# Lane detection project

**Aviv Weidenfeld 315589937**
**Koren Knafo 318202371**

**Frame Processing:**

Initially, we assumed the lane lines would be in specific regions of the image and identified these areas to remove unnecessary noise. We then focused our detection efforts on these predefined sections to improve accuracy. After identifying the predefined regions, we further assumed that there would be two distinct lane lines, each positioned on different sides of the cropped image. This assumption guided our lane detection strategy to identify and annotate the lanes accurately.

**Finding the edges:**

We utilized the Canny edge detection algorithm from the OpenCV library to detect edges in the image. Before applying Canny, we preprocessed the image by applying Gaussian blur to reduce noise. Subsequently, we assumed that the lane lines would appear white, so we applied binary thresholding to isolate the white regions. However, we observed significant noise in the corners of the image. To address this, we created a mask to crop out these noisy corners, improving the accuracy of our lane detection.

**The lane detection:**

We implemented the functions find_lines_with_hough_transform and ransac_line_fit as fundamental components of our lane detection system, aimed at precise identification and representation of lane markings within images.

In find_lines_with_hough_transform, we applied the Hough Transform algorithm to detect lines within edge-detected images. These lines were then categorized into left, right, and midlines based on their slopes and positions within the frame. Our approach assumed that lane lines were generally straight, inclined within specific ranges, and positioned in predefined regions of the frame. By segmenting lines based on slope and position characteristics, this function effectively pinpointed potential lane markings.

On the other hand, in ransac_line_fit, we leveraged the RANSAC algorithm to robustly fit lines to sets of points, which might be influenced by outliers. This involved iteratively sampling pairs of points, fitting lines using linear regression, and identifying inliers based on a predetermined threshold. Our strategy addressed the assumption that points exhibited approximate collinearity but could be disrupted by outliers stemming from noise or other factors. Through robust line fitting and outlier exclusion, ransac_line_fit accurately determined the parameters of lane lines, thereby enhancing the overall precision of our lane detection system.

Together, these functions constituted a comprehensive pipeline for detecting and fitting lines to portray lane markings, guided by assumptions about line characteristics and positions. Their integration was crucial in achieving the primary objective of accurately identifying lanes in images, a critical requirement for applications such as autonomous driving or lane departure warning systems.

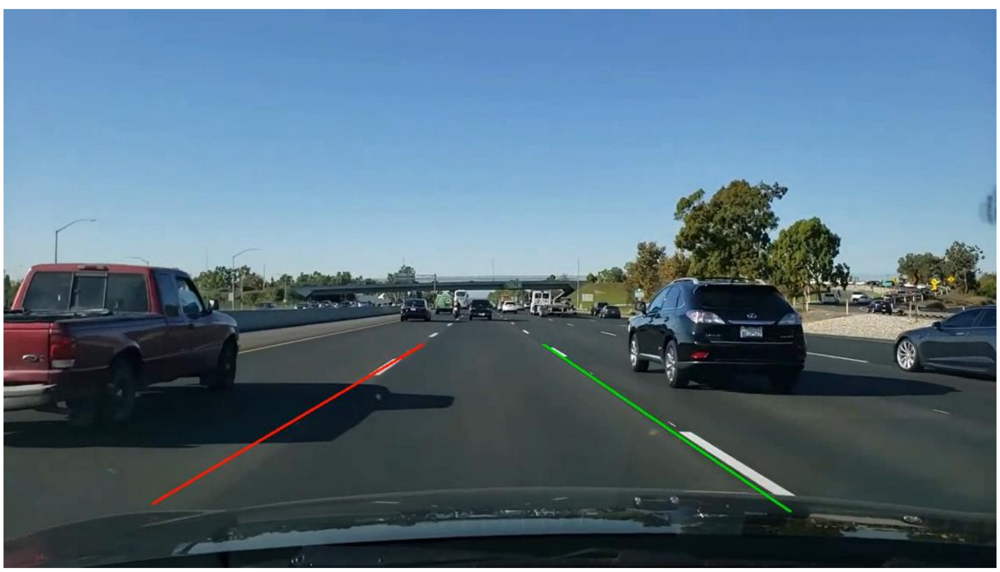
**<u>The night videos:</u>**

During the night video analysis, we adapted our lane detection algorithm to accommodate low-light conditions by adjusting the binary threshold accordingly. This adjustment ensured that lane markings remained discernible even in reduced visibility. Additionally, we incorporated a top threshold to mitigate false detections caused by strong lights, which were erroneously identified as lines. By fine-tuning these parameters to suit nighttime conditions, we aimed to enhance the accuracy and reliability of our lane detection system

**<u>The curve fitting:</u>**

In the curve fitting phase, our objective was to accurately fit a polynomial curve to the detected edges, striving to find the optimal fit. However, determining the separation between the lane lines posed a significant challenge, requiring careful consideration and experimentation to identify the most suitable separation. Despite the difficulty in precisely defining this separation, we made concerted efforts to find the best possible separation. We assumed the left line is yellow and the right line is white to aid in distinguishing between them. Additionally, this helps to remove noises.

**<u>Frame analsing:</u>**



In this image, we showcase the approach we adopted. We altered the image to remove the corners because the vehicle's wheels were being mistaken for lines. Additionally, a black and white threshold filter was applied to eliminate the car's shadow, which was also falsely detected as a line. We assumed that lane lines would be white and that corners would not constitute actual lane markings. The image reflects these adjustments.

## **Summary:**

We successfully achieved accurate detection and natural representation of lane lines, effectively capturing their movement. Additionally, we implemented robust noise reduction techniques, ensuring that only relevant edges remained prominent. However, challenges arose during lane switches, particularly in accurately identifying the mid lane. To address this limitation, we propose implementing perspective transformation to enhance clarity in the mid-region, potentially improving precision and facilitating better detection of the mid lane. This enhancement aims to refine our lane detection system, particularly in dynamic scenarios such as lane changes, ultimately enhancing its overall accuracy and effectiveness.