CV MINI PROJECT
Curved Lane Detection in Video

Submitted By:
Apeksha Bodade BT15CSE020
Zeel Satasiya BT15CSE098

1.INTRODUCTION

In any driving scenario, lane lines are an essential component of indicating traffic flow and where a vehicle should drive. It's also a good starting point when developing a self-driving car!

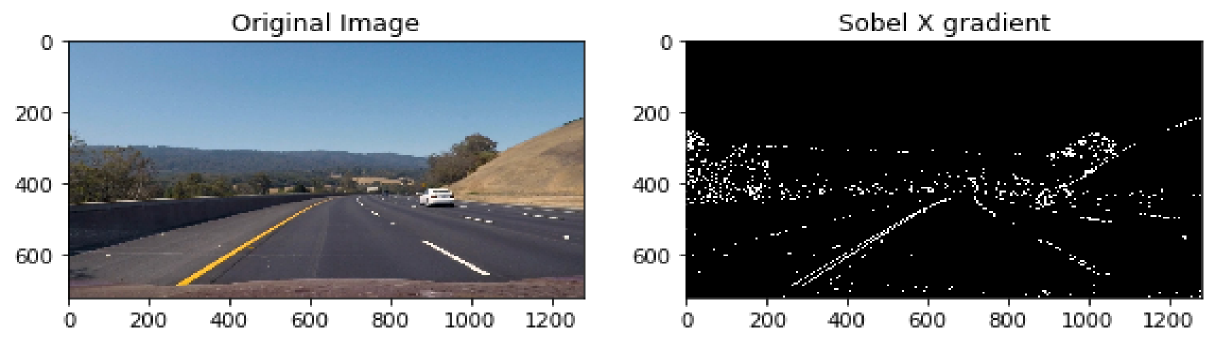
2. CREATION OF THRESHOLDED BINARY IMAGE:

I used a combination of color and gradient thresholds to generate a binary image.

1. Sobel operator for evaluating the gradients.
2. Color Thresholding.
3. Gradient Thresholding

Sobel operator for evaluating the gradients:

Sobel Operator calculates the gradient of every pixel by convolving a kernel on the image.



Color Thresholding:

We used HLS color channel for the purpose of color thresholding.
H - Hue L - Lightness S - Saturation
Lightness and Saturation channel proved out to be quite useful.

To detect yellow lines:

Yellow lines can be easily detected by using the saturation channel.

To detect white lines:

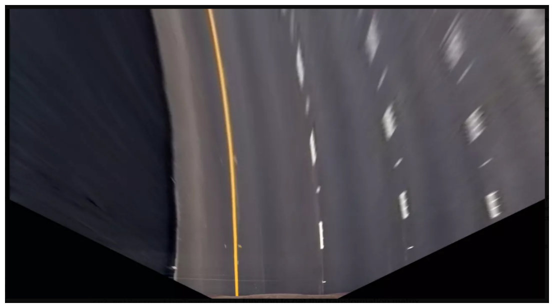
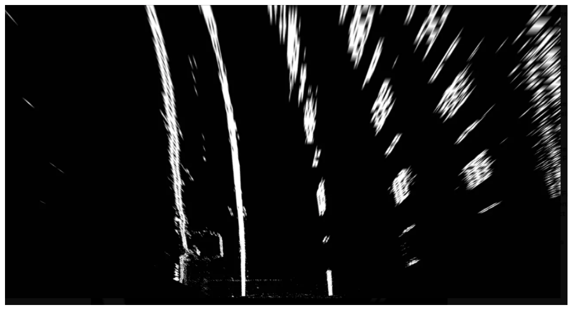
White lines are hard to detect using saturation chanel. But lightness channel is helpful in identifying white lines.

Gradient Thresholding:

Further edges can be detected by setting a threshold on the gradients. As lane lines are more inclined towards vertical than horizontal, thresholding gradients in horizontal direction (X) can detect lane lines. We add the pixels which pass our gradient threshold to a binary matrix representing the pixels in our image.

3.PERSPECTIVE TRANSFORMATION:

Detecting curved lanes in camera space is not very easy. What if we could get a birds eye view of the lanes? That can be done by applying a perspective transformation on the image.

