Design and Implementation of Autonomous Car using Raspberry Pi

Mohammad Dawud Ansari

Design and Implementation of Autonomous Car using Raspberry Pi

Gurjashan Singh Pannu
Thapar University, Patiala

Mohammad Dawud Ansari
Jamia Millia Islamia, New Delhi

Pritha Gupta
Netaji Subhas Institute of Technology, New Delhi

ABSTRACT

The project aims to build a monocular vision autonomous car prototype using Raspberry Pi as a processing chip. An HD camera along with an ultrasonic sensor is used to provide necessary data from the real world to the car. The car is capable of reaching the given destination safely and intelligently thus avoiding the risk of human errors. Many existing algorithms like lane detection, obstacle detection are combined together to provide the necessary control to the car.

Keywords

Raspberry PI, lane detection, obstacle detection.

1. INTRODUCTION

Rushing around, trying to get errands done, thinking about the things to be bought from the nearest grocery store has become a part of our daily schedule. Driver error is one of the most common cause of traffic accidents, and with cell phones, in-car entertainment systems, more traffic and more complicated road systems, it isn't likely to go away.

With the number of accidents increasing day by day, it has become important to take over the human errors and help the mankind. All of this could come to an end with self-driving cars which just need to know the destination and then let the passengers continue with their work. This will avoid not only accidents but also bring a self-relief for minor day to day driving activities for small items.

2. HARDWARE DESIGN

2.1 List of Hardware

A pre-built four wheel drive (4WD) chassis is used as a base on which following hardware components are fit [9]:

- Raspberry Pi (rev B) for GPU and CPU computations
- Wi-Fi 802.11n dongle to connect to Pi remotely
- Motor driver IC L293D which can control two motors
- 8 AAA batteries to provide power
- Jumper wires to connect individual components
- L shaped aluminium strip to support camera
- Pi camera
- Ultrasonic sensor to detect obstacles
- Servo motor to make the head (camera) flexible to rotation

2.2 Hardware and Software Description

2.2.1 Raspberry Pi

The Raspberry Pi is a credit card-sized single-board computer. There are currently five Raspberry Pi models in market i.e. the Model B+, the Model A+, the Model B, the Model A, and the Compute Module (currently only available as part of the Compute Module development kit). All models use the same SoC (System on Chip - combined CPU & GPU), the BCM2835, but other hardware features differ.

The A and B use the same PCB, whilst the B+ and A+ are a new design but of very similar form factor [17].The Compute Module is an entirely different form factor and cannot be used standalone.

In this project, we have used the model B Rev 2. It comprises of a 512 MB RAM model with two USB ports and a 10/100 Ethernet controller [17].

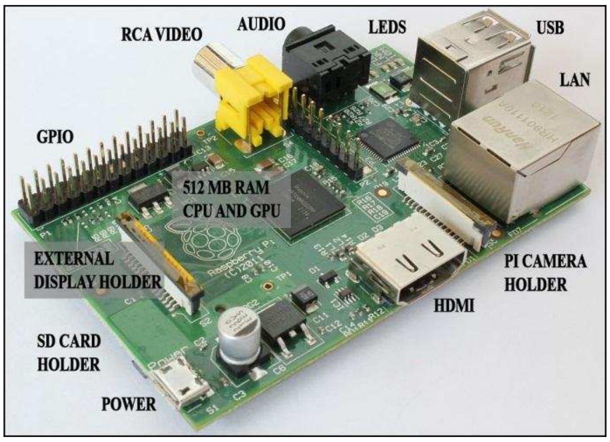

Fig 1: Features offered in Raspberry Pi Model B Rev 2

2.2.2 Pi Camera

It is the camera shipped along with Raspberry Pi [18]. Pi camera module is also available to which can be used to take high-definition videos as well as still photographs [18].

2.2.3 Ultrasonic Sensors

Ultrasonic sensors (also known as transceivers when they both send and receive, but more generally called transducers) evaluate attributes of a target by interpreting the echoes from radio or sound waves respectively [1]. In this project, they are used to detect the distance of obstacles from the car [1].

2.2.4 Raspbian OS

Of all the operating systems Arch, Risc OS, Plan 9 or Raspbian available for Raspberry Pi, Raspbian comes out on top as being the most user-friendly, best-looking, has the best range of default softwares and optimized for the Raspberry Pi hardware [19]. Raspbian is a free operating system based on

Debian (LINUX), which is available for free from the Raspberry Pi website [19].

2.2.5 *Python*

Python is a widely used general-purpose, high-level programming language [18,20, 21]. Its syntax allows the programmers to express concepts in fewer lines of code when compared with other languages like C, C++or java [20, 21].

2.2.6 *RPi.GPIO Python Library*

The RPi.GPIO Python library allows you to easily configure and read-write the input/output pins on the Pi's GPIO header within a Python script [18, 20]. This package is not shipped along with Raspbian.

