AUTONOMOUS PATH TRACKING AND COLLISION AVOIDANCE SYSTEM

CONTENTS :

TITLE:

AUTONOMOUS PATH TRACKING & COLLISION AVOIDANCE CONTROLLED SYSTEM

OBJECTIVE :

To develop a vehicle that can maneuver through a path, using computer vision, all by itself taking necessary decisions to steer and keep itself on the track.

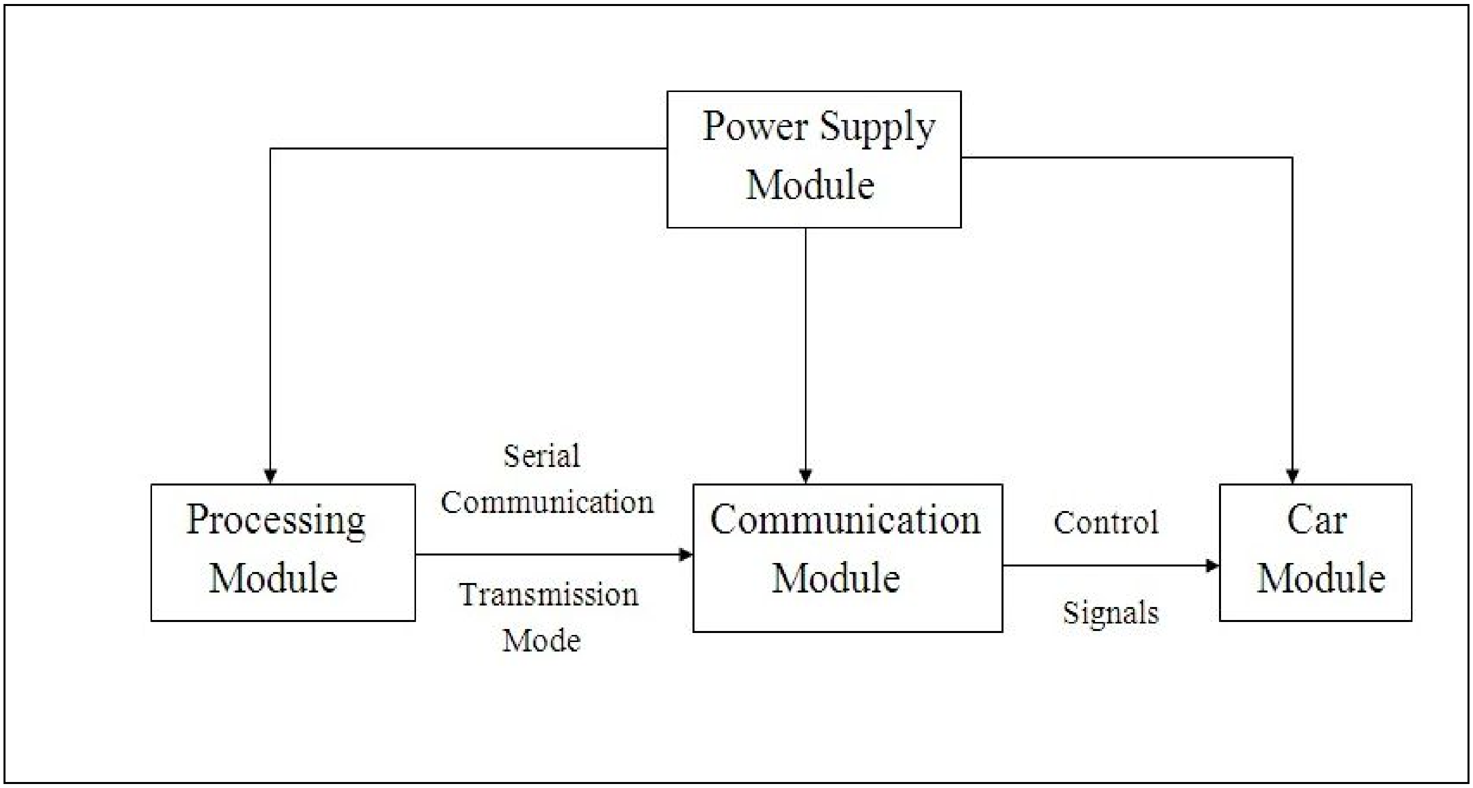
PROJECT OVERVIEW :

An autonomous controlled system also known as robotic or informally as driverless or self-driving, is an autonomous vehicle capable of fulfilling the human transportation capabilities of a traditional car. As an autonomous vehicle, it is capable of sensing its environment and navigating on its own. A human may choose a destination, but is not required to perform any mechanical operation of the vehicle. Autonomous vehicles sense the world with the help of computer vision. The image interprets the information to identify appropriate navigation paths, as well as obstacles. Autonomous vehicles typically update their memory maps based on information, such that they can navigate through uncharted environments.

As a result of making a self-driving car, it enables a lot of benefits.

- Fewer traffic collisions due to the autonomous system's increased reliability and decreased reaction time compared to human drivers.
- Increased roadway capacity and reduce traffic collisions due to reduced need of safety gaps[1] and the ability to better manage traffic flow.
- Removal of constraints on occupants' state such as physical disabilities or being drunk.
- Reduction of space required for vehicle parking.
- Reduction in the need for traffic police and vehicle insurance.

BLOCK DIAGRAM :



- ***Processing Module :*** Processes the image obtained from camera to detect the path and take decisions on which path the car have to move or change its direction.

- ***Communication Module :*** Communicates with the processing module in order to get information required for the path tracking and give proper electrical signals to the Driver IC for maneuvering the car along its path.

- ***Car Module :*** It's main purpose to steer the vehicle left-right, forward-backward by activating respective DC motors on the basis of the electrical signals obtained from communication module.

- ***Power Supply Module :*** It supplies the power required for all modules to function properly. From DC motor of 18v to micro-controller 5v.

Chapter 1 - Processing Module

1.2 – APPROACH I :

ABSTRACT :

This algorithm simply tracks the road and guides the car on its path. It can track straight road and also roads with small curvature. Algorithm tracks the road based on **Edge-Detection**. There is a camera on board which looks at the road and gives an image to the laptop as Input which processes the image based on following algorithm and gives control output to the microcontroller **Serially** which in turn drives the motor. The algorithm captures an image frame through the camera and processes it to track the road. We basically find the edge of the image and since the road ending contributes to a valid edge point this will help to guide the car moreover if there are any obstacles also then that will also contribute to edge point. We take a frame and find the direction in which there is max space available. We form a 1D array having the plot of the road contour and we convolve this with a pre-defined function to find the point of maximum match so as to find the direction where maximum free space is available. We then drive the car accordingly if the road is on its left then it drives the car towards left and vice- versa.

ALGORITHM & CODE :

1. We create a “ **video object** ” called vid to read external camera and set its resolution and colour –format as 240x320.

```
vid=videoinput('winvideo',1,'YUY2_320X240');
getselectedsource(vid);
set(vid,'ReturnedCOlorSpace','RGB');
figure,preview(vid);
```

FIG 1:- INPUT IMAGE

FIG 2:- Lenovo Bird View Camera

2. Read a frame from the camera through the created video object and then convert the RGB image into Gray Scale image

```
im1=getsnapshot(vid);

for iteration=1:200
    im=rgb2gray(im1);
```

FIG 3:- Gray Scale Image

3. Finds the edges on the Gray-scale image and stores it in the variable BW as a 2-D array and the image would look like in the following manner after edge-detection.

```
BW=edge(im,.0343);
imshow(BW);
```

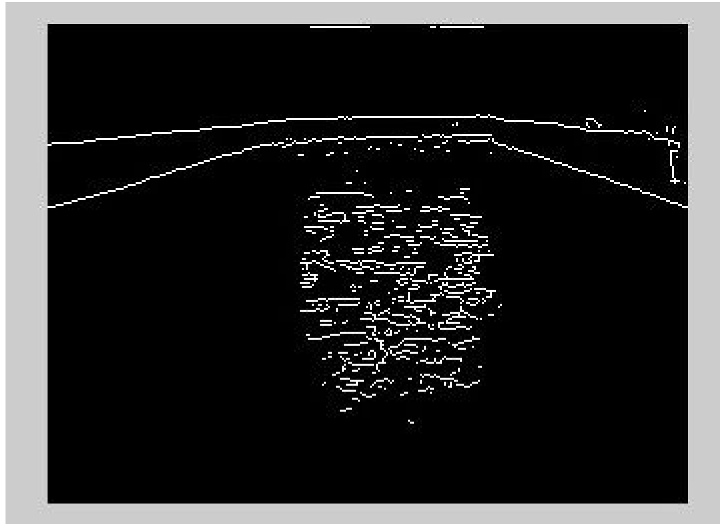


FIG 4:- Image just after Edge-Detection

4. Brightness -Correction function gives those unwanted edge-positions where there is high light illumination and then removes those edge points in image array.

```
[BR] = BRIGHTNESS_CORRECTION( im1 );
BW=BW&(~BR);
[L NUM]=bwlabel(BW);
```

