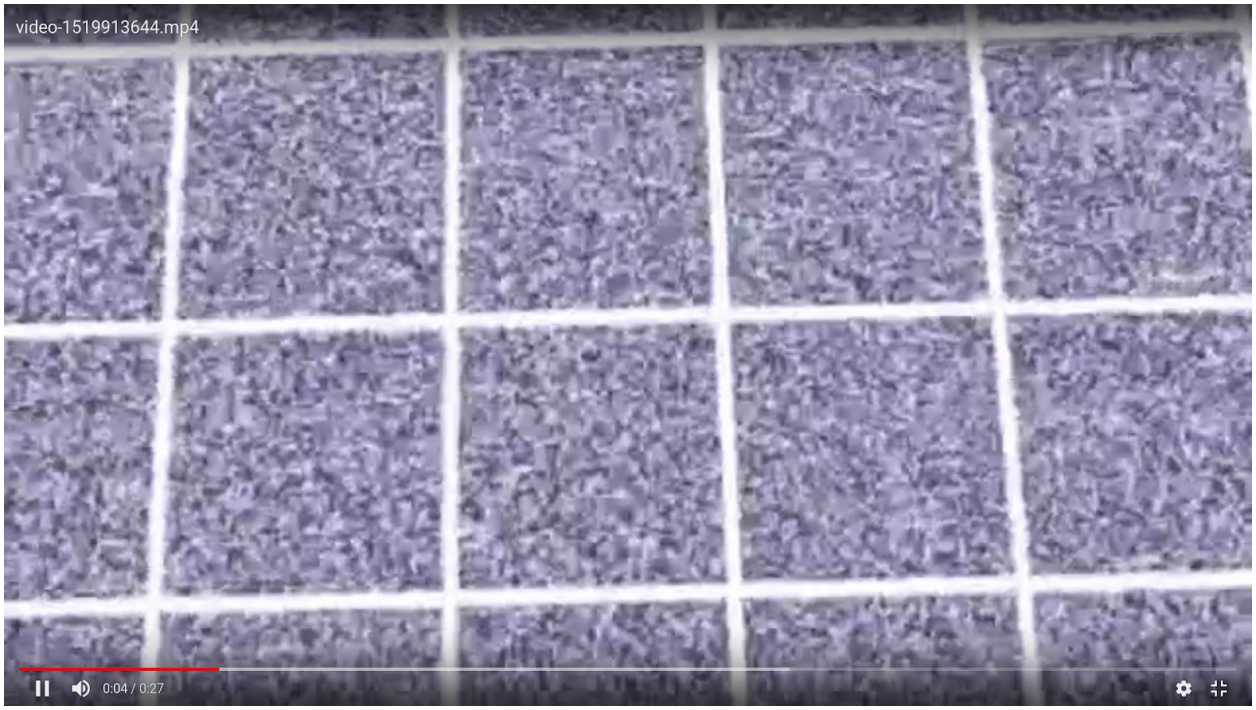
ARK-Software Tasks (Task-1)

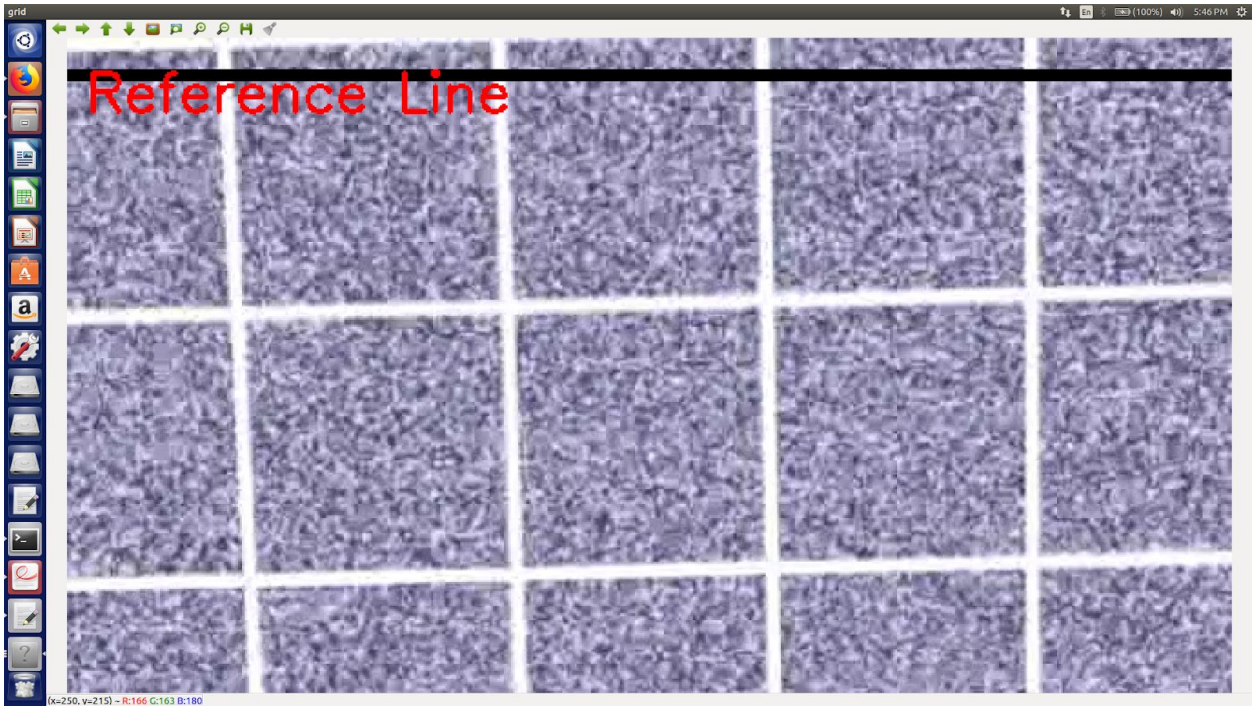
The aim of the task is to localize the quad moving over a grid. You will be provided with a camera feed of downward facing
camera. Assuming bottom left corner to be your origin you have to tell the coordinates of the bot at the end of the video
feed. It will be appreciated if you give real time coordinates as the video progresses. You have to only give x direction
coordinates. Expect random noise in video and quad motion.
This is a screenshot of the video.