2.2.7 *OpenCV*

It (Open Source Computer Vision) is a library of programming functions mainly aimed at real-time computer vision.
It has over 2500optimized algorithms, including both a set of classical algorithms and the state of the art algorithms in Computer Vision, which can be used for image processing, detection and face recognition, object identification, classification actions, traces, and other functions [21]. This library allows these features be implemented on computers with relative ease, provide a simple computer vision infrastructure to prototype quickly sophisticated applications [20, 21].

The library is used extensively by companies like Google, Yahoo, Microsoft, Intel, IBM, Sony, Honda, Toyota, and startupsarea as Applied Minds, Video Surf and Zeitera. It is also used by many research groups and government [21].

It is based on C++ but wrappers are available in python as well. In our project is used to detect the roads and guide the car on unknown roads [21].

2.3 Hardware Components Connection

The 4 wheels of the chassis are connected to 4 separate motors. The motor driver IC L293D is capable of driving 2 motors simultaneously [22]. The rotation of the wheels is synchronized on the basis of the sides i.e. the left front and left back wheels rotate in sync and right front and right back-wheels rotate in sync. Thus the pair of motors on each side is given the samedigital input from L293D at any moment. This helps the car in forward, backward movements when both side wheels rotate in same direction with same speed. The car turns when the left side wheels rotate in opposite direction to those in right [22].

The chassis has two shelves over the wheels separated by 2 inch approx. The IC is fixed on the lower shelf with the help of two 0.5 inch screws. It is permanently connected to the motor wires and necessary jumper wires are drawn from L293D to connect to Raspberry Pi [9, 22]. The rest of the space on the lower shelf is taken by 8 AA batteries which provide the power to run the motors.

To control the motor connected to pin 3 (O1), pin 6 (O2), the pins used are pin 1, pin 2 and pin 7 which are connected to the GPIOs of Raspberry pi via jumper wires[18,22].

Table I Truth Table to Control the Left Motor

Pin 1	**Pin 2**	**Pin 7**	**Function**
High	High	Low	Anti-clockwise
High	Low	High	Clockwise
High	High	High	Stop
High	Low	Low	Stop
Low	X	X	Stop

High +5V, Low 0V, X=either high or low (don't care)

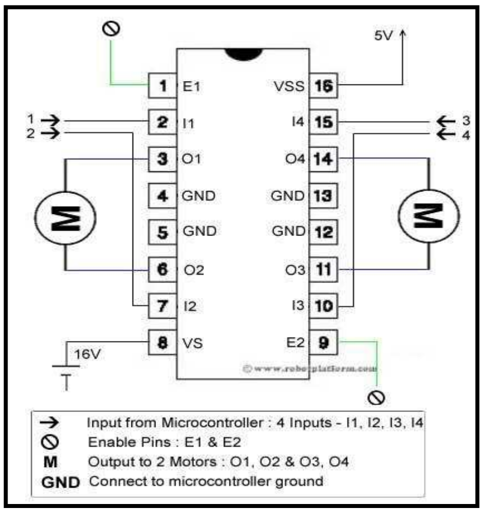
5V
1 E1
2 I1
3 O1
4 GND
5 GND
6 O2
7 I2
8 VS
16 VSS
15 I4
14 O4
13 GND
12 GND
11 O3
10 I3
9 E2
16V
M
Input from Microcontroller : 4 Inputs - I1, I2, I3, I4
Enable Pins : E1 & E2
Output to 2 Motors : O1, O2 & O3, O4
GND Connect to microcontroller ground

Fig 2: L293D IC motor driver

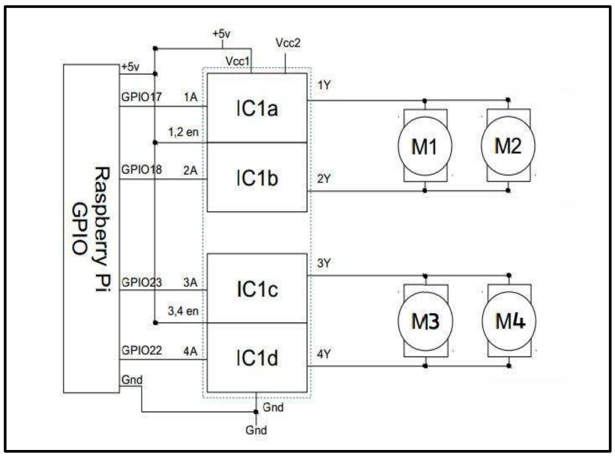

Fig 3: Hardware Connections

The raspberry pi case is glued on the top shelf along with the L shaped aluminum strip. The pi is fit in the case and the aluminum strip gives the support to the camera fit on servo motor and the ultrasonic sensor [1, 18, 20].

