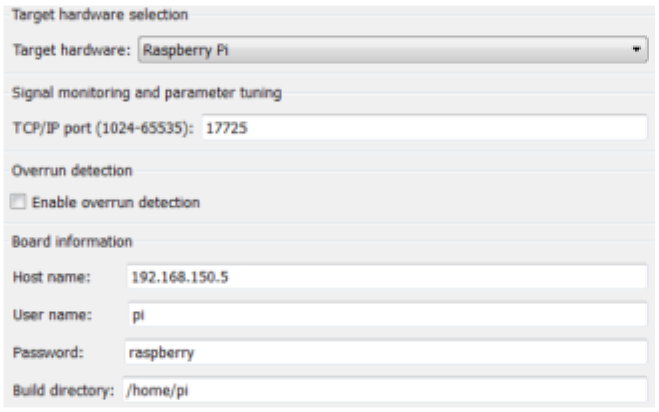

20°
76°
63.5°
-22°
(a) Final image using HT method.
18.8°
77.7°
64.6°
-24.4°
(a) Final image using Contour properties method.

Fig.12. Final Image using two methods by Simulink model

At building Simulink model algorithm on Raspberry Pi, Firstly, Confirming the information of the target hardware, in this case, Raspberry Pi, in a configuration window shown in (Fig.13). Ensure that an IP address, a user name and password are correct since these parameters are used to establish the connection.


Target hardware selection

Target hardware: Raspberry Pi

Signal monitoring and parameter tuning

TCP/IP port (1024-65535): 17725

Overrun detection

☐ Enable overrun detection

Board information

Host name: 192.168.150.5

User name: pi

Password: raspberry

Build directory: /home/pi

Fig.13. Run on target hardware configuration window

The External mode is selected from a drop-down menu as shown in (Fig.14). The desired running time input onto a space next to the mode option. The default is 10. In this case, it is set as *inf* to indicate the infinity running time. The simulation process begins by clicking a run button as shown in (Fig.15). The model run and displays the result on a computer's screen.

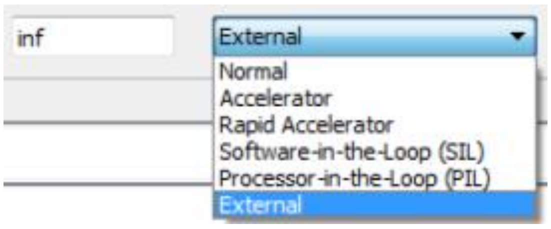


Fig.14. External mode option

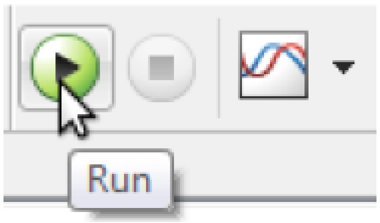


Fig.15. Run button

For deploying a Simulink model onto Raspberry Pi, after having confirm the information on a configuration window, a model can be deployed onto the board by clicking a deploy to hardware button shown in (Fig.16). The Simulink model will be uploaded onto the default folder of the board.

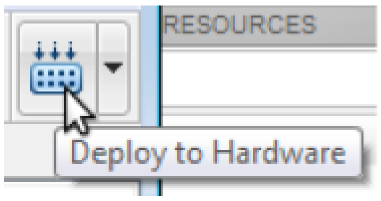


Fig.16. Deploy a Simulink model onto Raspberry Pi

But, in more than 20 trails for running and deploying the Simulink block algorithm on Raspberry Pi board, the Simulink algorithm is very sluggish and lazy and has time delay in the response of steering servo motor about 6 second in average, So, Python algorithm has choice.