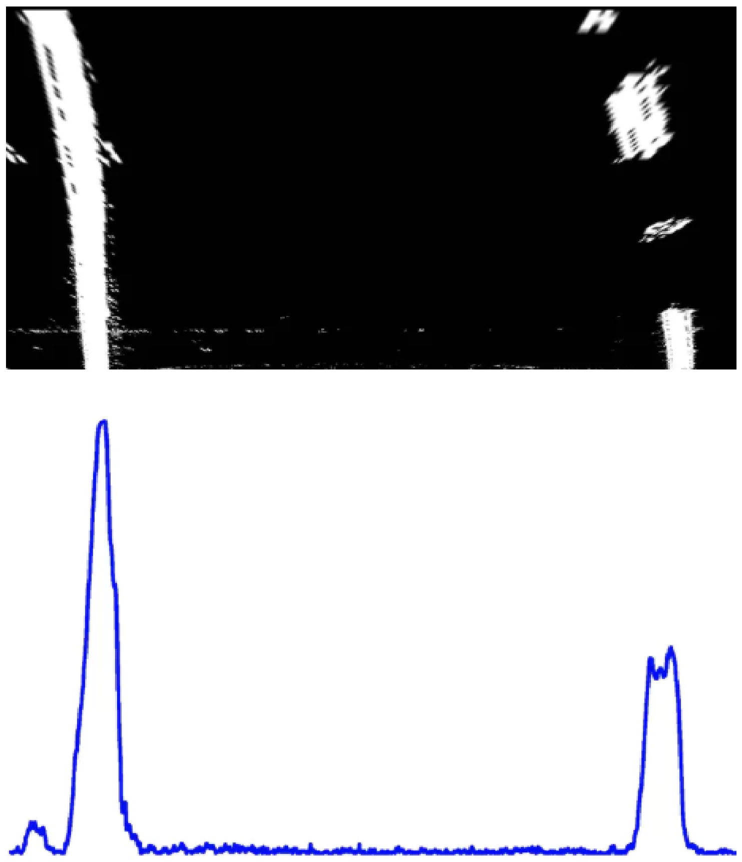



4.IDENTIFYING LANE-LINE PIXELS:

- Histogram Peak Detection
- Using Sliding Window Algorithm to find the lane pixels

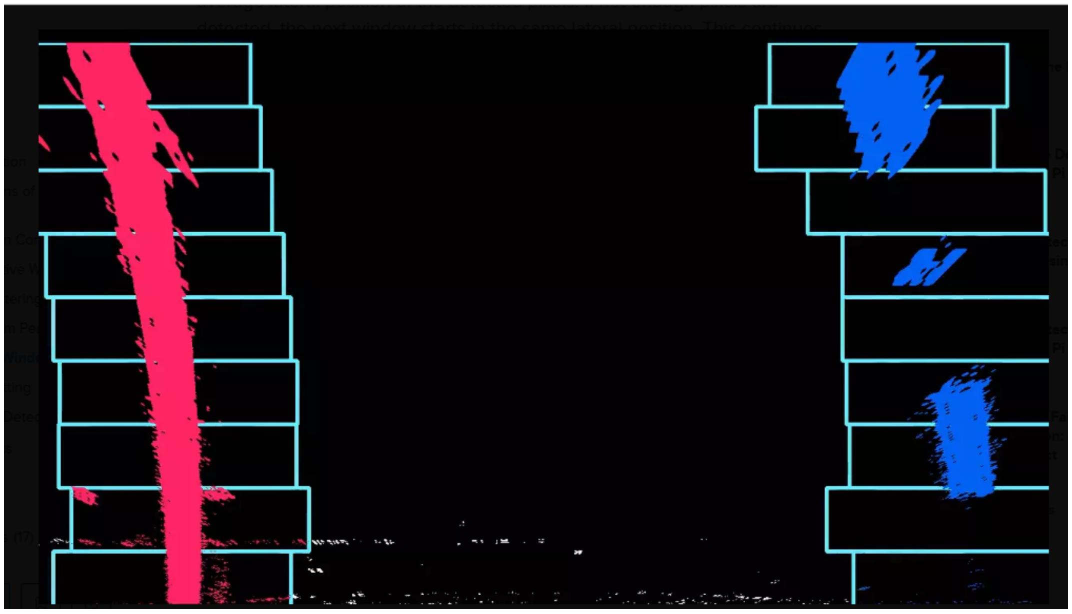
4.1Histogram Peak Detection:

We used the histogram method to find the peaks in the image to find the starting position for the sliding windows to identify the lane.
We'll be getting a histogram of the image with respect to the X axis. Each portion of the histogram below displays how many white pixels are in each column of the image. We then take the highest peaks of each side of the image, one for each lane line.



4.2 Sliding Window Algorithm:

Starting from the initial position(from histogram peak detection), the first window measures how many pixels are located inside the window. If the amount of pixels reaches a certain threshold, it shifts the next window to the average lateral position of the detected pixels. If not enough pixels are detected, the next window starts in the same lateral position. This continues until the windows reach the other edge of the image.The pixels that fall within the windows are given a marker. In the images below, the blue marked pixels represent the right lane, and the red ones represent the left.



5. CALCULATE RADIUS OF CURVATURE AND POSITION OF VEHICLE:

Calculated the mean of left and right lane to find the lane's center coordinates. Assuming the vehicle to be in the middle of the frame, we calculated the difference between the lane center and the vehicle's position(as found out by frame center).

The radius of curvature is calculated using the following formula:

$$K = \frac{\left|y''(x)\right|}{\left[1 + \left(y'(x)\right)^2\right]^{\frac{3}{2}}}.$$

The overall radius of curvature is calculated as the mean of both the left and right radius.

6. Conclusion:

The resulting video will highlight the curved lane area and print the radius of curvature and the offset of the vehicle at every 40 milliseconds.