The Wi-Fi dongle is attached to the USB port in Raspberry Pi in order to connect to it wirelessly. The complete connection of the raspberry pi with motor controller L293D can be found in fig 2[9, 22]. Since raspberry pi needed its own IP, it needs to be connected to a Wi-Fi router or Hotspot [9]. For the same we need to make some changes in the field specified so as to make raspberry pi recognize the router every time it boots up. Navigate to the file "/etc/network/interfaces" and add following lines to make the PI connect with your router after reboot.

iface wlan0 inetdhcp

wpa-ssid "Your Network SSID"

wpa-psk "Your Password"

3. LANE DETECTION

3.1 Lane Detection Algorithm

Traditionally, lane could be detected by two approaches namely feature based technique and model based technique. The feature based technique localizes the lanes in the road images by combining the low-level features, such as painted lines or lane edges etc [2, 4]. Accordingly, this technique requires well studied road having well-painted lines or strong lane edges, otherwise it will fail. Moreover, it has the disadvantage of not imposing any global constraints on the lane edge shapes, this technique may suffer from occlusion or noise [2, 4].

On the other hand, the model-based technique just uses a few parameters to represent the lanes [2, 4]. Assuming the shapes of lane can be presented by either straight line or parabolic curve, the processing of detecting lanes is approached as the processing of calculating those model parameters [2, 4]. This way, the model-based technique is much more robust against noise and missing data, compared with the feature-based technique. To estimate the parameters of lane model, the likelihood function, Hough transform, and the chi-square fitting, etc. are applied into the lane detection. However, as the most lane models are only focused on certain shapes of road, thus they lack the flexibility to modeling the arbitrary shape of road.

In the proposed algorithm to detect the lanes, a combination of feature and model base is used. In general, this algorithm is valid for all kind of roads (whether they are marked with white lanes or not).

The overall method consists of 7 major parts:

3.1.1 Extract the color range for the road

Extract the appropriate upper and lower range to determine the color of the portion on which the car is standing [13]. This is the primary and most important part of this algorithm. Using this range, a binary image of the current view is created.

3.1.2 Define the region of interest

A region of interest is defined starting from the bottom towards upward. As the view is taken from the camera on the car, the road surface closest to the car is at the bottom of the image. Thus the region of interest is defined from the nearest region to the farther distances by moving upwards in the image created. The height of the region of interest is usually not more than half the height of image [23]. This removes the sky area from the visual field and saves unnecessary computations and is better than the method proposed by H.Dahlkamp [3]. As we move away from the car, the width of the road seems to be narrowing. So, the actual region of interest is in a shape of a trapezium [3].

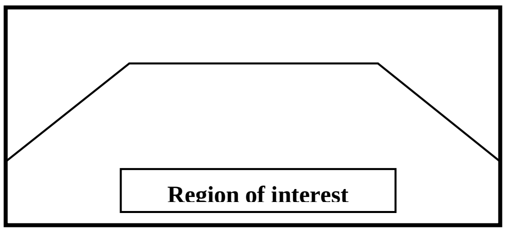

Fig 4: Define the region of interest

3.1.3 Convert complex region of interest into simple shape

This is an important step in determining the boundary of the road. In the region of interest, the contours are determined. Since the car is on the road, the largest contour which contains the central part of the bottom region is the road. Simplify the shape of this contour using approximations.

3.1.4 Determine the shape of the road

We draw Hough lines on the manipulated contour. A number of Hough lines are obtained along the left and right edges of the road. Out of these, only few lines represent the actual edge. Rests of the lines are due to the noise (due to irregularities in the road) along the edge [6].

3.1.5 Filtering the noise

The lines along the left edge of the road are tilted towards right and vice versa. So, any line which is tilted towards left and lies entirely in the left half the image is discarded. Similar lines are also discarded from the right half[6].For the left edge, a line with the smallest angle or having the least positive x intercept or y intercept is chosen as the left edge of the road. Similarly, find the right edge of the road. A general case could be represented as shown in Fig 5.

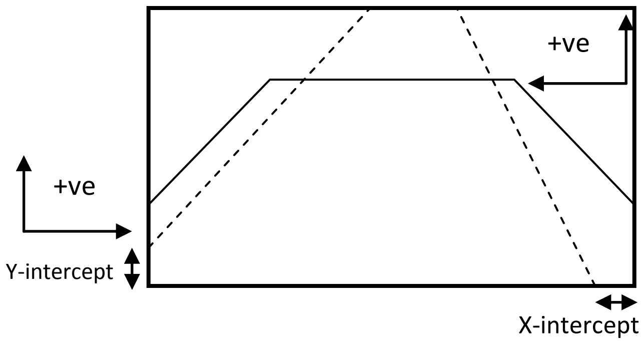

Fig 5: Convention for positive intercepts

3.1.6 Make it robust and unaffected from noise

For the moving car, the change in the direction of road cannot be abrupt. The edges of the road cannot change the angle in discrete fashion. So, any line which is far away from the line in previous frame is simply discarded.

Another factor called "tolerance" is considered. Basically, it is the count of the continuous frames which could be accepted without being able to determine the edge in entire above mentioned process. Its maximum value is 3. If we are not able to determine the edge of the road, tolerance value is decremented by 1. If it reaches 0, we retry to find the new colour range of the road at the run time [6, 7].