3.1.4 Python and OpenCV library preparation

Python is a general purpose programming language began by Guido van Rossum [15], which turn into very famous in short period due to code simplicity. Python has large standard libraries to perform many tasks such as Numpy that is a highly optimized library for numerical operations and makes the task easier. In addition to SciPy and Matplotlib which reinforces Numpy. In other hand, there is a very important library that deals with computer vision namely Open-CV library, which is an open source computer vision. This library is written in optimized C and can take advantage of multi core processors that executed under Linux, Windows, Mac OS, and Android etc. Open-CV supports a wide variety of programming languages like C++, Python, Java, Ruby, MATLAB, and other languages. Gary Bradsky at Intel began Open-CV in 1999 and the first release was in 2000. Vadim Pisarevsky united Gary Bradsky to manage Intel's Russian software Open-CV team [14]. In 2005, OpenCV was utilized on Stanley, that vehicle won 2005 DARPA Grand Challenge. OpenCV

boosts many algorithms related to Machine Learning and Computer Vision and it is expanding day-by-day.
The required library can be installed in Raspberry Pi for the completion of the proposed image-processing algorithm.

3.1.5 Algorithm using Python and OpenCV library

The captured image goes through the steps that mentioned in (section 0) to convert it into a format suitable for input to the Regionprops. The effect of each step is illustrated in (Fig.17).

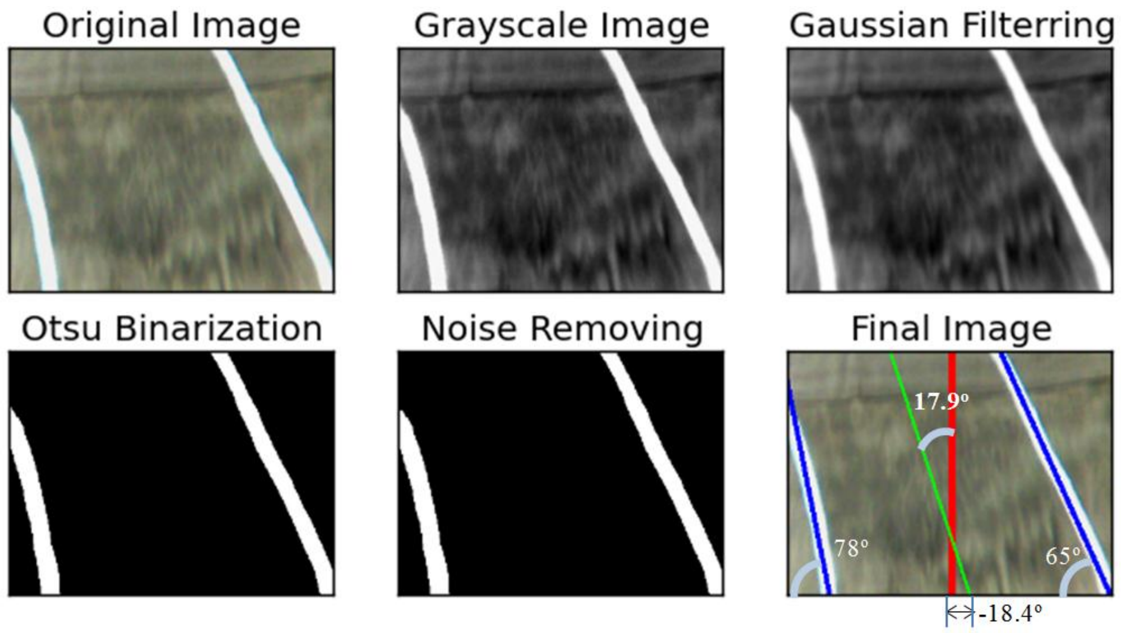


Fig.17. Image preprocessing steps

Fig. 18 shows using regionprops method do detect curved path, and the centroid of each lane line and bounding box are plotted on original image subplot.