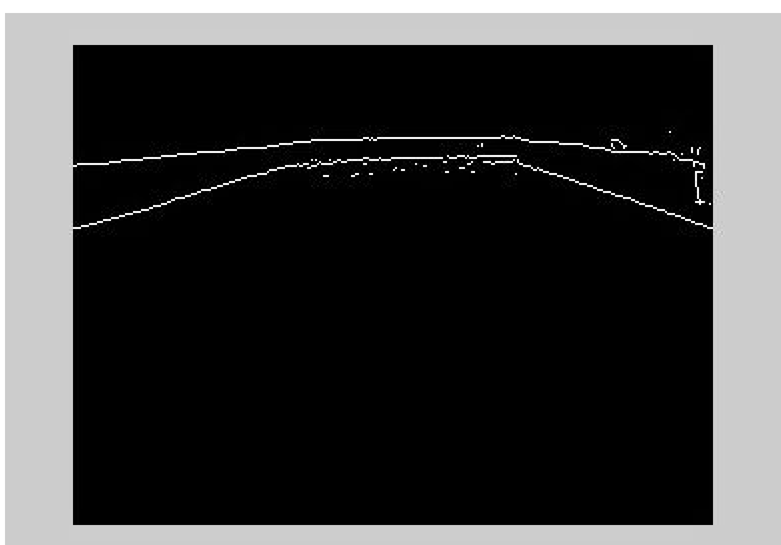


FIG 5: Image just after Brightness -Correction

5. Now the area of each connected components are calculated and then we remove those connected components whose area is below certain threshold level to get the complete edge picture of the road ahead. Now we form a new edge matrix after removing all unwanted noise edge points.

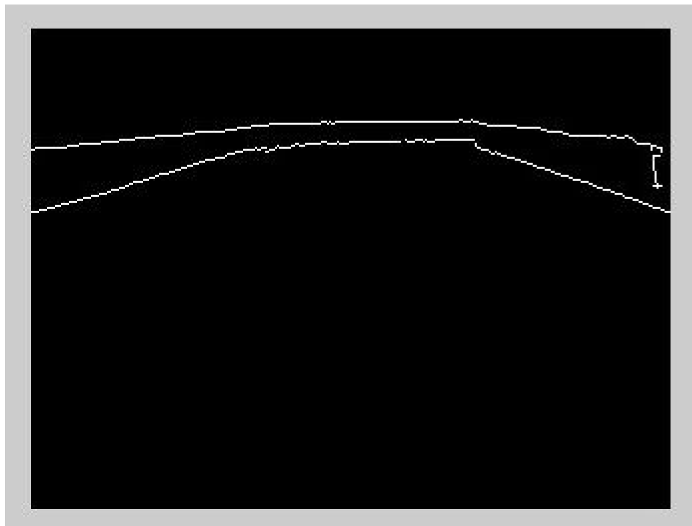


FIG 6: Image after the Area-Thresholding

6. We find the contour of the road or the shape of the road and store it in X1 variable.

```
I1=boolean(zeros(row,column));
for i=1:column
    for j=1:row
        if(I(row-j+1,i)==1)
            X1(i)=j;
            break;
        end
    end
end
```

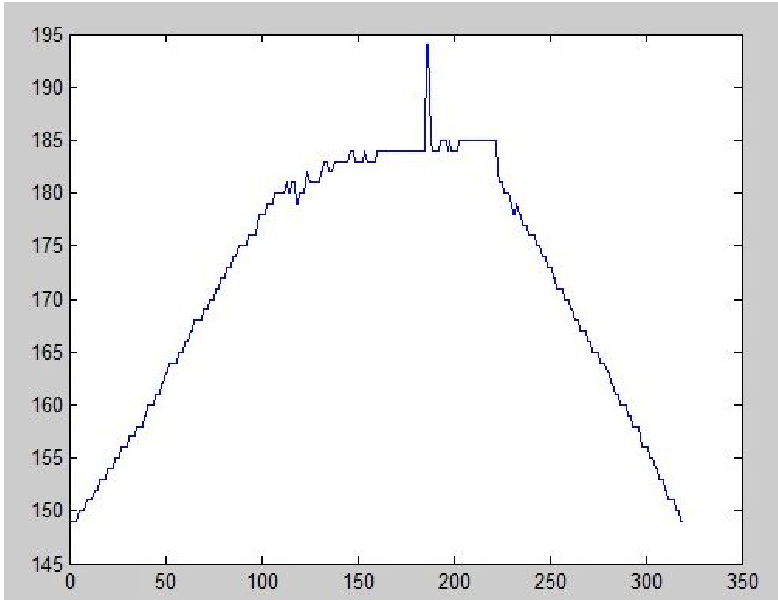
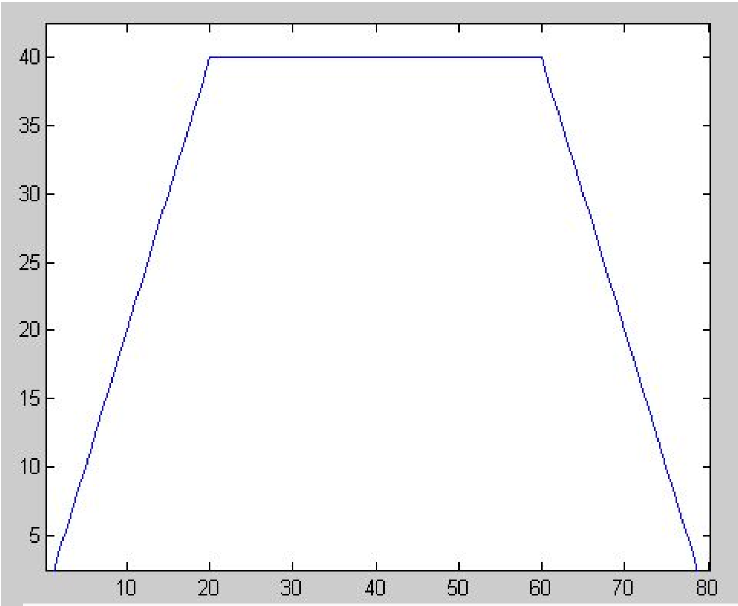
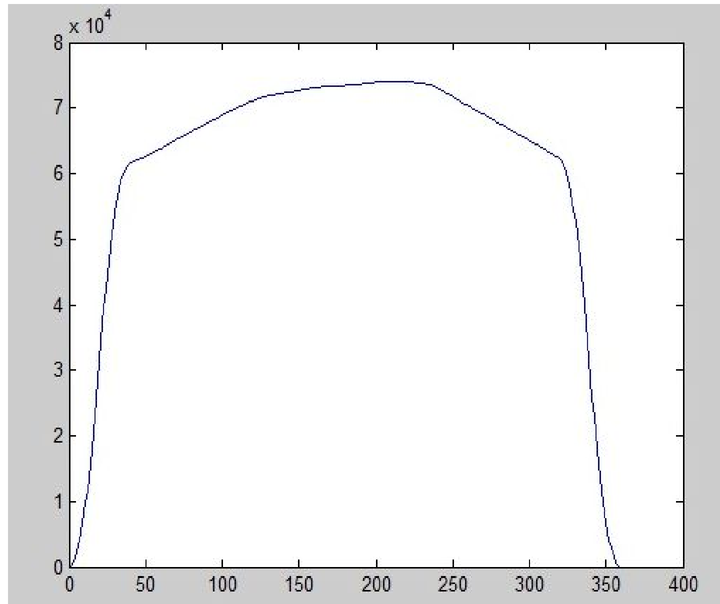


FIG 7: Contour of shape of the road (X1)

7. Convolute the X1 variable with a pre-defined shape stored in X2 variable.

```
X2=[1:column/16 (column/16)-(1:column/16)]
mid=uint8(column/16);
Y=conv(X1,X2);
[max1 index]=max(Y);
dir=index-mid+1;
```