3.1.7 Determine the turns in the road, if any, and give directions to the car

The different cases of the road (left turn, right turn, divergence, round - about etc.) are considered. Suppose there is a left turn as shown in Fig 6. Divide the region of interest in 3 parts in ratio 2:3:5. Compare the lines obtained in all three parts with the possible shape of the road. The dotted lines in each section in figure 3.1 show the lines that are obtained applying Hough lines in these sections separately.

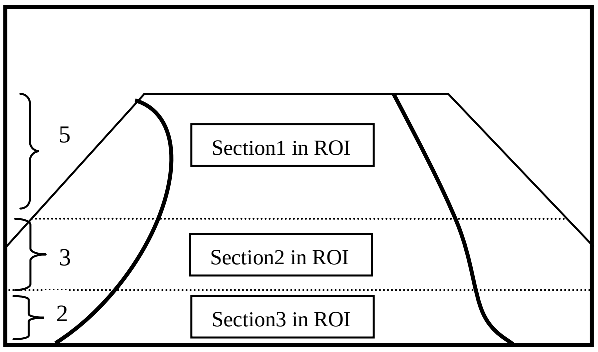

Fig 6: Division of region of interest in 2:3:5

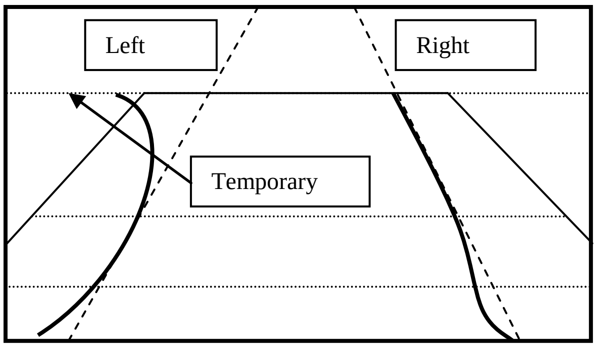

Fig 7: Determine the potential left and right edges

Lines making same angle with the x axis and lying in either all three sections or two sections namely top and middle or even only one section is its middle or bottom section are chosen and predicated to be the edges of the road shown in figure 7.
Depending upon the continuation or break of same line in 3 different sections, the final ROI is modified which further assists to decide the turn present on the road.
The above fig 6 shows a road that has a left turn ahead. Applying the steps mentioned above,the three parts of the region of interest are divided as shown. Finding the left and right edges in these different sections can be used to find the possible edges of the road as is depicted by two dotted lines connecting the bottom of the ROI to the top (Fig 7).

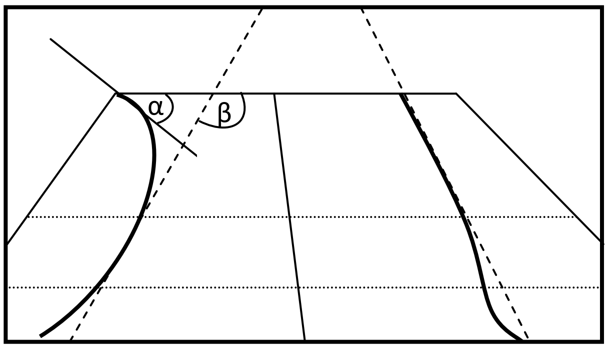

Fig 8: Modified ROI grasps top of left edge within itself

Next challenge lies in determining the turns and changing the line of motion of the car. A temporary line, as shown in fig 7, overlapping the top edge of the ROI from left to right is drawn. A tangent to the contour is also drawn at the intersection point of the temporary line with contour. If the intersection point lies outside the ROI, the ROI is modified to accommodate this point within itself. This could be seen in Fig 8, where the top point of the left edge of the road is within the ROI which was lying outside the ROI in Fig 7.In Fig 8 the tangent is making an angle α with the top of ROI. And β is the angle between the determined left edge and top of ROI. For the left lane these angles are different. Similarly these angles are calculated for right lane and are found same. These parameters are used to come to a conclusion that there is a turn ahead and that turn is a left turn. After this step there is a change in line of motion of the car depending upon the new line obtained by joining the mid points of the top and bottom line of the modified ROI.

4. PROJECT PHASES

4.1 Phase I: Remotely Controlled Car

Controlling the movement of the car can be done using some kind of remote control interface. It may be a web interface or a mobile interface. Since setting up python and SSH in raspberry pi is very easy without any cons, it is recommended to use a mobile based user interface for controlling the car motion. One of the most common approach people use to send command to control the GPIO ports in raspberry pi is WebIOPI. Using WebIOPI one can change any GPIO port of the raspberry to input or output. Also you can easily send a high and low signal on any port specified just by send a GET and POST request. The problem you encounter in this approach is "how to keep the latency to its least", since every process, may be setting a port to a output pin or making it high or low is considered as an individual operation we encounter a lot of delay while making a moving vehicle stop using web interface. To overcome this people generally use REST API provided along with the WebIOPI implementation for designing their own function for moving forward, to stop and to take turn. First of all it is a tough thing to implement if you have very less knowledge of REST APIs. Though it may be to some extent useful for one who can ignore the delay but it is not considered as the best approach for controlling the motion of the car. To solve these problems an android app is made which can communicate using an open channel whose TTL is set high so as to keep the channel connected as long as possible.

