ENPM673 Project 2

Kumar Sambhav Mahipal, Raghav Agarwal and Vasista Ayyagari

March 2020

1 Problem 1: Improving Poorly Lit Images/video Sequences

A simple technique to improve the poorly lit image was to use gamma correction. However, this introduced a lot of noise and applying gamma correction distorted the color scheme. However, it improved the lighting conditions and the road could be seen better. A better approach was converting the image from RGB color space to YUV where Y is a representation of Luminosity and U and V is that of chrominance (as mentioned in wikipeda). Then, Adaptive Histogram Equalisation was applied to the image. This improved the lighting of the image without distorting the color scheme but also aggravated noise. Please check Figure 1.

2 Problem 2 Lane Detection and Segmentation Algorithm

2.1 Step 1: Preprocessing the input: Undistorting, Denoising, Color Segmentation and Cropping

The video provided in the data sets are divided into frames which are processed individually.The image from the video is undistorted at the beginning of the algorithm to remove either barrel or the fish eye effect. The inputs to perform this function are the image, the camera calibration matrix K and the distortion coefficients.The camera calibration matrix and the distortion coefficients were provided along with the data sets. After undistorting the image, Gaussian blur is applied to remove noise from the image. A range for the color "White" and "Yellow" is defined in the RGB and HLS color space. Using these ranges, the image is masked such that only the pixels within either of the ranges will be white. Finally, the image is cropped to only give us the Region Of Interest (ROI).

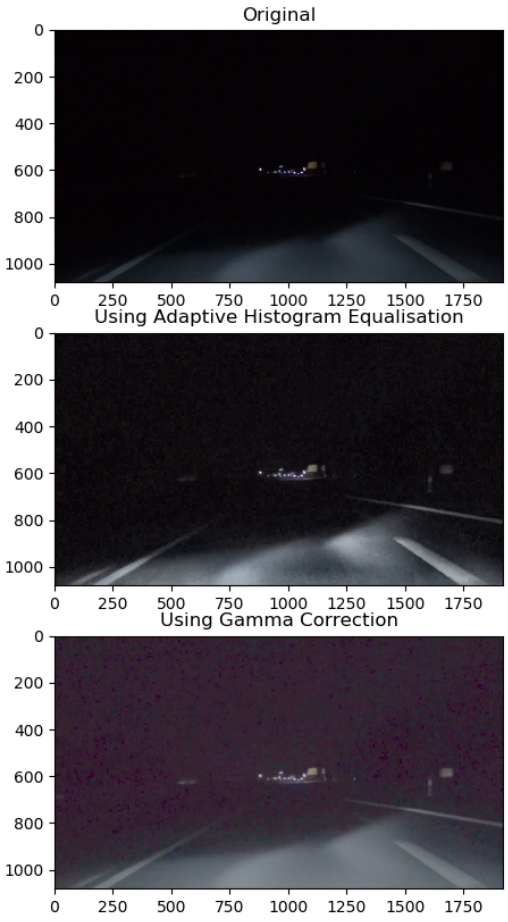

Figure 1: Comparing Gamma Correction to Adaptive Histogram Equalization

2.2 Step 2: Homographic Transformation

After pre-processing, homography matrices from camera to world view and vice versa are computed. The lanes on the road are assumed to be parallel to each other and almost always 'vertical', if we view the image straight from the top. The actual camera does not look at the road 'from the top', but it always looks at the road at the same angle.

Hence computing homography and taking the perspective view will obtain birds eye view of the road and the lanes which were converging in the original image become parallel. The four points for homography in the camera and world view are selected in such a way that two points are taken on each side of the primary lane.

Once Homography is completed, Warp Perspective is performed on the image which takes the homography matrix as the input and displays the top view of the image as the output. We decompose the Matrix A using the Singular Value Decomposition Method:

$$[U, S, V] = svd(A) \tag{1}$$

2.3 Step 3: Edge Detection : Sobel Operator

The warped binary image (in the world view) is taken and sobel operator is applied. We also tried with cv2.canny function for edge detection but the sobel operator and simple thresholding gave decent results.