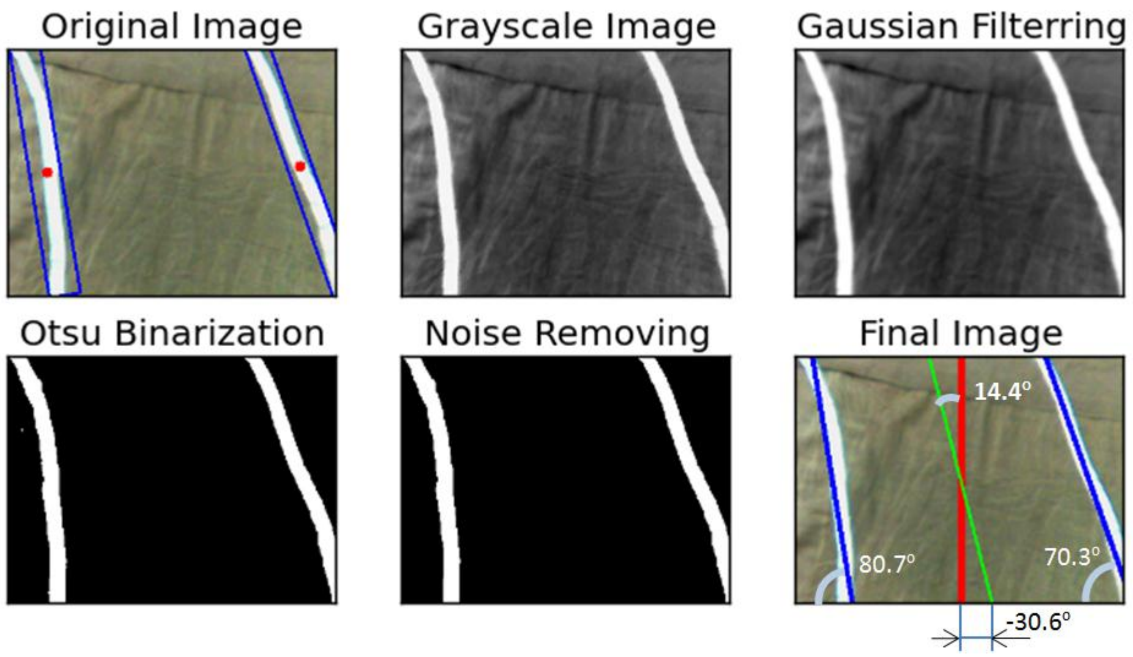


Fig. 18: Image preprocessing example by using regionprops at curved path

3.2 Design of the Lateral Controller

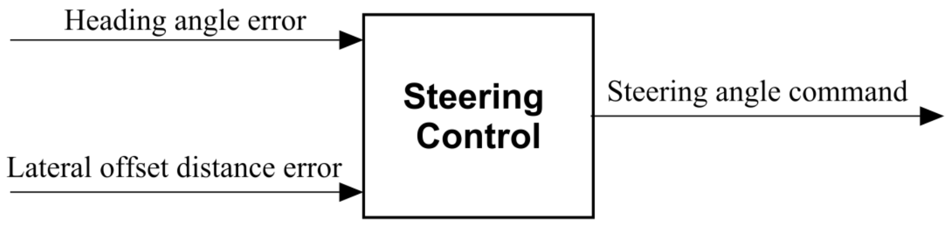


Fig.19. Control system block diagram

This section is concerned with the vehicle navigating along a specific trajectory. The steering control as shown in (Fig.19) is not only involves ensuring that the vehicle is in the correct position but also has the correct heading.
The controller is designed by successive loop closure technique as shown in (Fig.20). The basic idea behind successive loop closure is to close several simple feedback loops in succession around the open-loop plant dynamics rather than designing a single (presumably more complicated) control system [16]
In all these controllers, the outer loops control the slower variables and the inner loops control the faster ones.
The outermost loop tries to rectify, through feedback, the error between desired and actual values of the slowest changing variable by computing through proportional blocks what the desired value of faster variable should be. The inner loops handle errors in the faster variables with the innermost loop dealing with the fastest variable.