8. A serial port object has been created to access COM PORT 10 with specified set of parameters.

```
s=serial('COM10');
set(s,'BaudRate',9600);
fopen(s);
```

9. We plot the lines in the image being displayed and determine how to steer the vehicle through the path by giving it proper instructions (Left, Right, Straight & Stop)

```
imshow(im1);hold on
xy=[axis-30,1;axis-30,row];
plot(xy(:,1),xy(:,2),'LineWidth',2,'Color','yellow');
xy=[axis+30,1;axis+30,row];
plot(xy(:,1),xy(:,2),'LineWidth',2,'Color','yellow');
xy=[dir,100;dir,row-50];
plot(xy(:,1),xy(:,2),'LineWidth',4,'Color','black');
plot(row-X1);hold off
if(axis-30 > dir)
    fprintf(s,'L'); %left
else
    if(axis+30 < dir)
        fprintf(s,'R');% right
    else
        fprintf(s,'S');% straight
    end
end
```

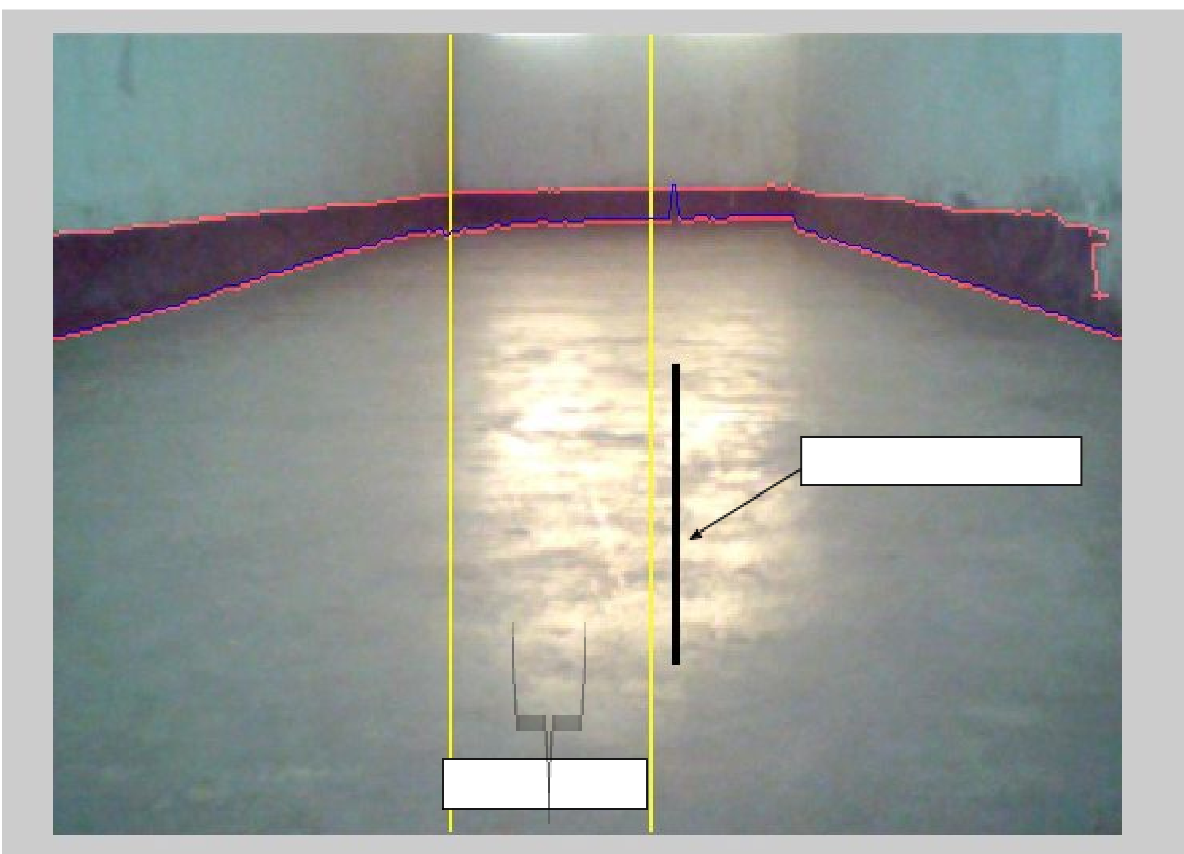


FIG 10 :- FINAL IMAGE

- Both the serial port accessing object and the video object both are terminated after executing the whole program.

```
fprintf(s,'T');% STOP
fclose(s);
delete(s);
delete(vid);
clear vid s;
```

Advantages & Disadvantages –

1. Fast dynamic illumination invariant robust, we are finally using this model only.
2. No as such disadvantage.

1.3 - Approach II :-

This algorithm uses K-means clustering algorithm to segment the road area from the input image and identify the road and give driving instructions.

K-MEANS ALGOITHM –

The k-means algorithm uses an iterative refinement technique. Given an initial set of k means m1,…,mk , the algorithm proceeds by alternating between two steps:

Assignment step: Assign each point to the cluster whose mean is closest to it.

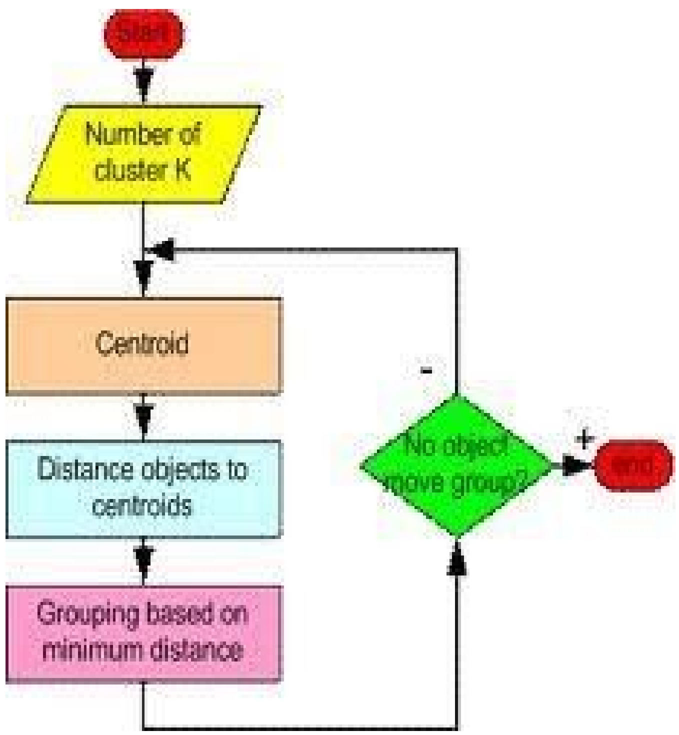
$$S_i^{(t)} = \{x_p : \|x_p - m_i^{(t)}\| \leq \|x_p - m_j^{(t)}\| \ \forall\ 1 \leq j \leq k\},$$

where each x_p is assigned to exactly one $S^{(t)}$, even if it could be is assigned to two or more of them.

Update step: Calculate the new means to be the centroids of the observations in the new clusters.

$$\mathbf{m}_i^{(t+1)} = \frac{1}{|S_i^{(t)}|} \sum_{\mathbf{x}_j \in S_i^{(t)}} \mathbf{x}_j$$

The continue the above steps till it converges to a single value.



1. Click in the first snapshot.
2. Segment the image using k-means algorithm into 4 clusters out of which one cluster will be the road. We align the camera such that the road takes the maximum area. So out of the 4 clusters the cluster having the maximum area is the road. This is time consuming step.

3. Now k-means algorithm has high time complexity. It takes a lot of time to find out the road, so we found out another shortcut solution

to this problem. After computing the clusters for the first time from the first frame we use the clusters to segment the image from next frame and also update the clusters from the 2nd frame i.e. we just run the k-means clustering algorithm iteration for only one time. This saves a lot of time. So our algorithm takes a lot of time to find the road for the first time but after that it becomes fast.

4. Then after getting the road area we find out the road contour and then find out the direction in which the robot must go to stay on road. We give this direction to communication module which drives the car accordingly.
5. Go to step 3.