This is a screenShot of the video.

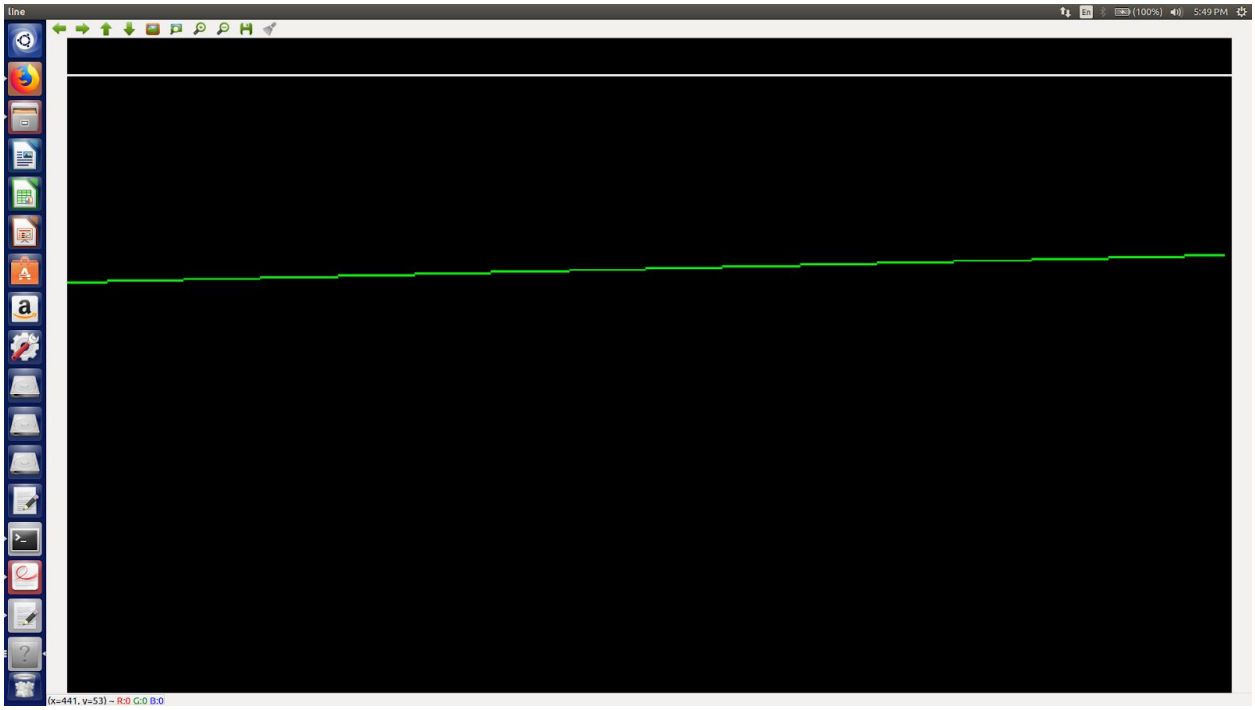
So from the image we can see that the difficult part of this task was the reduce the noise.

First I define a reference line with respect to which I will calculate the Y coordinate.



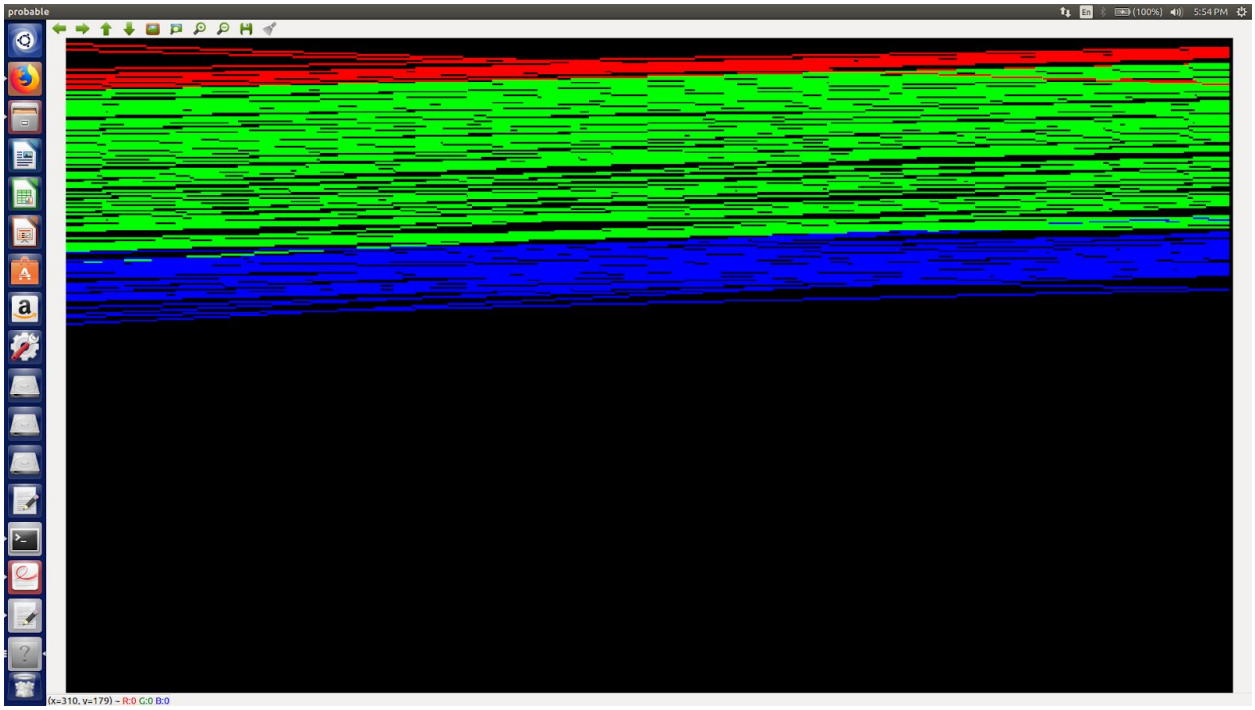
Img-1

Now to make it simple , i just filter th upper most line and the reference line, like-



Img-2

And then define three regions like -



Img-3

Actually the reference line that I defined earlier is the boundary of the red and green region.