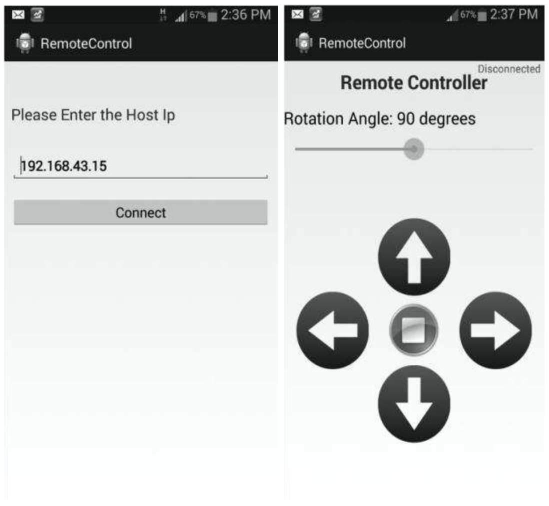

Fig 9: Screenshots of android app to connect to car remotely

4.1.1 Mobile Application

Android is chosen for designing the UI as most of the people today have an android phone. Also the development, installation and debugging of a sample app is very easy in android. The backend of the application use a well-known JSch library written in Java to connect with the SSH channel of the SSH server running in raspberry pi. Since both the mobile and raspberry pi is connected to a common network wirelessly. The first page of the mobile app is meant to choose the IP assigned to Raspberry PI so as to make a successful connection withit. The second page shows the connection status of the channel connected. The left, right, top and bottom buttons are used to control the left, right, move forward and move back. The screenshots are shown in Fig 9.

4.2 Phase II: Car with Autonomous Obstacle Avoidance

In robotics, obstacle avoidance is the task of satisfying the control objective subject to non-intersection or non-collision position constraints [1]. Normally obstacle avoidance involves the pre-computation of an obstacle-free path along which the controller will then guide a robot.

Though inverse perspective mapping helps to find the distance of the objects far away from the car with the help of known camera parameters and generating a model but it takes more computations. Using ultrasonic sensor is better option in this case as it doesn't require high CPU computations and detects the obstacles as well as help us finding the distance[15][16].

Ultrasonic sensors used to detect the distance of nearby flat objects so as to avoid obstacles in general. This is a very low power device and has a very extensive use in mini autonomous robots and cars.

The working can be explained as transmitting a low frequency sound from the sensor to the object which after reflection is received by the receiver of the sensor. Depending on the time taken to receive the reflected signal, the distance

of the nearby vehicle or any other obstacles detected. One demerit of this approach is if the reflecting surface is at certain angle with the sensor the distance measure may be ambiguous [1] and had to be supported with other techniques like OpenCV and image processing before making any decision about the turn.

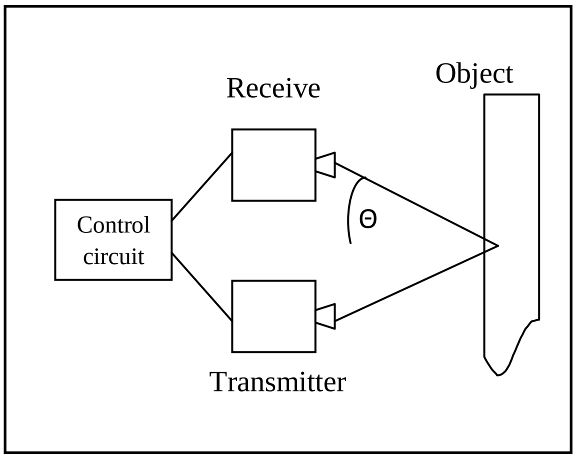

Fig 10: Concept of ultrasonic sensor

The ultrasonic sensor is mounted on a servo motor at the front of the chassis. The sensor rotates periodically and checks for the potentially threatening obstacles which may or may not be in the line of motion but may hit the car if precaution is not taken.

4.2.1Algorithm

Watch the surrounding after a fixed interval of time i.e. 300ms. The following steps are repeated every interval.

1. Watch the surroundings to calculate the distance of the obstacles from the car.
2. The minimum threshold distance that is safe for the car is 1 metre. If the distance calculated comes out to be lesser than threshold, stop the car and check other sides.
3. Rotate the car and move ahead.

4.3 Phase III: Car with Autonomous Navigation

This phase is about making the car autonomous i.e. the car determines the road by itself and finds it line of motion. Following techniques is performed step-by-step to achieve autonomous behaviour in the car.

4.3.1 Algorithm

Extracting the color range for the roads in HSV color-space requires manual data collection in the beginning. Same portion of the road is recorded at different times of the day and in different weather conditions [13]. Considering the changes in variation of the color and the intensity of light, a specific set of upper and lower bounds for the HSV values is generated [13].