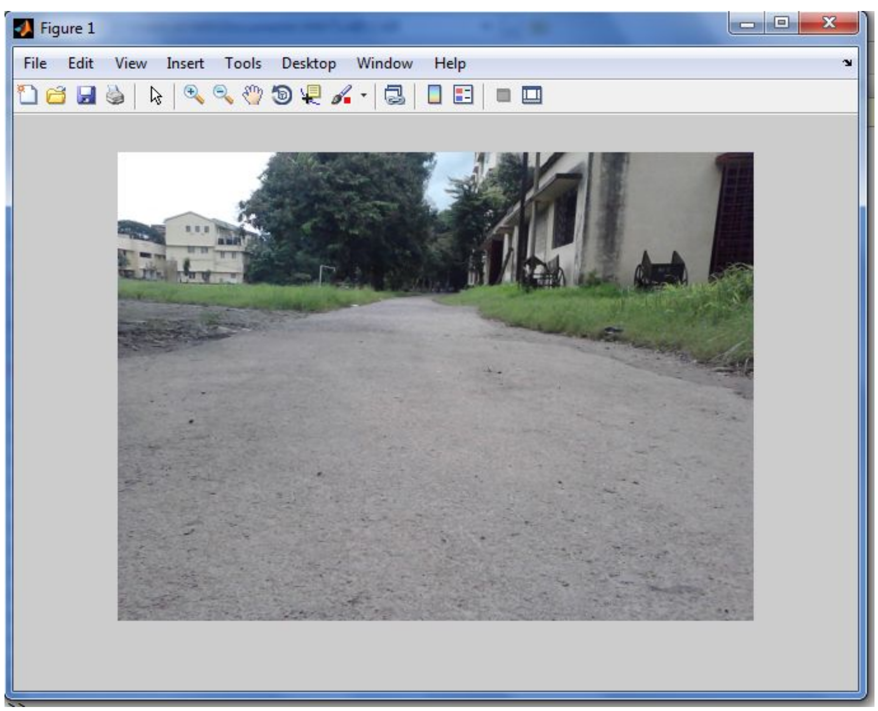


Fig. 11 – Input Image

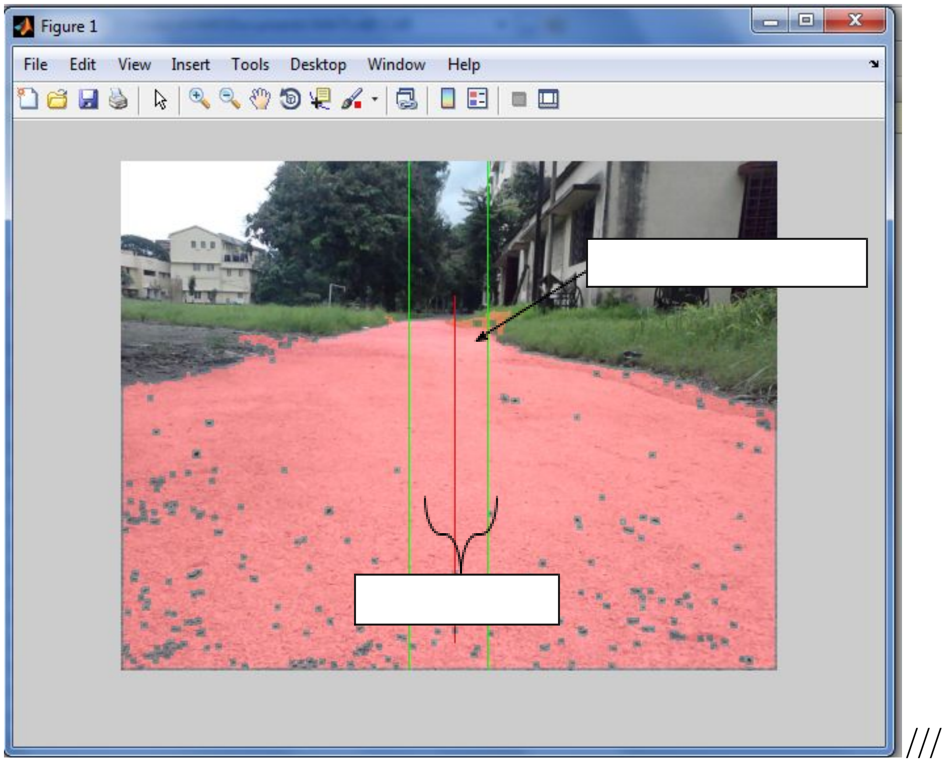


Fig. 12 – Detected Road Area

Advantages and Disadvantages :-

1. This algorithm works detects the road very effectively and is very robust.
2. It is immune to lighting variations i.e. it can detect roads during 12.00am and also 5.00 pm.
3. It can detect any kinds of roads i.e. mud roads and good highways.
4. It is comparatively slower.

But as we are driving our car in indoor environment(due to the problems listed below) where simple and faster algorithms can be used to detect the road so we have not used it but this algorithm can be used here also. It has even been tested in indoor environment with success.

1.4 - Approach III : -

This algorithm tracks the road based on color segmentation. There is a camera on board which sees the road and inputs the image in the laptop which processes the image to give a control output to the microcontroller which drives the motor. The program captures an image frame through the camera and processes it to find the road. The image has noise in it to reduce the noise level median filtering is done on the image matrix. Then it scans the entire matrix to identify the cells having a color matching with the color of the road. This is enough to find the road pixels but it also selects some non-road pixels. Then it forms a matrix with 1 and 0s(The points that lies within the color window are selected one and other points as zero). Then the entire binary image is eroded. The image is segmented to find only the road area and eliminates all the other areas. Thus we get the road area. Once we get the road we check whether the direction of the car matches with the direction of the direction of the road or not. If it matches then we drive the car in forward direction if the road is on its left then it drives the car towards left and vice- versa.

Advantages and Disadvantages:-

1. This algorithm is the fastest and very simple takes very small computation time.
2. It is tuned to particular color of the road this is not that dynamic to changes in lighting conditions and changes in color of the road.

1.4– SOFTWARES & PLATFORMS USED :

- Laptop

  - 2.1Ghz Intel Core to Duo Processor
  -1 GB DDR2 RAM
  -USB 2.0 Ports

- **MATLAB R2010A**

  -With **Image Acquisition & Image processing** Toolbox

- Microcontroller : Atmel **AT89S52**
- C compiler : **Keil**
- Programmer : **Prog ISP**

Chapter 2 – Communication Module

2.1- ABSTRACT :

The main objective is to translate the control signals coming serially from the processing unit and compare it with sensor outputs to give accurate motor controlling signals which drive the robot in proper direction. The programming in micro-controller is done in such a way that it takes the serial processing unit data simply in terms of some ASCII characters and after applying some algorithm ,the serial ASCII characters are translated into motor controlling signals which in turn drive the motors using motor deriver H bridge .

2.2- Approach :

- At first, we would like to built a fully autonomous vehicle on road. So, we try to design a sensor system which will in turn work with computer vision to drive the car. We try to build it as a rotating platform with ultra sound sensor which will send ultra sound in all directions. By receiving this signals, we can create a map of the objects around the car in real time.

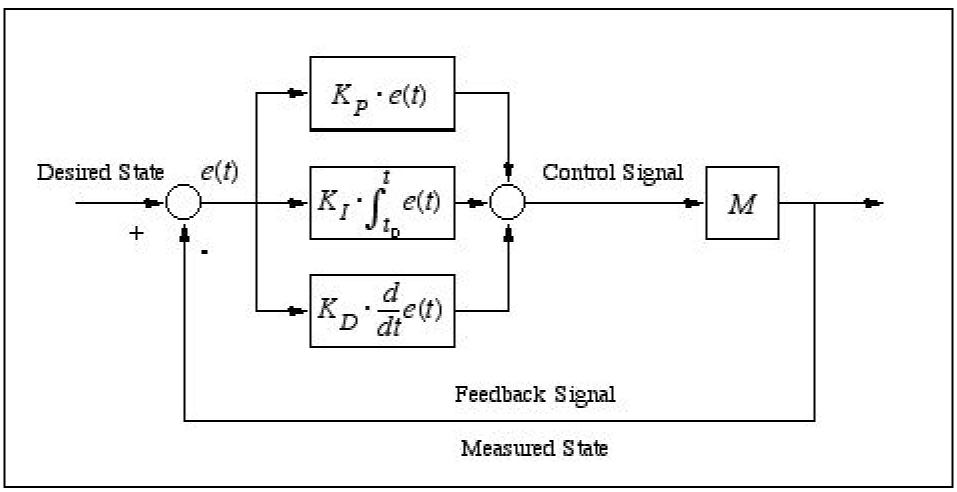
  But mounting the ultra-sound sensor on a rotating platform would require complex mechanical counter-measures to stabilize it. The sensor itself is also costly and their receivers to. Although this can be done but the amount of computation needs to be done to create a map from these data would create a big overhead for a system which would be also running image processing.