Now if the upper line that I filtered (see Img-2) goes from green region to red region that will implies the lines in the video are going upward and are crossing the reference line. So the Y coordinate will decrease by One.

And if the line goes from blue region without crossing the green region that will imply that the upper line goes downward and a new upper line has come. That means y coordinate is increasing.

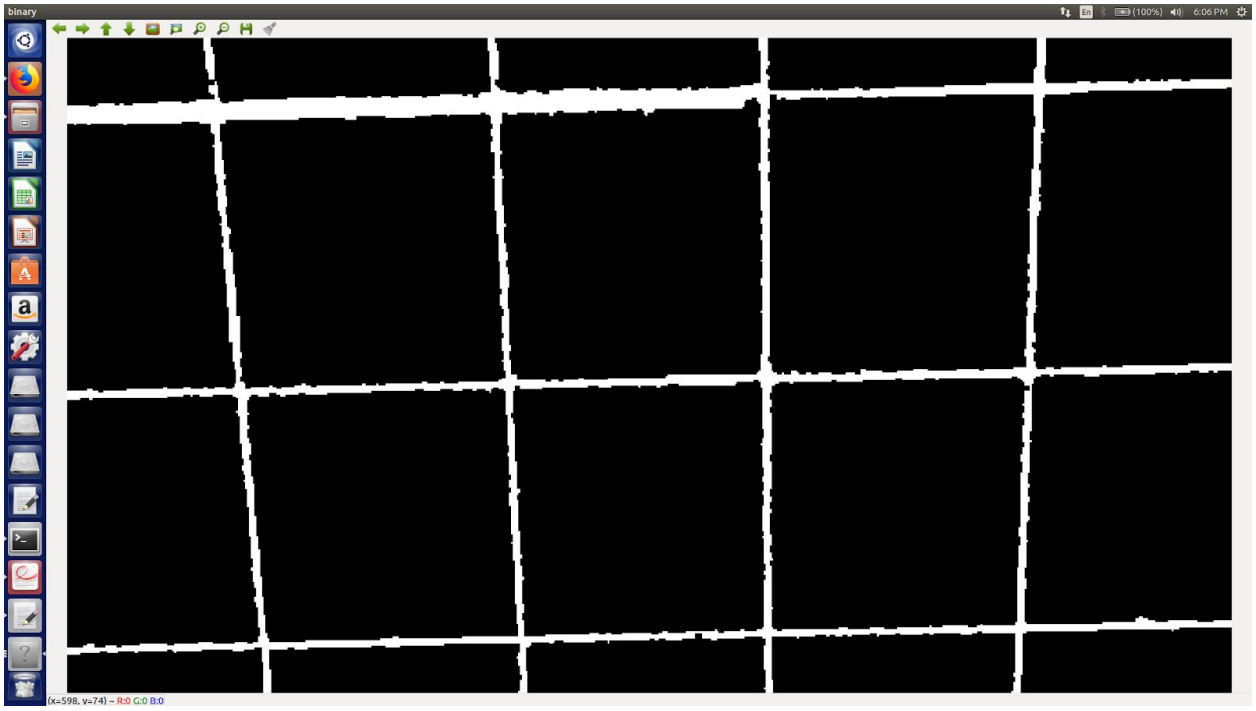
How did I filter--

Firstly I convert the video into a binary one using a Threshold 220, here the maximum noise was reduced.

```
vid >> frame;
Mat binary(frame.rows,213,CV_8UC3,Scalar(0,0,0));
 for(i=0;i<frame.rows;i++)
   {
          for(j=0;j<frame.cols;j++)
   {
          if(frame.at<uchar>(i,j)>220)    //threshold is 220
          binary.at<uchar>(i,j)=255;
          else
```

```
            binary.at<uchar>(i,j)=0;
        }
        }
```

After converting it to binary it will look like this



Img-4

Then I tried erosion and dilation to make it clear but it did not work properly .

Then I tried a new thing that I never had tried before. I fitted a regression line corrosponding to the upper line using linear regression equation.

I plotted regression line of Y on X . The regression line will look like in Img-2(Green Line).

About Linear regression equation --

$\sqrt{2181}\sqrt{2581}$ = $\frac{\quad}{46.70 \times 50.80}$ = 0.43.

§14.7 REGRESSION

By *regression* of a variable y (say) on another variable x (say), we mean the dependence of y on x, on the average. In bivariate analysis, one of the major problems is prediction of the value of the dependent variable y when the value of the independent variable x is known. The problem becomes simplified if we can express y as a mathematical function of x, $y = h(x)$, say. Then this equation is called the regression equation of y on x. In the simplest case, when y is linearly related with x, either exactly or approximately, we can write

$$y = a + bx,$$

so that $a + bx_0$ is the predicted value of y when $x = x_0$.

Here we confine our discussion to linear regression only.