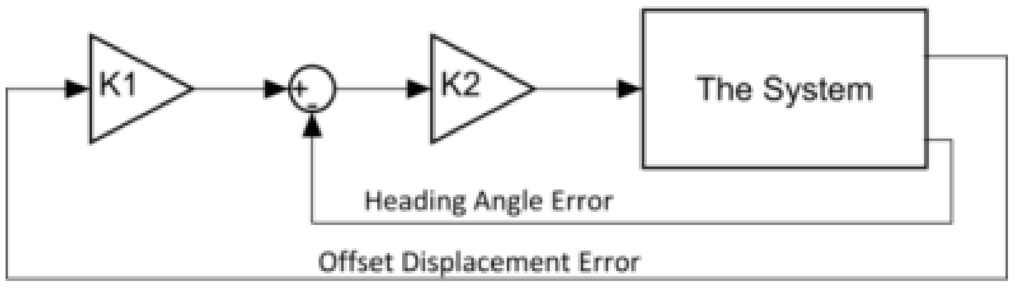


Fig.20. Successive Loop Closure Diagram

3.3 Data Type transfer from Raspberry Pi to Arduino

The Arduino Ethernet shield (target hardware) is connected to the network with Raspberry Pi 2 (host hardware) with an Ethernet cable via the UDP protocol (Universal Datagram Packet). IDE of Arduino board is received UDP messages from a remote host, identified with a unique IP address and port number via an Ethernet cable
UDP packet is sent from Raspberry Pi 2 to Arduino board in form int8 data type that is varying from -127 to 127. The proposed method uses the number from 000 to 127 as an UDP packet. the first number of the message indicates to the condition (off/on) of the driving DC motor whereas 0 and 1 indicate to off and on respectively. the second and third number of the message indicate to the required image command angle value that is mapped to suitable value from 00 to 27 before sending to Arduino board, then the message that received by Arduino board is mapped to be suitable for the required servo command angle. The message is mapped by applying the following equation and the values that are shown in (Table 4).

$$y = y_a + (y_b - y_a)\frac{x - x_a}{x_b - x_a} \qquad (2)$$

Table 4. The message mapping

	The sending message by Raspberry Pi			
		The received message by Arduino		
Wheel Position	Image heading angle limitation	Steering command limitation	Angle limitation for servo motor	Angle limitation for vehicle wheel
Right	-30	00	125	-25
Middle	0	13.5	90	0
Lift	30	27	55	25

3.4 Microcontroller Software Implementation

Arduino programming language to program the Arduino Mega 2560 the Arduino IDE has been used. The Arduino program is written in C and uses the wiring framework to make the I/O operations easier. The Arduino code receives the message from Raspberry Pi via Ethernet and decrypts the message to give the required steering angle command to the servo steering motor and power (on/off) to the DC motor.

4. Results

During field tests all the images were taken by the webcam are stored in the computer for testing them again inside the lab and re-test the experimental without the need to take place within the field of test. Fig. 21 shows the sample of images stored during field testing. And Fig. 22 shows that the vehicle moving between the lane in real experiment.
The proposed algorithm that is implemented on the embedded computer (Raspberry Pi 2) can perform the following tasks:

1. Capture the image by rate 4:6 frame/sec.
2. Image processing.
3. Getting offset distance and heading angle errors.
4. Recording all captured images.
5. Recording a text file (.txt) contained all important data of each capture images such as:

A. Length, angle, and intersection points of the lane lines with image frame.
B. Offset distance errors, heading angle errors, and steering command.