1. Define the data structures required for the algorithm. i.e. a 2D array of Scalar which contains the set of possible upper and lower threshold color values, and tolerance for both left and right side with default value as 3.
2. Select the threshold values as per the conditions and convert the image frame into a binary image.
3. Reduce the region of interest from full frame to a trapezium with its base touching the bottom. The region of interest is divided into 3 parts of height 50%, 30% and 20% starting from the top and moving towards bottom. Steps 4 to 11 are applied for all three parts of the region of interest.
4. Find the contours in the region of interest.
5. Determine the largest contour containing the horizontally central part of bottom of the chosen section in the region of interest.
6. Approximate the contour into a simpler polygonal curve. This curve (basically a contour) represents the potential road.
7. Find the Hough lines on the probable road curve.
8. The lines with negative slope (lines going from right to left as we move upward) present in the left half and the ones with positive slope (lines going from left to right as we move upward) present in right half are ignored.
9. The lines with the least positive x-intercept or y-intercept on both left and right halves are selected as left and right edge lines respectively.
10. If no such left or right edge line is obtained then the corresponding tolerance is decremented by 1.
11. If both left and right tolerance has become less than 1, we need to change the threshold values for road color in HSV color-space.
12. If the three lines representing the edges in three parts of region of interest are along the same straight line,

(check this using the angles they are making) the road is concluded straight. If we get the line in the middle and the bottom part, then the road is turning. On encountering a turn, region of interest is modified accordingly.

13. A line dividing the region between the determined left and right edges in two equal halves vertically gives the line of motion for the car.
14. While following the movement line defined, use the ultrasonic sensor to detect the obstacles.

Using the following algorithm and obstacle avoidance from phase II is combined together to implement a final car which is fully autonomous.

5. RESULT

Most of the experiments done are focussed on the task of detecting the variety of roads in different conditions. In order to do this, different tests were run on different images of the roads.

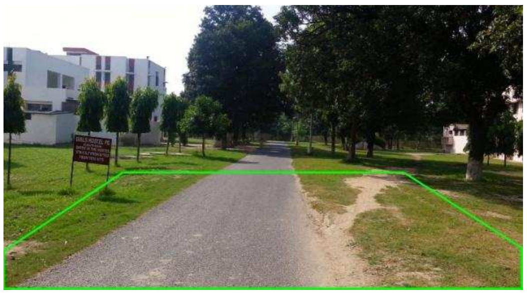

Fig11: Original road with region of interest (ROI)

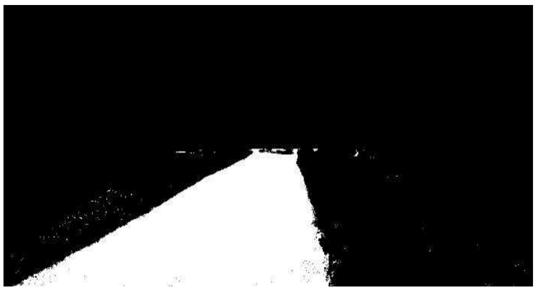

Fig12: The detected road surface

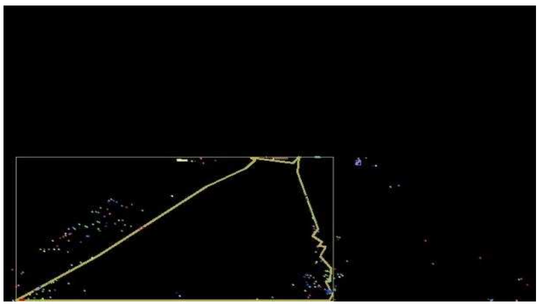

Fig13: The contour around the road

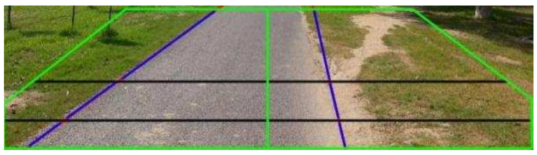

Fig 14: Detected road edges in different parts of ROI

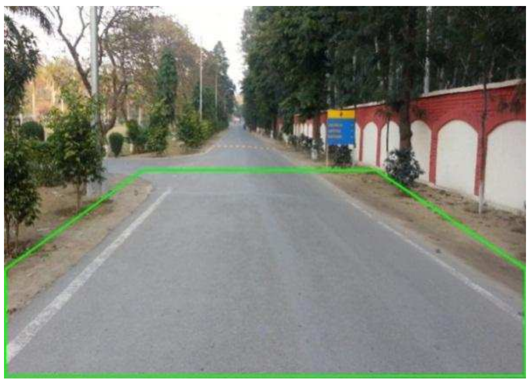

Fig 15: Original road with region of interest (ROI)

Fig 11 and Fig 15 shows the originals images of different road with the modified assumed ROI. Fig 12 and Fig 16 shows the binary images that are obtained using pre calculated threshold HSV. Fig 13 and Fig 17 shows the contours obtained from the road after approximating it to nearest polygon shape.