14.7.1 Derivation of linear regression equation

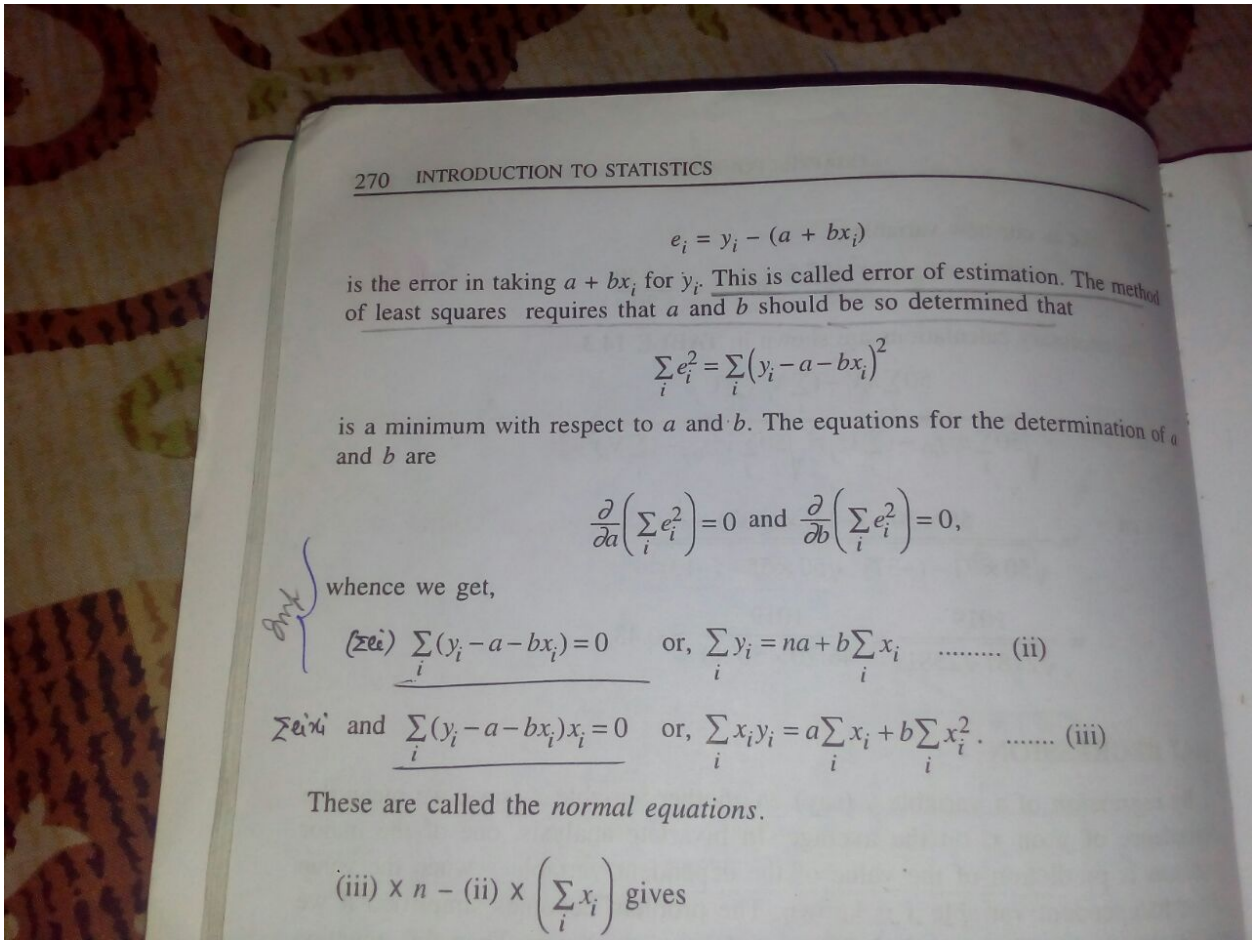
Let the linear regression equation of y on x be

$$y = a + bx. \qquad \ldots\ldots (i)$$

Since we would like to use this equation for prediction purposes, the constants a and b have to be estimated on the basis of observed values of x and y. Suppose we are given n pairs of values (x_i, y_i), $i = 1(1)n$, of x and y. From among different methods that are available for determination of a and b, we use here the method of least squares (described at the end of this chapter) which has many desirable properties.

When $x = x_i$, the observed value of y is y_i and the predicted value of y is $a + bx_i$. So,

Img-5



$$e_i = y_i - (a + bx_i)$$

is the error in taking $a + bx_i$ for y_i. This is called error of estimation. The method of least squares requires that a and b should be so determined that

$$\sum_i e_i^2 = \sum_i (y_i - a - bx_i)^2$$

is a minimum with respect to a and b. The equations for the determination of a and b are

$$\frac{\partial}{\partial a}\left(\sum_i e_i^2\right) = 0 \text{ and } \frac{\partial}{\partial b}\left(\sum_i e_i^2\right) = 0,$$

whence we get,

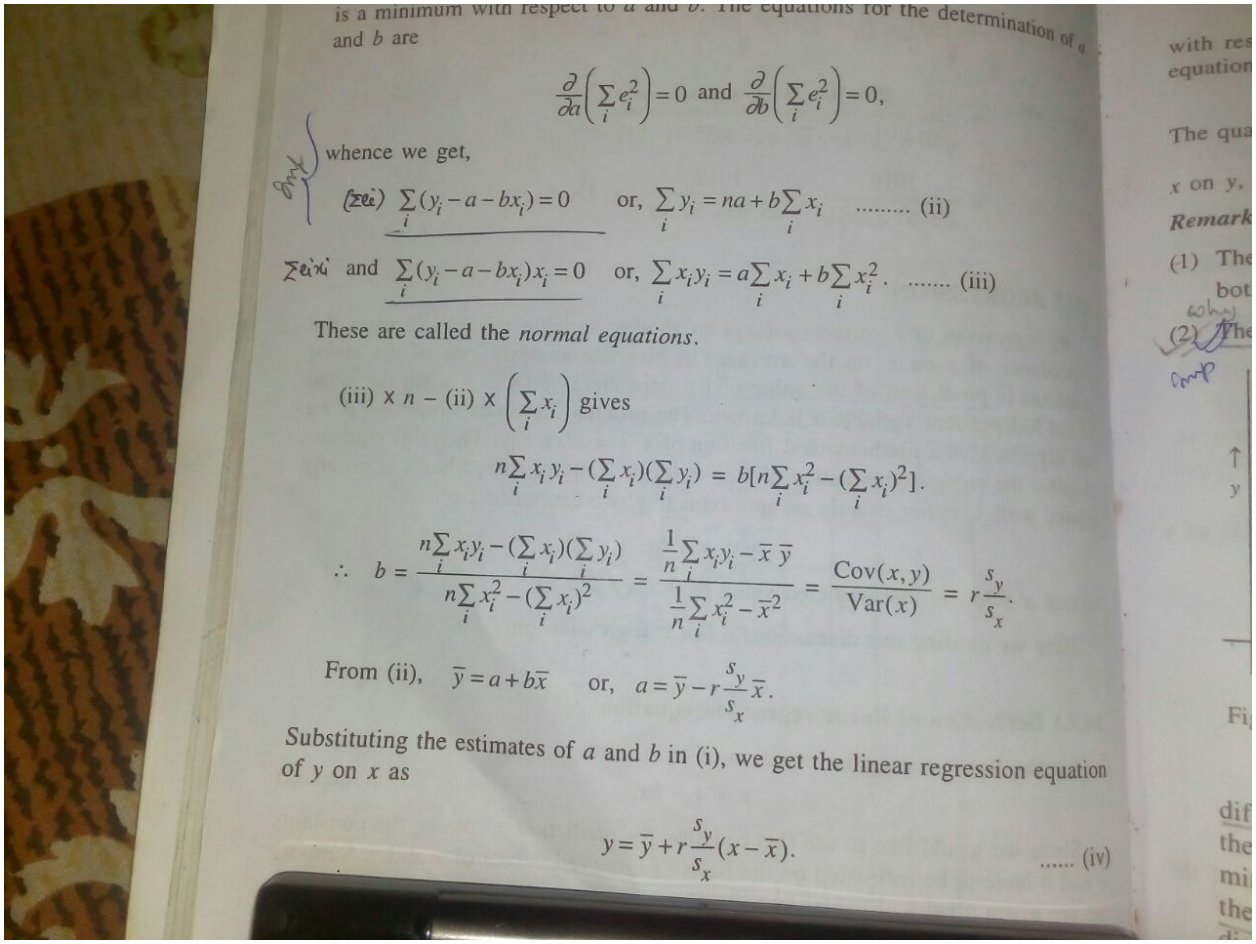
$$\sum_i (y_i - a - bx_i) = 0 \quad \text{or,} \quad \sum_i y_i = na + b\sum_i x_i \quad \ldots\ldots \text{(ii)}$$