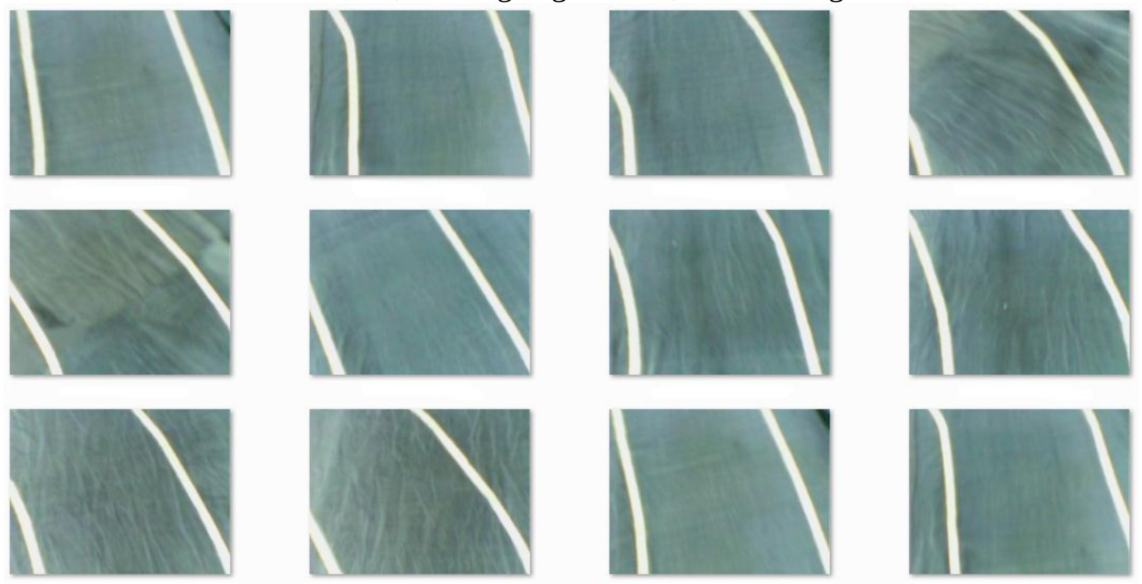


Fig. 21. Sample of successive images which is stored during field testing

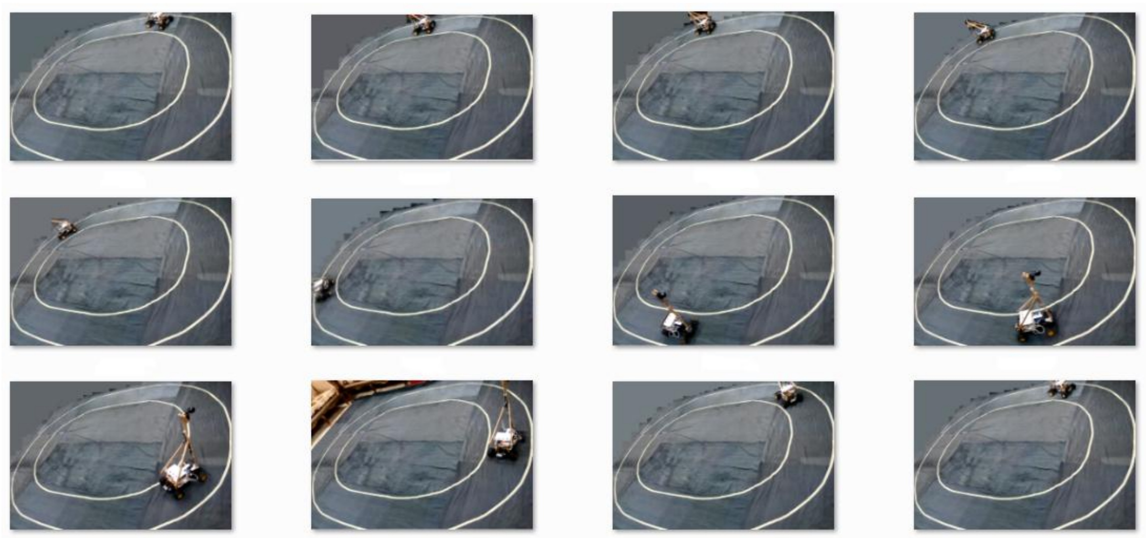


Fig. 22. The vehicle moving between the lane lines in real experiment

The computational time of the proposed system for lane detection and tracking is about 0.147 sec per frame per second. And this is not a problem as the experimental vehicle has a small velocity 0.1 m/s as shown in Table 5.
The computational time calculated for lane detection by Nasim Arshad et al [17] is about 0.2 sec. greater than author calculated computational time (0.147 sec.).
And for algorithm introduced by Xiaodong Miao et al [18] is 0.0756 sec. less than author calculated computational time (0.12 sec.) but with take into consideration embedded system is used by Xiaodong Miao.