- Then we decided to reduce the overhead by bringing in it indoors as a robot and a sensor system is decided to implement using LED and LDR. The working principal is simple ,the voltage generated across LDR is less when more light incident on it than normal condition. So, we set a threshold voltage in order to determine whether the obstacle is in front of the robot or the walls are coming closer the robot. Just simple obstacle avoidance system.

  The problem lies with LED & LDR it-self. The normal LED are not powerful enough to be effective enough above 1m distance. This problem can be solved using IR LED but first it needs to be modulated. Digital IR LED boards are best for this but again they are costly. LDR is also not very sensitive and different light conditions would it's working as threshold voltages would not be constant. Another sensitive detector is their called Photo-detector which is very useful in this cases but costly and here we need more than 8 photo-detectors around the robot to successfully detect all the obstacles around the robot.

- We also try to incorporate PID control in the programming which will take the sensor outputs and compare it with image-processing output to give proper diving signals for smooth driving and always try to minimize the error which is it's main advantage.

When we have to implement PID logic in controller program, we have to specify the corresponding values of proportional gain, derivative gain & integral gain. But these values are not fixed and vary from system to system. So, it is an trial and error method to determine the suitable values as a wrong proportional gain value would increase steady-state error very much.



- Last we try some kind of algorithm in order to correct the false directions given by the processing unit due to the noise introduced by ambient light conditions. We take three arrays of a predefined length for left, right & straight movement and set the value of element of the array as 1 according to the output of image-processing unit. Then we look at the arrays for maximum no. of occurrence of 1and take it as the desired direction for the robot to move in.

It also could not solve the problem of noise alone as the size of the array is critical here. It also creates some sort of problem when the robot has to move in the corner of the path as the direction would come only one or two times so it might be possible that system nay overlook that decision.

2.3- Choice of Micro-controller :

There are certain criterions like

I. ***Task and Cost Efficiency***

II. ***Availability of Software Development Kits***

III. ***Complexity of the Configuration***

IV. ***Availability of the Micro-controller***

- Based on these criterions, we choose the Atmel **AT89S52** micro-controller chip, based on the 8051 micro-controller architecture, with high processing speed and comparable low cost. Here, the micro-controller communicates with the laptop through **UART** (Universal Asynchronous Receiver or Transmitter)

- 8051 micro-controllers are chosen over Atmega and PIC controllers because of cheap price and easily available programming soft-wares and programming kits.

- The **AT89S52** is a low-power, high-performance **CMOS** 8-bit microcontroller with 8K bytes of in-system programmable Flash memory. The AT89S52 provides the following standard features:
    1. *8K bytes of Flash*
    2. *256 bytes of RAM*
    3. *32 I/O lines*
    4. *three 16-bit timer/counters*
    5. *a full duplex serial port*

- The Power-down mode saves the RAM contents but freezes the oscillator, disabling all other chip functions until the next interrupt or hardware reset.

2.4 – Communication Module Hardware Circuit :

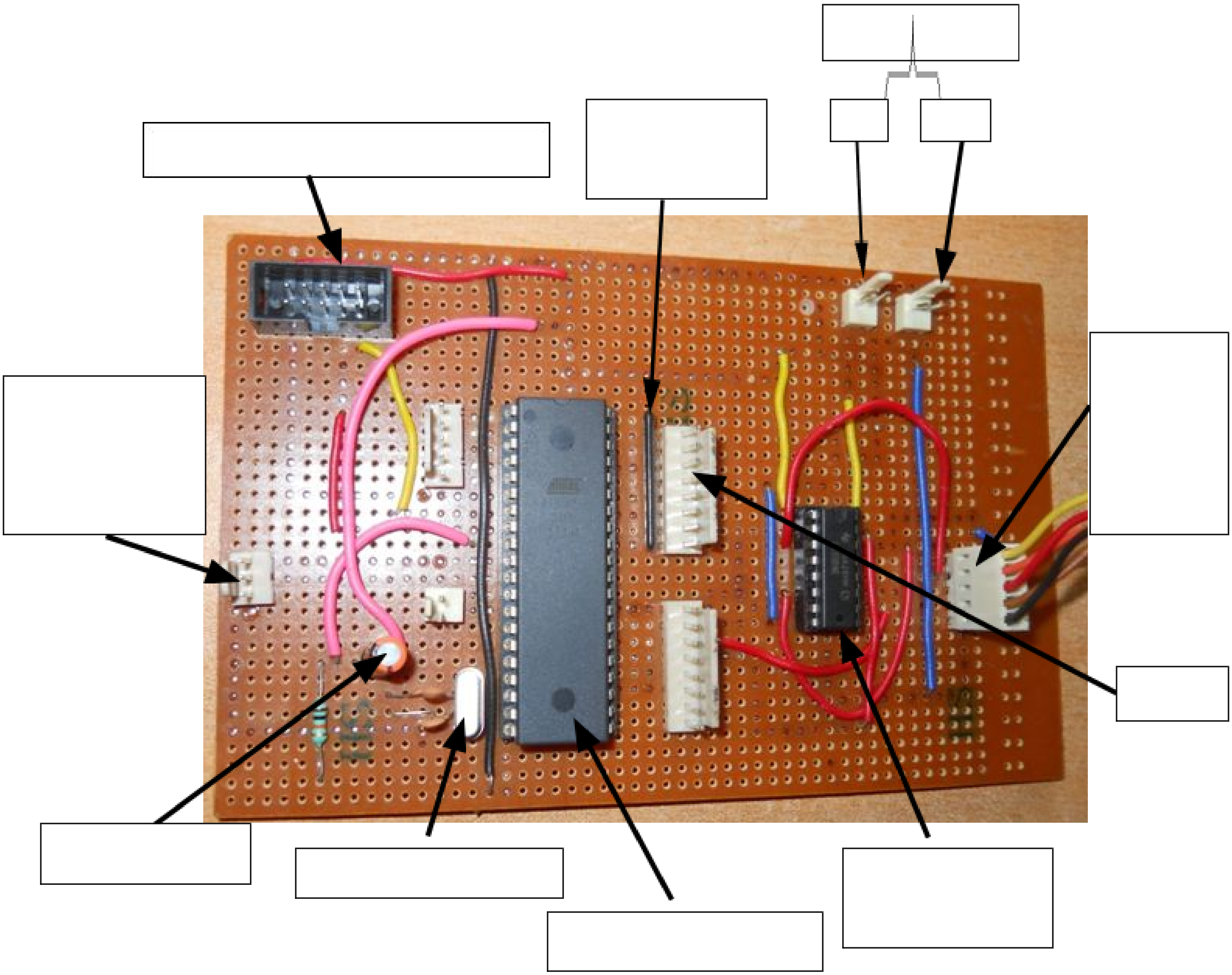


FIG 13 :- COMMUNICATION MODULE

2.5 – Serial Communication using Micro-controller :

- There is a full-duplex serial communication available in AT89S52 chip with dedicated SBUF register and Rx, Tx pins in the port 3. There is also a serial interrupt can be generated using this

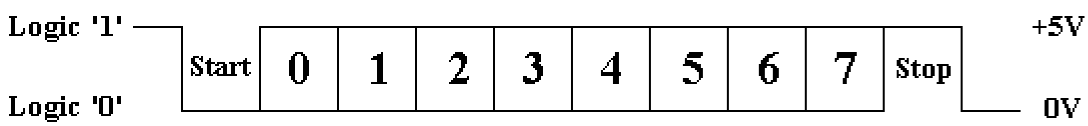
2.5.1- UART Protocol:

A **UART** (Universal Asynchronous Receiver or Transmitter) is the protocol with programming that controls a computer's interface to its attached serial devices.

- The **UART** also converts the bytes it receives from the computer along parallel circuits into a single serial bit stream for outbound transmission.

- Adds **Parity Bit** (if it's been selected) on outbound transmissions and checks the **parity of incoming bytes** (if selected) and discards the parity bit.