and $$\sum_i (y_i - a - bx_i)x_i = 0 \quad \text{or,} \quad \sum_i x_i y_i = a\sum_i x_i + b\sum_i x_i^2. \quad \ldots\ldots \text{(iii)}$$

These are called the *normal equations*.

(iii) × n − (ii) × $\left(\sum_i x_i\right)$ gives

Img-6



is a minimum with respect to a and b. The equations for the determination of a and b are

$$\frac{\partial}{\partial a}\left(\sum_i e_i^2\right)=0 \text{ and } \frac{\partial}{\partial b}\left(\sum_i e_i^2\right)=0,$$

whence we get,

$$\sum_i (y_i - a - bx_i) = 0 \quad \text{or,} \quad \sum_i y_i = na + b\sum_i x_i \quad \ldots\ldots\ldots \text{(ii)}$$

and $$\sum_i (y_i - a - bx_i)x_i = 0 \quad \text{or,} \quad \sum_i x_i y_i = a\sum_i x_i + b\sum_i x_i^2. \quad \ldots\ldots\ldots \text{(iii)}$$

These are called the *normal equations*.

(iii) $\times n$ − (ii) $\times \left(\sum_i x_i\right)$ gives

$$n\sum_i x_i y_i - (\sum_i x_i)(\sum_i y_i) = b[n\sum_i x_i^2 - (\sum_i x_i)^2].$$

$$\therefore \quad b = \frac{n\sum_i x_i y_i - (\sum_i x_i)(\sum_i y_i)}{n\sum_i x_i^2 - (\sum_i x_i)^2} = \frac{\frac{1}{n}\sum_i x_i y_i - \bar{x}\,\bar{y}}{\frac{1}{n}\sum_i x_i^2 - \bar{x}^2} = \frac{\text{Cov}(x,y)}{\text{Var}(x)} = r\frac{s_y}{s_x}.$$

From (ii), $\bar{y} = a + b\bar{x}$ or, $a = \bar{y} - r\frac{s_y}{s_x}\bar{x}$.

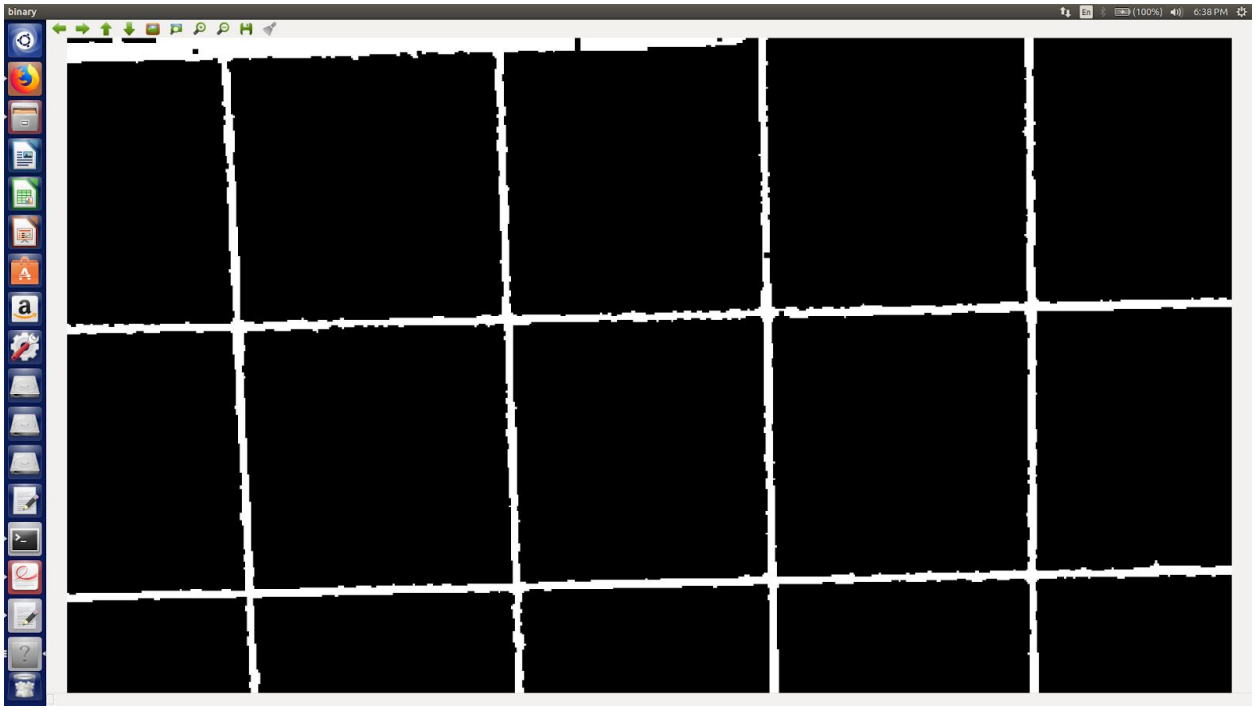
Substituting the estimates of a and b in (i), we get the linear regression equation of y on x as