Table 5. Vision algorithms execution time for each frame

Image processing steps for each frame	**Time**	Unite
Gray	0.001	[Sec]
Gaussian Filtering	0.005	[Sec]
Binarization	0.001	[Sec]
Noise Removing	0.011	[Sec]
Regionprops	0.124	[Sec]
Steering command calculation	0.005	[Sec]
Total time for each frame	**0.147**	[Sec]

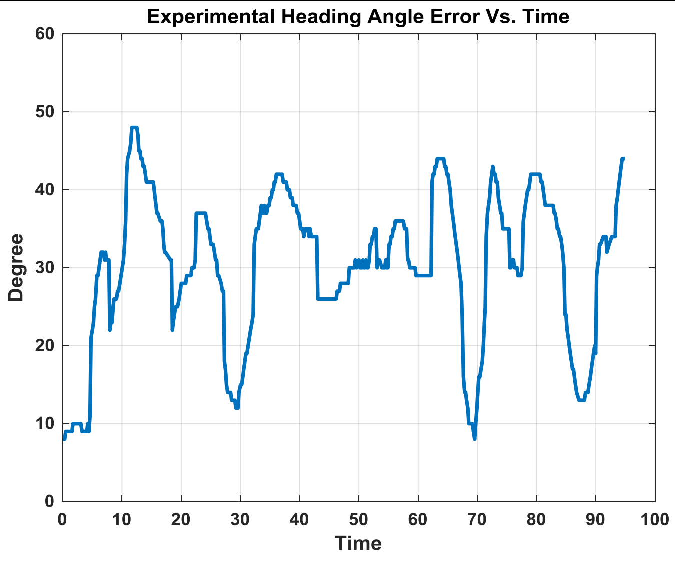


Fig. 23. Experimental Vehicle Heading Angle Error Vs. Time

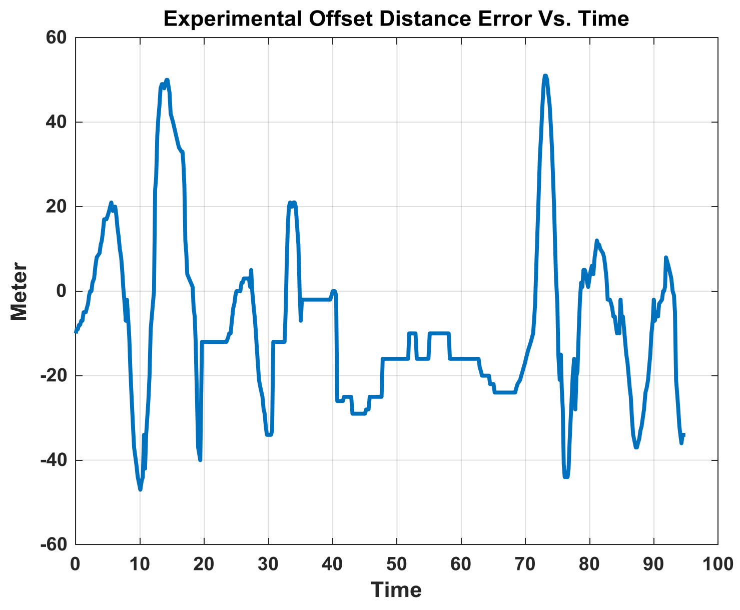


Fig. 24. Experimental Vehicle Offset Distance Error Vs. Time

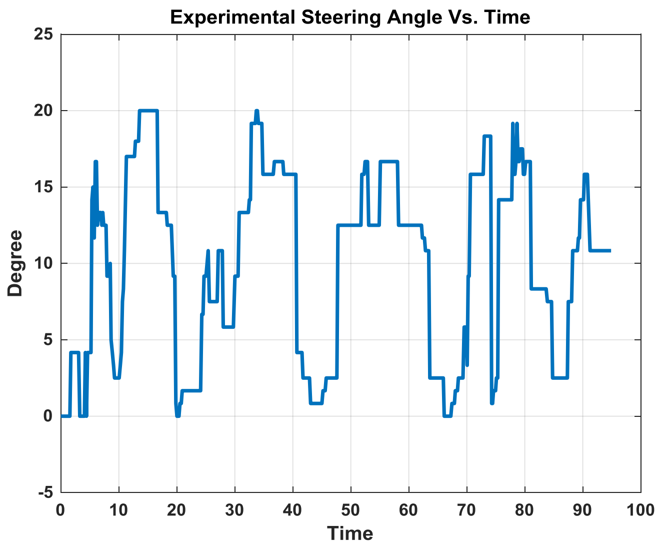


Fig. 25. Experimental Vehicle Wheel Steering Angle Vs. Time

5. Conclusion

In this paper, a method to make an autonomous vehicle is presented. The different hardware components and their assembly are clearly described. A method to determine the lane road edges is explained in details relying upon OpenCV using single digital camera. The algorithm mentioned in the paper has been successfully implemented on a small autonomous vehicle. The paper presents the design and implementation of a real-time vision-based system for detecting lane road. The vision algorithm is fast, robust, and computationally inexpensive.

6. References

[1] T.-H. Chang, C.-h. Lin, C.-s. Hsu, and Y.-j. Wu, "A vision-based vehicle behavior monitoring and warning system," in *Intelligent Transportation Systems, 2003. Proceedings. 2003 IEEE*, 2003, pp. 448-453.

[2] C. Rouff and M. Hinchey, *Experience from the DARPA urban challenge*: Springer Science & Business Media, 2011.

[3] J. Canny, "A computational approach to edge detection," *IEEE Transactions on pattern analysis and machine intelligence*, pp. 679-698, 1986.

[4] B. Yu and A. K. Jain, "Lane boundary detection using a multiresolution hough transform," in *Image Processing, 1997. Proceedings., International Conference on*, 1997, pp. 748-751.

[5] Q. Li, N. Zheng, and H. Cheng, "Springrobot: A prototype autonomous vehicle and its algorithms for lane detection," *IEEE Transactions on Intelligent Transportation Systems*, vol. 5, pp. 300-308, 2004.