- The Bit-format of UART Protocol :



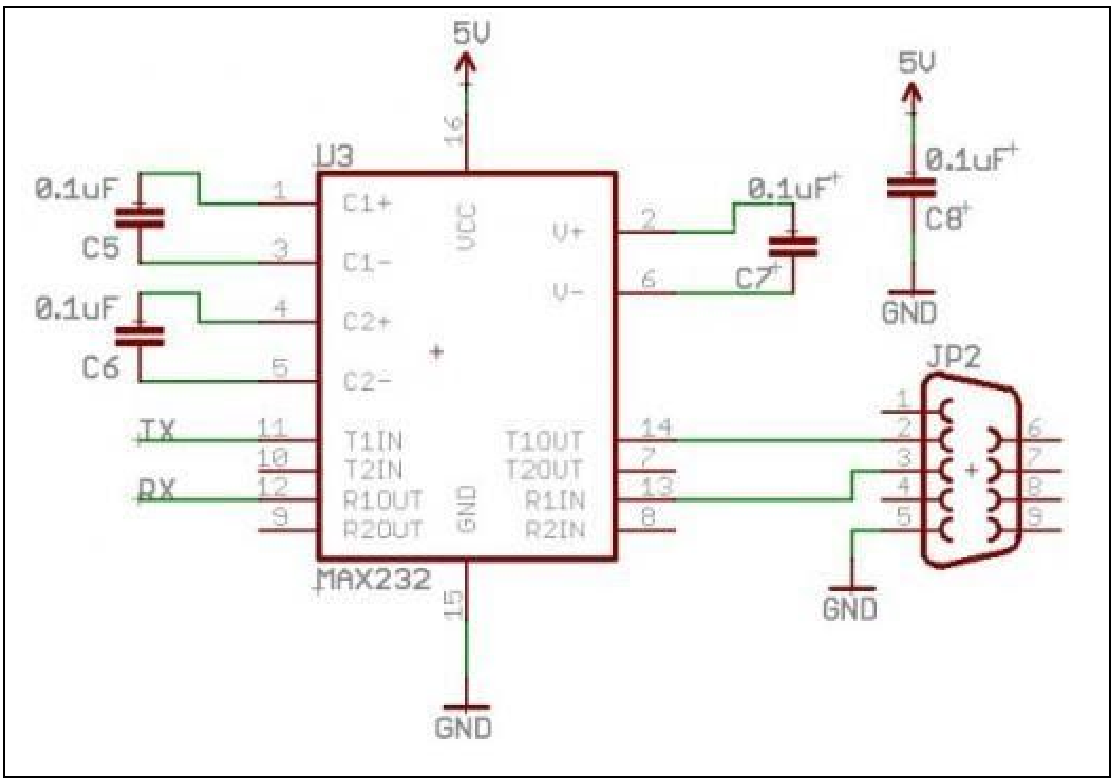
- In Serial Communication, high and low bits are represented by flowing voltage ranges:

Bit	*Voltage Range (Volt)*	
0	*+3*	*+25*
1	*-25*	*-3*

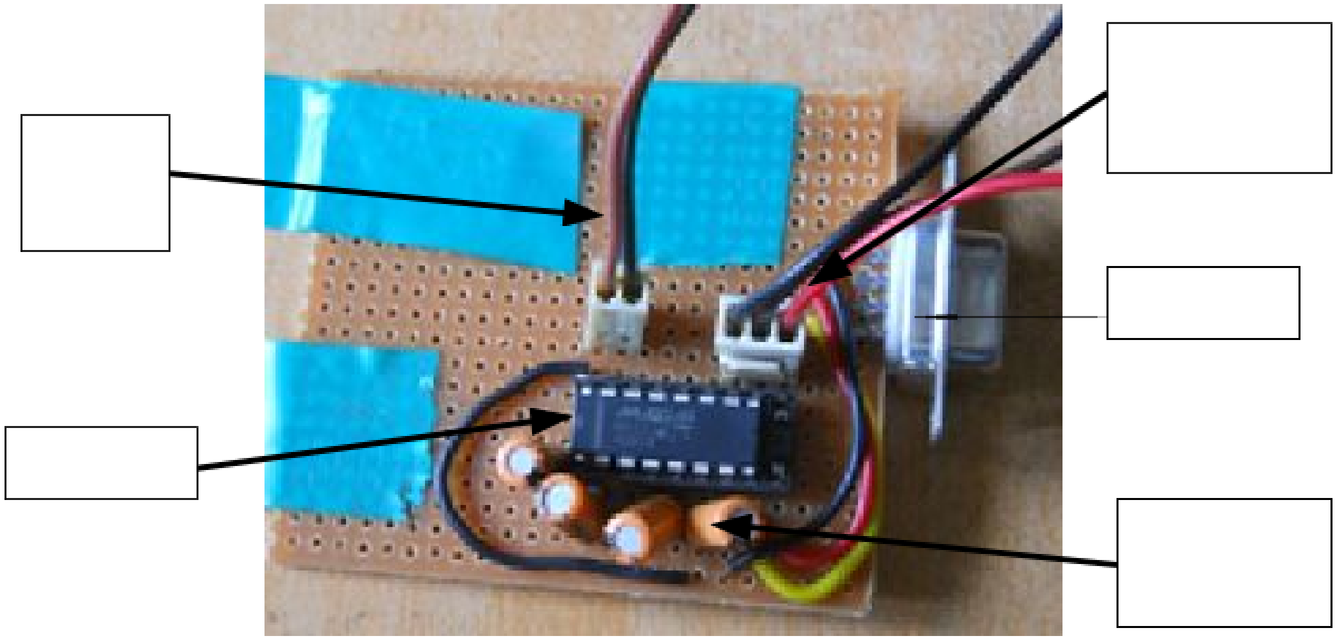
2.5.2 - Line Driver IC :

- As RS232 voltage levels are not compatible with that of TTL logic , a voltage converter is required while connecting serially to any microcontroller system. This converter converts the microcontroller output level to the RS232 voltage levels, and vice versa. IC **MAX232**, a well-known line driver, is very commonly used for this purpose.

- MAX232 has two sets of **line drivers** for transferring and receiving data. The line drivers used for transmission are called T1 and T2, where as the line drivers for receiver are designated as R1 and R2. The connection of MAX232 with computer and the controller is shown in the circuit diagram.



- The connection between a **PC** and microcontroller requires three pins, **RxD** (DB9, pin2), **TxD** (DB9, pin3) and ground (DB9, pin5) of the serial port of computer. **TxD** pin of serial port connects to **RxD** pin of controller via **MAX232**. Similarly, RxD pin of serial port connects to the **TxD** pin of controller through **MAX232**.