$$y = \bar{y} + r\frac{s_y}{s_x}(x - \bar{x}). \quad \ldots\ldots \text{(iv)}$$

Img-7

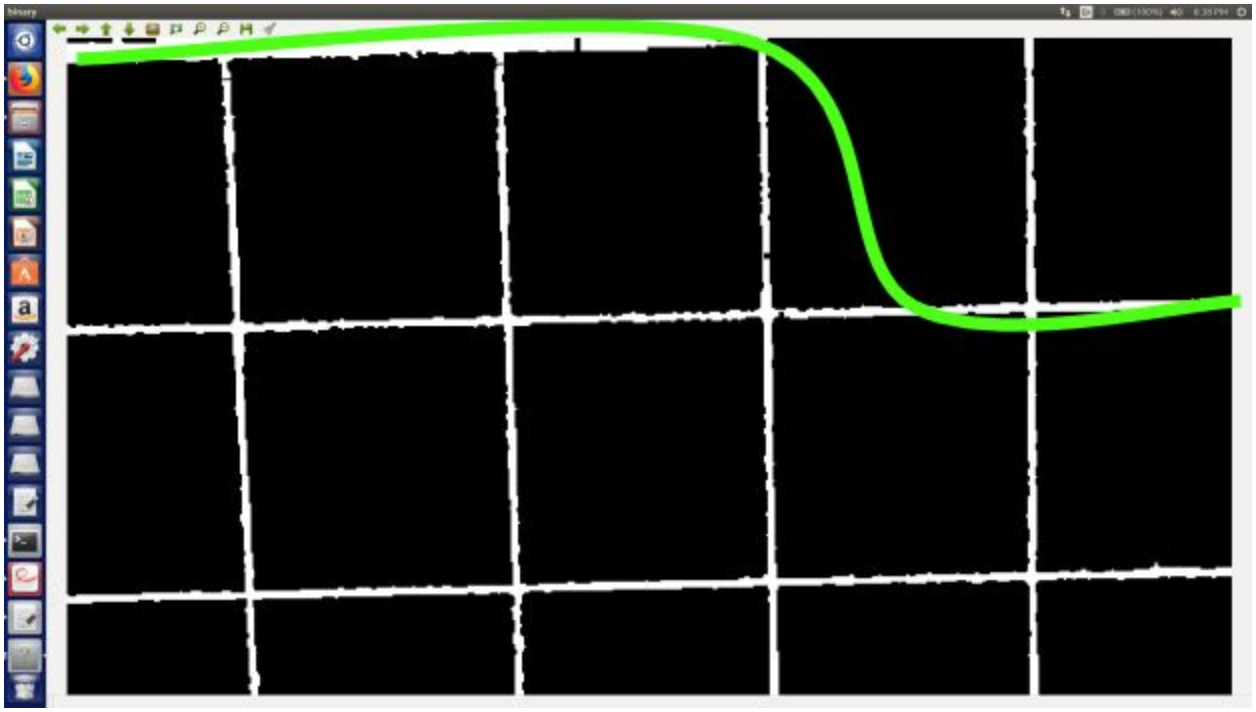
Ref: Introduction to Statistics By- PK Giri And Jibon Banerjee.