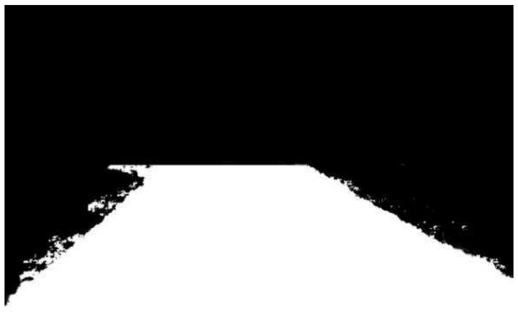

Fig 16: Detected road surface

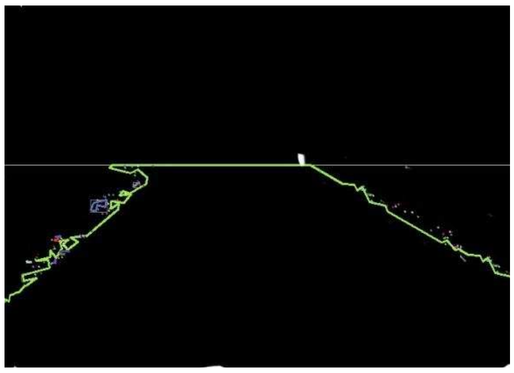

Fig 17: The contour around the road

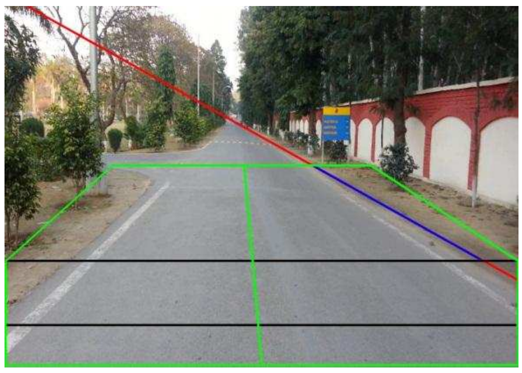

Fig 18: Detected road edges in different parts of ROI

Fig 14 and Fig 18 shows the final images obtained which help in calculating the new line of motion of the road. In Fig 14 both left and right lane is detected exactly so the blue line is found from section 1 extending to section 2 and section3. In Fig 18 only right lane in the section 1 is detected so it is blue colored. Also since the left lane is turning towards left so angles α and β(check Fig 8) are calculated and thus after necessary calculation the results obtained says that line of motion is turning left and thus the ROI is modified accordingly.Since a lot of data has been collected to determine the thresholds for the road in different conditions, the shadows on the road are no longer a problem. No special case is to be considered to remove the shadows [14].

6. CONCLUSION AND FUTURE WORK

6.1 Conclusion

In this paper, a method to make a self driving robot car is presented. The different hardware components and their assembly are clearly described. A novel method to determine the uneven, marked or unmarked road edges is explained in details relying upon OpenCV. Using ultrasonic sensors, the collisions with obstacles is avoided. The algorithm mentioned in the paper has been successfully implemented on a small autonomous car.

6.2 Future Work

The work could be enhanced by improving the algorithm by adding machine learning to it. The present algorithm performs the operations on all the frames. It is accurate but its efficiency could be further enhanced if it starts learning by itself and avoid unnecessary calculations of the regions which are already known or familiar. Once the car starts travelling on the roads, it determines the obstacles (mainly static) on the way and note their characteristic features. An XML file is generated every time the car travels. It stores the following information:

- The distance between the two nodes.
- The number of roads diverging from a particular node.
- The number of speed breakers and other static obstacles on the road joining two nodes.
- The distance of speed breakers and other static obstacles from a specific node.
- Height and the width of an obstacle.

The information stored in the XML helps the car understand and remember the path being followed for the next time.

7. REFERENCES

[1] Johann Borenstein & Yoram Koren, Obstacle Avoidance with Ultrasonic Sensors, IEEE JOURNAL OF ROBOTICS AND AUTOMATION, VOL. 4, NO. 2, APRIL I988, pp. 213-218

[2] Yue Wanga, Eam Khwang Teoha & Dinggang Shenb, Lane detection and tracking using B-Snake, Image and Vision Computing 22 (2004) , available at:www.elseviercomputerscience.com, pp. 269–280.

[3] H. Dahlkamp, A. Kaehler, D. Stavens, S. Thrun, and G. Bradski. Self-supervised monocular road detection in desert terrain. G. Sukhatme, S. Schaal, W. Burgard, and D. Fox, editors& *Proceedings of the Robotics Science and Systems Conference*, Philadelphia, PA, 2006.

[4] Joel C. McCall &Mohan M. Trivedi, Video-Based Lane Estimation and Tracking for Driver Assistance: Survey, System, and Evaluation, IEEE Transactions on Intelligent Transportation Systems, vol. 7, no. 1, March 2006, pp. 20-37.