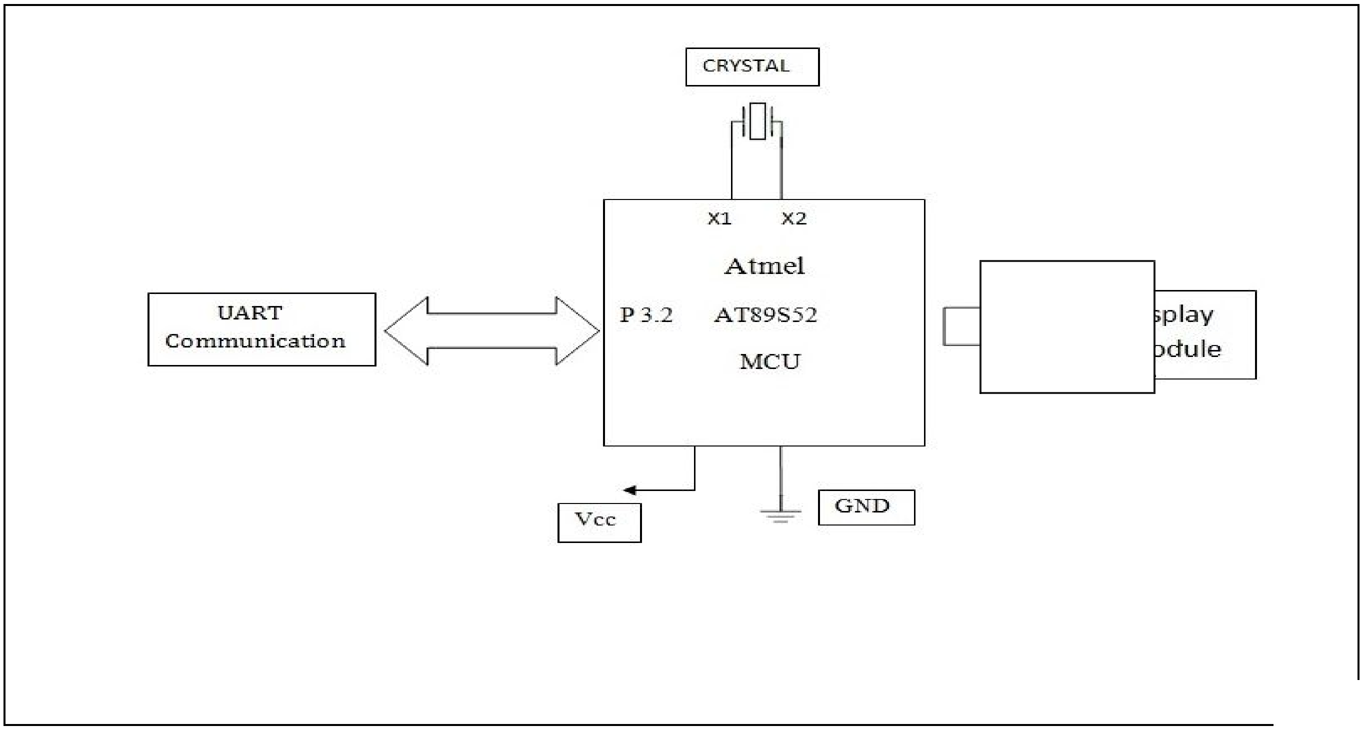


2.5.3 – Controller Programming For Serial Communication :

- An important parameter that needs to be considered during programming is the **Baud Rate** which is defined as number of bits transmitted or received per second. It is generally expressed in **bps** (bits per second). AT89S52 microcontroller can be set to transfer and receive serial data at different baud rates using software instructions.

- Timer1 is used to set the baud rate of serial communication for the microcontroller. For this purpose, Timer1 is used in **Mode2** which is an **8-bit Auto Reload** mode. In this project baud rate **9600bps** is used.

- For serial communication AT89S52 has registers **SBUF** and **SCON** (**Serial Control Register**). SBUF is an 8-bit register. For transmitting a data byte serially, it needs to be placed in the **SBUF** register. Similarly whenever a data byte is received serially, it comes in the SBUF register, i.e., SBUF register should be read to receive the serial byte.



- **SCON register** is used to set the mode of serial communication. The project uses Mode1, in which the data length is of 8 bits and there is a start and a stop bit. The SCON register is bit addressable register. The following table shows the configuration of each bit.

- **TI (transmit interrupt)** is an important flag bit in the SCON register. This indicates that the next byte can be transferred now. The TI bit is raised at the beginning of the stop bit.

- **RI (receive interrupt)** is also a flag bit of the SCON register. On receiving the serial data, the microcontroller skips the start and stop bits, and puts the byte is SBUF register. The RI flag bit is then raised to indicate that the byte has been rec

2.5.4 - Hyper Terminal :

- Hyper Terminal, a Windows application, can be used to receive or transmit serial data through RS232 and it is used in our project for testing the vehicle by sending the this application serially to micro-controller.

- To start a new connection, Select the following parameters :

Bit rate	:-	**9600bps**
Data bits	:-	**8**
Parity	:-	**None**
Stop bit	:-	**1**
Flow control	:-	**None**

- COM Port No. (USB to Serial) have to be selected during setting up the connection of the Hyper Terminal.

- In program, Timer1 is used with auto reload setting. The baud rate is fixed to **9600bps** by loading **TH1** to **0xFD**. The value **0x50** is loaded in the **SCON** register. This will initialize the serial port in **Mode1** of serial communication.

Chapter 3 – Car Module :

3.1- Objective :

Here, we have used a four wheel mini toy vehicle whose four wheels are controlled by dc motors in order to maneuver the car in forward-backward and left-right direction.

<u>Operation:</u>

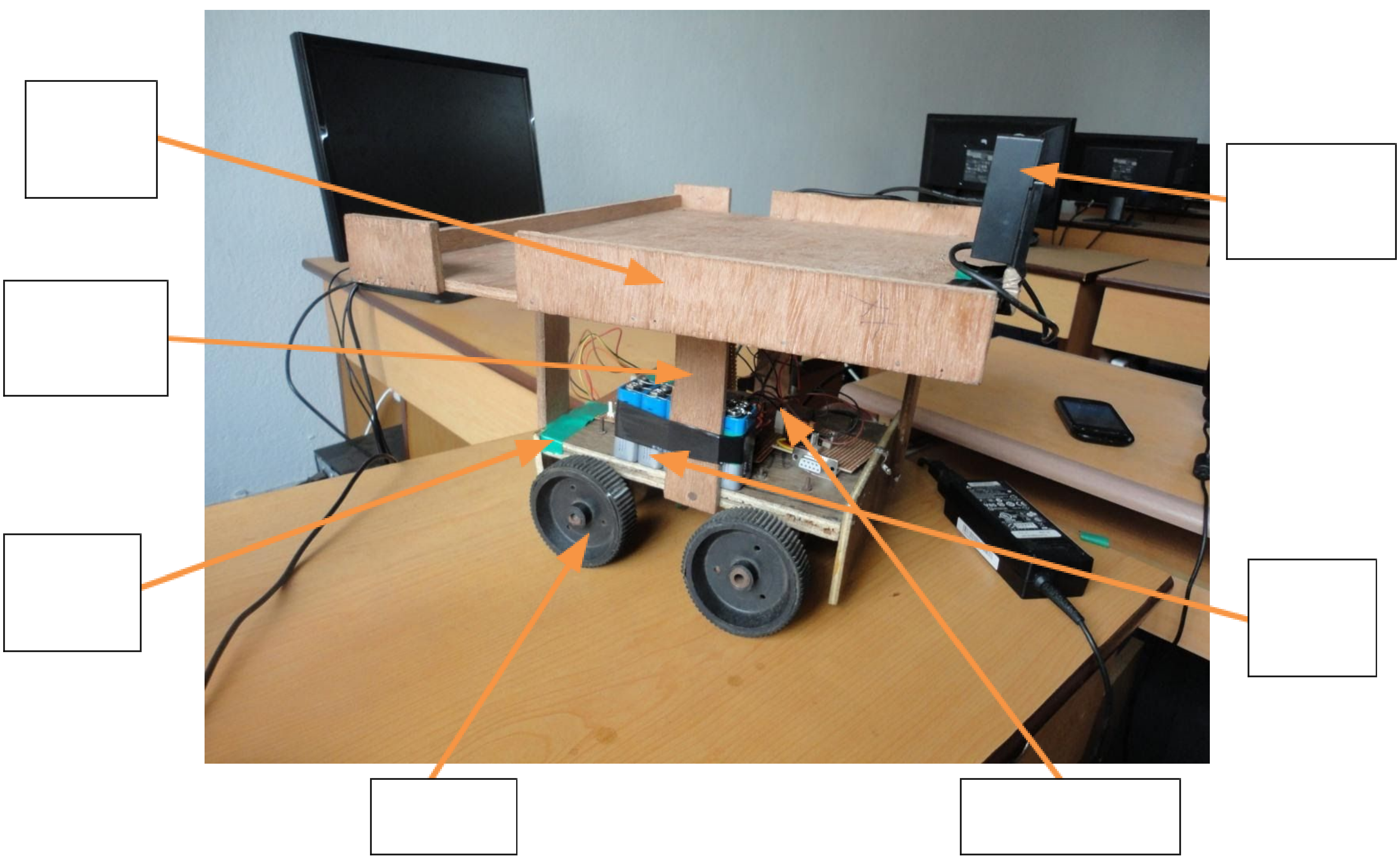
1. At first an H- Bridge Circuit has been created using L293D Driver IC.

2. The H-bridge arrangement is generally used to reverse the polarity of the motor.

3. It can also be used to 'break' the motor, where the motor comes to a sudden stop.

4. The motor's terminals are shorted, or to let the motor 'free run' to a stop, as the motor is effectively disconnected from the circuit.