The only problem till now in this way is this case

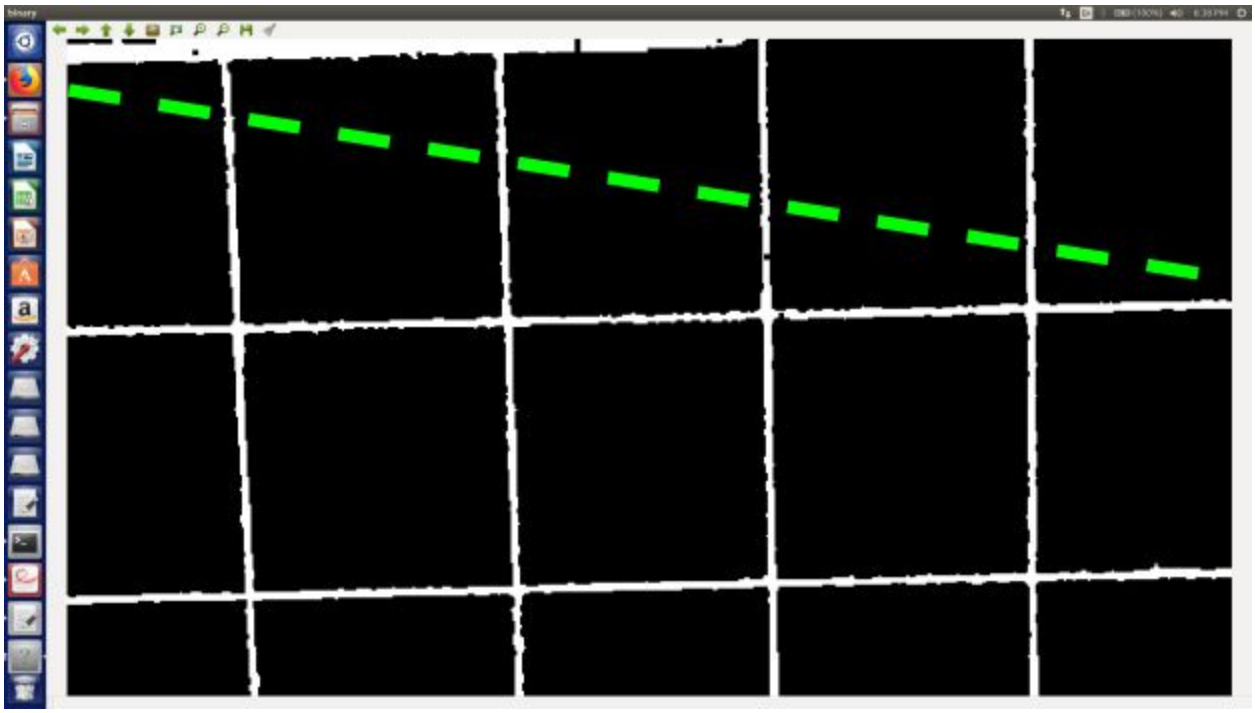


Img-8

Because here the point taken to calculate the regression line will be those on the green curve in the image below



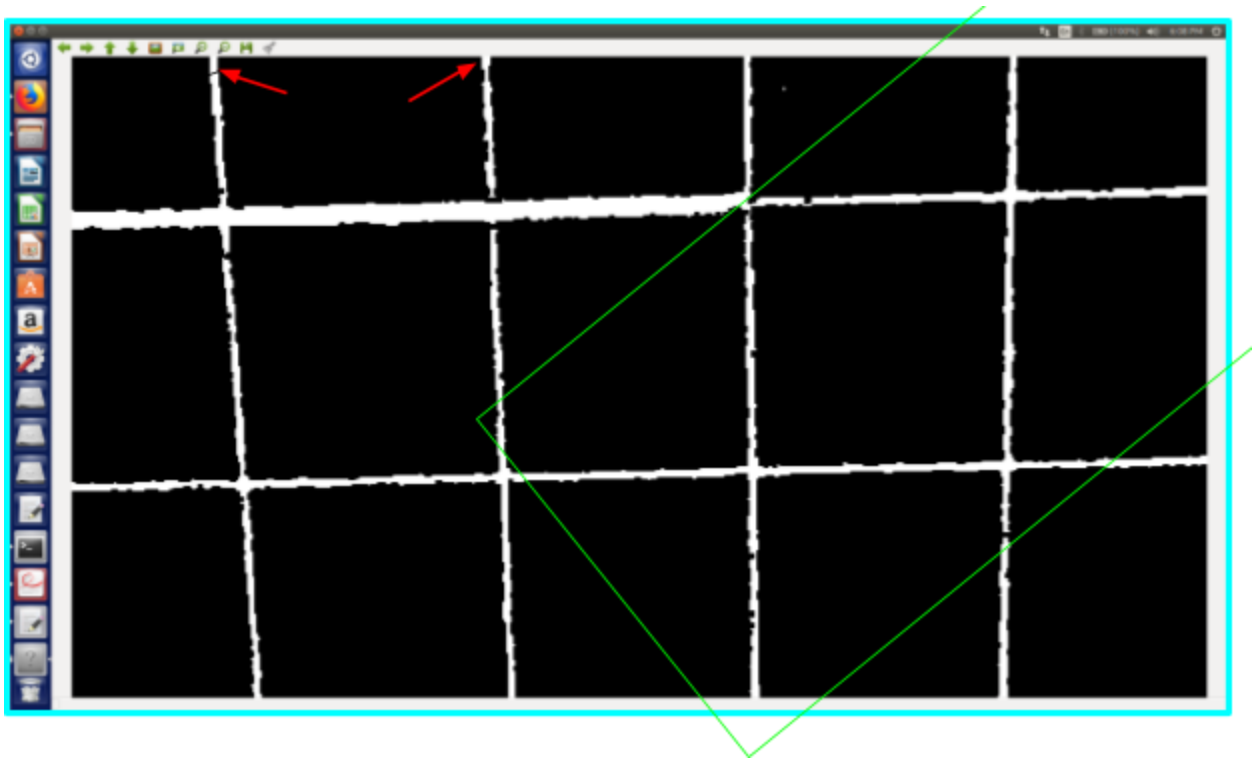
And the regression line will be someWhat like this--



Img-10

In this case, i have calculate approximately the maximum slope and I have skipped that case i.e in that case computer will assume that no line is coming in that case .

The outlier points will also cause problem. So I had to skipped also the outliers in my code.



Img-8 (outliers)

Code-

```
#include "opencv2/highgui/highgui.hpp"      // Including necessary libraries
#include "opencv2/imgproc/imgproc.hpp"
#include "opencv2/core/core.hpp"
#include<iostream>
#include<stdio.h>
#include<stdlib.h>
#include<math.h>
#include<string>
#include<queue>

using namespace cv;
using namespace std;

double X,Y;
```

```
int previousPosition=1;        //initializing the previous position by 1
int currentPosition;
Mat frame;
Mat probable(358,638,CV_8UC3,Scalar(0,0,0));

int yCoordinate=0;
//returns the mean of x
double meanX(queue<Point> data)
  {
        int i;
        double sum=0,mean;
        Point temp;
        for(i=0;i<data.size();i++)
        {
        sum+=data.front().x;
        temp=data.front();
        data.pop();
        data.push(temp);
        }
        cout << "sumX" <<sum<<endl;
        mean=(sum/i);
        return mean;
  }

//returns the mean of y
double meanY(queue<Point> data)
  {
        int i;
        double sum=0,mean;
        Point temp;
        for(i=0;i<data.size();i++)
        {

        sum+=data.front().y;
        temp=data.front();
        data.pop();
        data.push(temp);
        }
        mean=(sum/i);
        return mean;
  }

//returns the covarience of the data
double Covarience(queue<Point> data)
  {
        double XY,sum=0,mean,cov;
```

```
        int i;
        Point temp;
        XY= X*Y;
        for(i=0;!data.empty();i++)
        {
        sum+=(data.front().x * data.front().y);
        temp=data.front();
        data.pop();
        //data.push(temp);
        }
        mean=(sum/i);
        printf("XY=%lf\n",XY);
        printf("%lf\n",mean);
        cov=(mean-XY);
        return cov;
 }
```