[6] M. Aly, "Real time detection of lane markers in urban streets," in *Intelligent Vehicles Symposium, 2008 IEEE*, 2008, pp. 7-12.

[7] K. H. Ang, G. Chong, and Y. Li, "PID control system analysis, design, and technology," *IEEE transactions on control systems technology*, vol. 13, pp. 559-576, 2005.

[8] K. J. Åström and T. Hägglund, "PID controllers: theory, design, and tuning," *Instrument Society of America, Research Triangle Park, NC*, vol. 10, 1995.

[9] T. S. A. Al-Zaher, A. M. Bayoumy, A.-H. M. Sharaf, and Y. H. H. El-din, "Lane tracking and obstacle avoidance for Autonomous Ground Vehicles," in *Mechatronics (MECATRONICS), 2012 9th France-Japan & 7th Europe-Asia Congress on and Research and Education in Mechatronics (REM), 2012 13th Int'l Workshop on*, 2012, pp. 264-271.

[10] J.-W. Perng, Y.-H. Wen, G.-Y. Chen, and W.-J. Chang, "Intelligent Vehicle Driving Controller Design and Implementation Evaluation," *Journal of Mechanics Engineering and Automation*, vol. 2, pp. 422-430, 2012.

[11] S. Monk, Raspberry Pi cookbook: Software and hardware problems and solutions: " O'Reilly Media, Inc.", 2016.

[12] M. H. Rashid, Power electronics handbook: devices, circuits and applications: Academic press, 2010.

[13] A. Apte, "Set up and Blink - MATLAB and Simulink with Raspberry Pi," 2015.

[14] A. Mordvintsev and K. Abid, "OpenCV-Python Tutorials Documentation," ed: Itseez May, 2014.

[15] G. Van Rossum and F. L. Drake, *Python language reference manual*: Network Theory, 2003.

[16] R. W. Beard and T. W. McLain, *Small unmanned aircraft: Theory and practice*: Princeton university press, 2012.

[17] N. Arshad, K. Moon, S. Park, and J. Kim, "Lane Detection with Moving Vehicles Using Color Information," in *World Congress on Engineering and Computer Science*, 2011.

[18] X. Miao, S. Li, and H. Shen, "On-board lane detection system for intelligent vehicle based on monocular vision," *International journal on smart sensing and intelligent systems*, vol. 5, pp. 957-972, 2012.