FIG 16 :- CHASIS



The following table summarizes operation of DC motors (S1- S4) corresponding to the above shown diagram

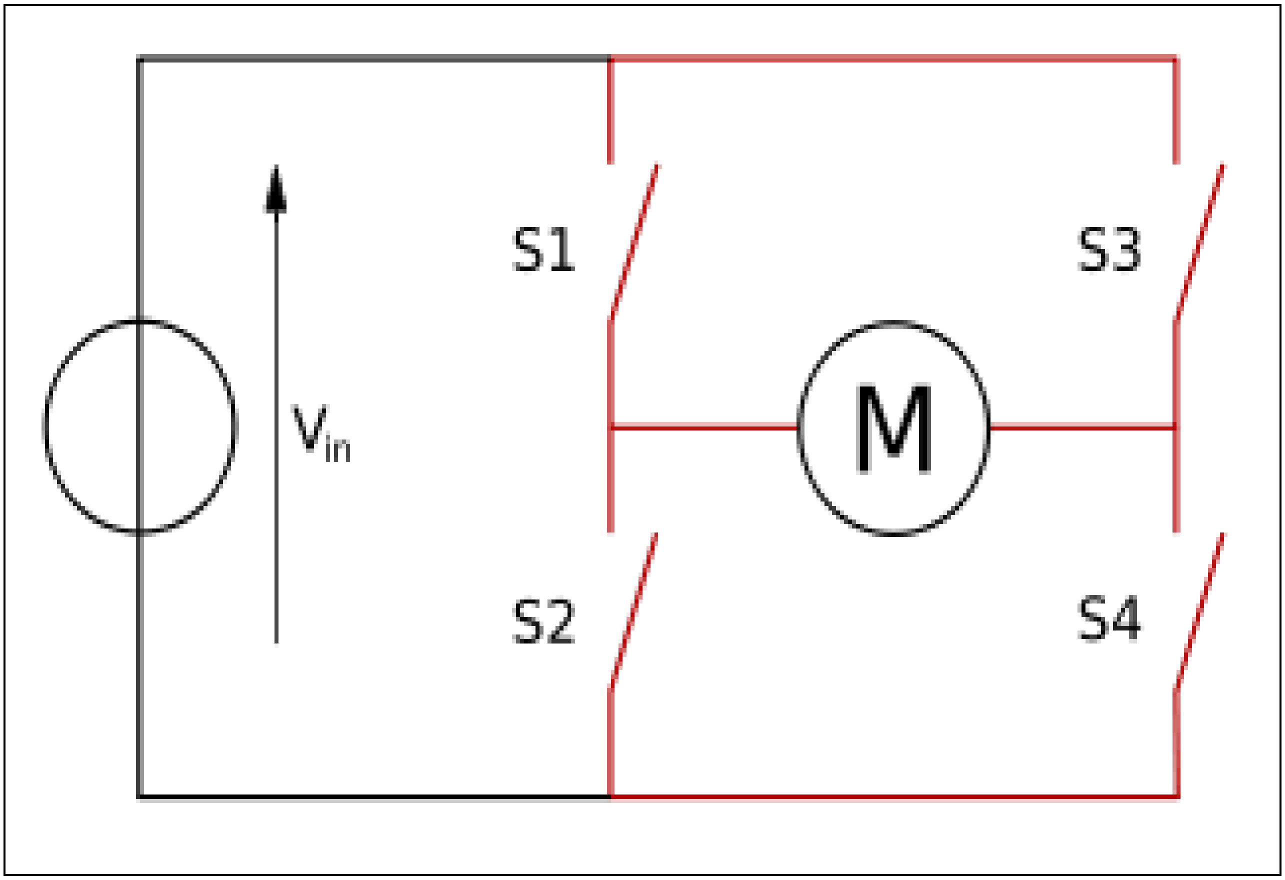
S1	S2	S3	S4	Result
1	1	1	1	Motor moves Straight
0	0	0	0	Motor Stops
0	1	0	1	Motor moves Left
1	0	1	0	Motor moves Right

3.2 - Motor Driver H-Bridge:

1. L293D is a dual H-bridge motor driver integrated circuit (IC).
2. Motor drivers act as current amplifiers since they take a low-current control signal and provide a higher-current signal.
3. This higher current signal is used to drive the motors.
4. L293D contains two inbuilt H-bridge driver circuits.
5. In its common mode of operation, two DC motors can be driven simultaneously, both in forward and reverse direction.
6. The motor operations of two motors can be controlled by input logic at pins 2 & 7 and 10 & 15.

7. Input logic 00 or 11 will stop the corresponding motor. Logic 01 and 10 will rotate it in clockwise and anticlockwise directions, respectively.

3.2.1 – Block Diagram :



Chapter 4 - Power supply Module

4.1- Power Supply:

- For Power supply unit, there are two options: from USB module as an adapter or from another external power supply. We have made a power supply of our own using adjustable voltage regular (LM 317) in order to obtain standard rating of voltage and current for micro-controller and DC motors.

- Following figure shows the power supply voltage :

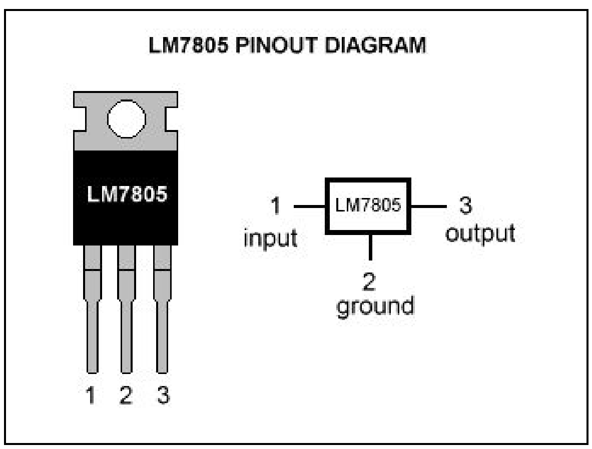


Figure 17: IC LM7805

- We can use 9V batteries in Parallel combination with a capcity of 550mAh . Similarly, we can also use pencile batteries connected in series.

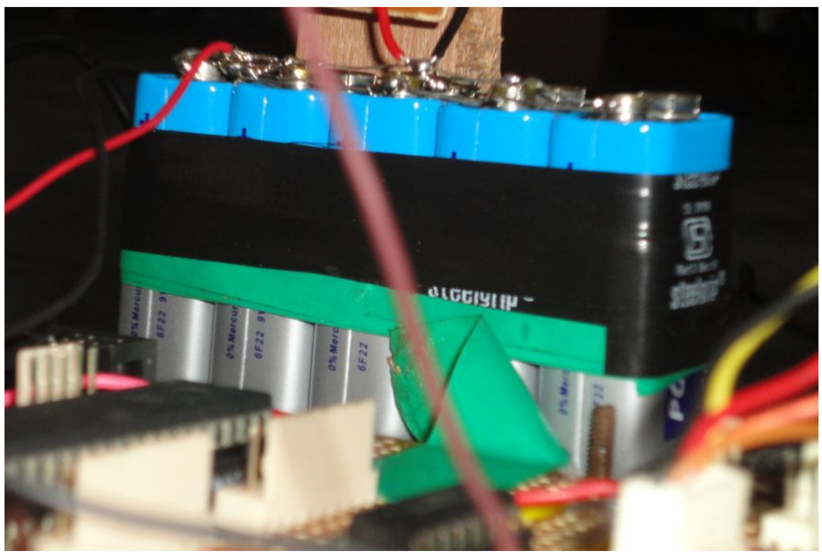


FIG 19 :- Battery Module

FUTURE SCOPE

- With the availability of ARM processor (1 GHz and higher) we will be able to make the entire setup mobile because at present the entire processing is done by the laptop thereby making the entire system not mobile enough.

- A 3D mapping sensor along with a camera will further enhance and improve the driving and sensing capability of the vehicle, which will help increase the accuracy to stay in its track thereby increasing the controllability of the system.

- At present we have planned to use a small scale wheeled robot. Real time implementation will be possible with the presence of a 4 wheeler car.

Conclusion :

First we thought to make a golf cart sized car and do this work in outside environment. But since we didn't get a golf cart so we had to sized down to a small chaises car. Now while using small robot on road there was a lot of jerking which was blurring the image too much to process. Then we used Lenovo bird eye PC camera which when used in sunny day the entire picture becomes completely white due to excessive illumination. Then we needed lidar and other high costly instruments but which was unavailable. For these reasons we shifted to indoor envirnent. In Indoor environment we faced problems like uneven lightening conditions. This uneven illumination caused unwanted edge points. We solved this problem removing the edge in places where there is excessive illumination. Then there were still unwanted small edge pixels due to noise. We solved this problem by removing small connected components as the noise edge pixels must be small in size compared t desired edge pixels. Now we are doing all this work in Matlab which is slow environment which was causing so the robot speed is slow. Then when the car goes fast then the camera picture becomes blurred so the speed had to be limited.