[5] Tushar Wankhade & Pranav Shriwas, Design of Lane Detecting and Following Autonomous Robot, IOSR Journal of Computer Engineering (IOSRJCE) ISSN: 2278-0661 Volume 2, Issue 2 (July-Aug. 2012), pp. 45-48.

[6] Xiaodong Miao, Shunming Li & Huan Shen, On-Board lane detection system for intelligent vehicle based on monocular vision, International Journal on Smart Sensing and Intelligent Systems, vol. 5, no. 4, December 2012, pp. 957-972.

[7] A. Bar Hillel, R. Lerner, D. Levi, & G. Raz. Recent progress in road and lane detection: a survey. Machine Vision and Applications, Feb. 2012, pp. 727–745

[8] Narathip Thongpan, & Mahasak Ketcham, The State of the Art in Development a Lane Detection for Embedded Systems Design, Conference on Advanced Computational Technologies & Creative Media (ICACTCM'2014) Aug. 14-15, 2014

[9] Narayan Pandharinath Pawar & Minakshee M. Patil, Driver Assistance System based on Raspberry Pi, International Journal of Computer Applications (0975 – 8887) Volume 95– No.16, June 2014, pp. 36-39.

[10] J.M.A. Alvarez, A.M. Lopez & R. Baldrich, *Illuminant-Invariant Model-Based Road Segmentation.* Intelligent Transportation Systems, IEEE Transactions on, 12, 2008, pp 184–193.

[11] W. Maddern, A. Stewart, C. McManus, B. Upcroft, W. Churchill, & P. Newman, *Illumination Invariant Imaging: Applications in Robust Vision-based Localization, Mapping and Classification for Autonomous Vehicles*, Proceedings of the Visual Place Recognition in Changing Environments Workshop, IEEE International Conference on Robotics and Automation (ICRA), Hong Kong, China, 2014.

[12] Neha A Ghaisas & R. R. Sedamkar, *Inter-vehicular Communication using Packet Network Theory,* IJRET: International Journal of Research in Engineering and Technology, ISSN: 2319-1163 | ISSN: 2321-7308, Volume: 03 Issue: 09 | Sep-2014, available at: http://www.ijret.org, pp. 72-78

[13] Trifan, A., Neves, A.J.R. & Cunha, B, *Evaluation of color spaces for user-supervised color classification in robotic vision.*17th International Conference on Image Processing, Computer Vision, & Pattern Recognition, Las Vegas, Nevada, USA (July 2013).

[14] R. Cucchiara, C. Grana, M.Piccardi& A. Prati, *Detecting moving objects, ghosts, and shadows in video streams,* IEEETransactions on Pattern Analysis and Machine Intelligence(PAMI), Vol. 25(10), 1337 - 1342, 2003.pp.25-28

[15] S. Tuohy, D. O'Cualain, E. Jones, & M. Glavin, *Distance determination for an automobile environment using inverse perspective mapping in OpenCV,* in Proc. Irish Signals and Systems Conference 2010.

[16] Li, M., Zhao, C., Hou, Y. & Ren, M. , *A New Lane Line Segmentation and Detection Method based on Inverse Perspective Mapping, International Journal of Digital Content Technology and its Applications. Volume 5, Number 4, April 2011, pp. 230-236*

[17] Dhaval Chheda, Divyesh Darde & Shraddha Chitalia, *Smart Projectors using Remote Controlled Raspberry Pi*, International Journal of Computer Applications (0975 – 8887),Volume 82 – No. 16,2013, pp.6-11

[18] Stewart Watkiss, *Design and build a Raspberry Pi robot* [Online], available at: http://www.penguintutor.com/electronics/robot/rubyrobot-detailedguide.pdf

[19] David Hayward, *Raspberry Pi operating systems: 5 reviewed and rated* [Online], available at: http://www.in.techradar.com/news/software/

[20] Matt Richardson, Shawn Wallace, *Getting Started with Raspberry Pi*, 2nd Edition, Published by Maker Media, Inc., USA, 2014. Book ISBN: 978-1-457-18612-7

[21] Gary Bradski, Adrian Kaehler, *Learning OpenCV: Computer Vision with the OpenCV Library*, "O'Reilly Media, Inc.". Copyright.September 2008, 1st Edition, Book ISBN: 978-0-596-51613-0

[22] Ch. Krishnaveni , Ms. A. Siresha , Mr. B. J. Prem Prasana Kumar & Mr. K. Sivanagireddy, *Implementation of embedded systems for pedestrian safety using haar features*, IJEC: International Journal of Electrical Electronics and Communication,ISN 2048 – 1068,Volume: 06 Issue: 20 l Oct -2014, pp. 761-766

[23] Tan-Hung Duong , Sun-Tae Chung , Seongwon Cho, Model-Based Robust Lane Detection for Driver Assistance, available at: http://www.kpubs.org/article/articleMain.kpubs?spotType=low&articleANo=MTMDCW_2014_v17n6_655