2.4 Step 4: Finding line equations for left and right lanes and for radius of curvature

Polyfit function takes in the image as the input. the function then divides the image into left and right parts and stores them in two separate variables. The coordinates for nonzero pixels (which represent the lanes) are computed .These points are used to fit a second degree polynomial.The radius of curvature is computed as the mean of the radii of curvature for each lane equation which is obtained from equation 2

$$R = \left| \frac{(1 + y'^2)^{\frac{3}{2}}}{y''} \right| \tag{2}$$

The polyfit function then returns three equations equation for left and right lanes and equation for radius of curvature.

2.5 Step 5: Overlaying the detected lane on to the original image

The overlay-lanes function takes in five parameters, the original image, image showing edges, inverse homography and the polyfit equations for left and right

Figure 2: Lane Segmentation for Dataset 1

Figure 3: Lane Segmentation for Dataset 2

lanes. The cv2.fillPoly function draws a filled contour representing the lane onto an blank image in the world coordinates which is unwarped using the inverse homography matrix. The final result will be a weighted sum of the original image and the unwarped binary image.

2.6 Will the pipeline generalize to other similar videos?

The bottleneck to this algorithm is the color- based detection of the lanes. Under different lighting conditions, especially in shadowy regions, the detection does not work well. Moreover, the presented pipeline will not generalise to extremely curved roads as the curve fitting is done by simply partitioning into left and right side of the image. However, the pipeline will generalize well for well lit and slightly curved or straight roads.

2.7 Our understanding of homography and how it is used

When we are dealing with the perception of objects in real world through the computer/camera, we are mapping the points from one coordinate system to another coordinate system. Any point/object in 3D world, when projected onto the camera plane, gets mapped to the 2D image plane. Therefore, we can consider this as a basic example of projective geometry.

$$\begin{bmatrix} x_c \\ y_c \\ w_c \end{bmatrix} = H \begin{bmatrix} X_w \\ Y_w \\ 1 \end{bmatrix} \tag{3}$$

Using these homography, we transformed the point from one plane(world) to the other(image). Similarly, we can have homography between two image planes as well. In this project for lane detection, we have one plane as the original frame as extracted from the video. In order to have the top-view(birds-eye-view) of that lanes, we need to compute homography between that plane and the desired plane (the dimensions of the new image plane in which we want the view). For this, we simply use the inbuilt function from OpenCV.
i) cv2.getPerspectiveTransform
ii) cv2.findHomography
Both these functions yield desired homography transformation between two planes

2.8 How do hough lines work?

Hough Lines uses a parametric model of a line(here, d and θ). Each point in the hough space is a line in the Cartesian space. Each point in the Cartesian space is a sinusoid in the hough space. For n points in the Cartesian space, there will be n sinusoids. An intersection between 2 sinusoids will give a single point in the hough space which will be the line joining the two points in the Cartesian space that initially generated the two intersecting sinusoids. For a binary image, all pixels with a high value will correspond to a point in cartesian space and a sinusoid in the hough space. The hough lines are generated via a voting scheme. The hough space is discretized into bins where votes will be cast by each point in the cartesian space. Specifically, for each point, and each value of θ, a corresponding d is computed and the nearest bin to (d, θ) will have an additional vote. The bin with the maximum votes (or that has a local maxima) will be considered as the fitted line in the Cartesian space.

Using multiple local maximas, multiples lines can be detected in one shot. The computational cost increases with the number of parameters describing a curve but the voting scheme is accurate.

2.9 Challenges faced

1. Choosing the 4 points for homography transformation: We needed to choose 4 points that were robust enough for to display a significant length of the road without going out of bounds from the world view

2. Choosing the color range: manually tweaking was required for the color detection to work robustly

3. Outliers edges obtained after applying the sobel operator affected the parameters of the fitted curves

4. Without a reference distance for the road, we were unable to transform the fitted curve into actual measurements and compute the actual radius of curvature of the road. Hence, we proceeded to compute the curvature of the function only.