//returns the varience of the x from the data

```
double varienceX(queue<Point> data)
 {
        int i;
        Point temp;
        double sum=0,X,sqX,mean,varX;
        X=meanX(data);
        sqX=X*X;
        for(i=0;i<data.size();i++)
        {
        sum+=(data.front().x * data.front().x);
        temp=data.front();
        data.pop();
        data.push(temp);
        }
        mean=(sum/i);
        varX=(mean-sqX);
        return varX;
 }
```

// this function is checking the position of the regression line wheather in blue,green or red region

```
int checkWhere(Mat Line)
 {
        int i;
        for(i=0;i<Line.rows;i++)
        {
        if(Line.at<Vec3b>(i,Line.cols/2)[1]>240)break;
```

```
        }
        if(i>20 && i<110)return 1;
        else if(i<=20) return 0;
        else return 2;
 }


Mat regressionLine1(Mat img)
 {
        int i,j;
        Point p1,p2;

        double covXY,varX,slope;
        int x1,x2,y1,y2;
        x1=0;
        x2=635;
        Point temp1;
        queue<Point> data;
        for(i=0;i<img.cols;i+=20)
        {
        for(j=0;;j++)
        {
        if(img.at<Vec3b>(j,i)[0]==255)
                {
                if(img.at<Vec3b>(j,i)[1]==255)
                {
                if(img.at<Vec3b>(j,i)[2]==255)
                {
                if(j==0)
                {
                        i+=10;      // in this way we are neglecting the out liers
                        continue;
                }
                else
                {
                        temp1.x=i;
                        temp1.y=-j;
                        data.push(temp1);   // if it is not an outlier the we keep it in
our data storage i.e the queue named data
                        break;
                        }

                }
                }
                }
        }
```

```
}
X=meanX(data);    // calculate the mean of x coordinate of the corrosponding x coordintes in our scatter diagram

Y=meanY(data);    // calculate the mean of x coordinate of the corrosponding x coordintes in our scatter diagram

covXY=Covarience(data); //calculate the covarience of the bivariate dataset I have

varX=varienceX(data);   // calculating the varience of x

slope=-(covXY/varX);   //calculating the slope of the regession line

cout <<" mean of X = " << X <<endl;
cout <<" mean of Y = " << Y <<endl;
cout <<" covarience= " << covXY<<endl;
cout <<" varience =  " <<varX<<endl;
cout <<" slope =        " <<slope<<endl;

y1=Y+slope*(x1-X);
y2=Y+slope*(x2-X);


cout <<"  Y1 = "<<y1<<endl;
cout <<"  Y2 = "<<y2<<endl;

p1.x=x1;
p2.y=-y1+5;
p2.x=x2;
p1.y=-y2+5;

Mat reg(img.rows,img.cols,CV_8UC3,Scalar(0,0,0));
if(abs(p1.y-p2.y)<60)    //eliminating the 1st problem i have in this way
{
line(reg,p1,p2,Scalar(0,255,0),1);   //drawing the regression line
if(checkWhere(reg)==1)
line(probable,p1,p2,Scalar(0,255,0),1);  //checking in which region the regression line is
else if(checkWhere(reg)==0)
```

```
        line(probable,p1,p2,Scalar(0,0,255),1);
        else if(checkWhere(reg)==2)
        line(probable,p1,p2,Scalar(255,0,0),1);
        }
        return reg;
 }


int main()
 {
        namedWindow("grid",WINDOW_NORMAL);     // creating necessary
windows
        namedWindow("binary",WINDOW_NORMAL);
        namedWindow("line",WINDOW_NORMAL);
        namedWindow("probable",0);

        VideoCapture vid("GRID.mp4");   //reading the video

        Mat last;
        Mat Line;
        int i,j;

        while(1)
        {
        vid >> frame;
        Mat binary(frame.rows,frame.cols,CV_8UC3,Scalar(0,0,0));
        for(i=0;i<frame.rows;i++)
        {
        for(j=0;j<frame.cols;j++)
        {
//converting it to binary using a threshold 220

                if(frame.at<Vec3b>(i,j)[0]>220 && frame.at<Vec3b>(i,j)[1]>220 &&
frame.at<Vec3b>(i,j)[2]>220)
                {
                binary.at<Vec3b>(i,j)[0]=255;
                binary.at<Vec3b>(i,j)[1]=255;
                binary.at<Vec3b>(i,j)[2]=255;
                }
                else
                {

                binary.at<Vec3b>(i,j)[0]=0;
                binary.at<Vec3b>(i,j)[1]=0;
                binary.at<Vec3b>(i,j)[2]=0;
                }
```

```
        }
        }

        Mat element=getStructuringElement(MORPH_RECT,Size(3,3),Point(0,0));
//eroding to make it some clear
        erode(binary,binary,element);

        last=binary.clone();
//getting the regression line by calling the function
        Line=regressionLine1(last);

        currentPosition=checkWhere(Line);
        printf("        currentPosition = %d  previousPosition = %d
",currentPosition,previousPosition);

//if the line goes from blue region to red directly the y coordinate increments
        if(currentPosition==1 && previousPosition==0)
        {
        yCoordinate++;
        }
//if the line goes from green region to red directly the y coordinate decrements
        else if(currentPosition==0 && previousPosition==1)
        {
        yCoordinate--;
        }
        previousPosition=currentPosition;


        line(Line,Point(0,20),Point(Line.cols,20),Scalar(255,255,255),1);
        line(frame,Point(0,20),Point(frame.cols,20),Scalar(0,0,0),5);
        putText(frame,"Reference
Line",Point(10,40),FONT_HERSHEY_SIMPLEX,1,Scalar(0,0,255),2);

        printf("******************* current Y coordinate =  %d
**********\n",yCoordinate);

        imshow("line",Line);
        imshow("grid",frame);
        imshow("binary",binary);
        imshow("probable",probable);

        if(waitKey(300)>=0)break;
        }
```

```
        waitKey(0);
}
```

And the out put will be like this