Table Of Contents

[Lane Detection 2](#_Toc187617474)

[Overview 2](#_Toc187617475)

[Key Functionalities 3](#_Toc187617476)

[Camera Calibration 3](#_Toc187617477)

[Image preprocessing 3](#_Toc187617478)

[Edge Detection 3](#_Toc187617479)

[Perspective Warp 3](#_Toc187617480)

[Lane Separator 4](#_Toc187617481)

[Curve Fit 4](#_Toc187617482)

[Temporal Analysis 4](#_Toc187617483)

[Lane Change Detection 4](#_Toc187617484)

[Results 5](#_Toc187617485)

Table Of Figures

[Figure 1: Camera Calibration - checkboard 5](#_Toc187617486)

[Figure 2: Left - distorted lanes, Right - undistorted lanes 6](#_Toc187617487)

[Figure 3: Left - Edges after Image Processing and Canny, Right – Edges after Perspective Warp and ROI Cropping 6](#_Toc187617488)

[Figure 4: Temporal visualization to determine the correct threshold for lane change detection 7](#_Toc187617489)

[Figure 5: Temporal Median Grayscale Intensity to understand the range of shadowed regions for shadow removal 8](#_Toc187617490)

[Figure 6: Region of Interest Visualization for debug 9](#_Toc187617491)

# Lane Detection

## Overview

In this task, we implement a comprehensive pipeline for lane detection using computer vision techniques. The system is designed to operate under both day and night conditions, handling various challenges such as shadows, lane curvature, and lane changes. The pipeline utilizes Python and OpenCV, integrating several preprocessing and image processing steps to achieve robust lane detection.

We implement the following process:

1. **Camera Calibration**: Undistort the original image, correcting for lens distortion and enhancing accuracy in lane detection.
2. **Preprocessing and Edge Detection**: Adjust image contrast using histogram equalization techniques like CLAHE to enhance visibility under varying conditions. Convert images to grayscale and use adaptive thresholding, Gaussian blur, and the Canny edge detector to detect lane edges.
3. **Perspective Transformation**: Apply a perspective warp to create a bird's-eye view of the road, simplifying the detection of lane lines and curves.
4. **Lane Detection Pipeline**: Detect left and right lane boundaries using the sliding window technique. Fit polynomial curves to the detected lane points to model lane boundaries.
5. **Temporal Analysis**: Compute a temporal median from multiple frames to reduce noise and adapt to transient changes in brightness. Detect lane changes by analyzing the deviation in fitted lane curves across frames. Determine threshold for lane change using temporal analysis.
6. **Video Processing**: Process video frames to overlay detected lanes and lane change indicators and save the results to a video file.

## Key Functionalities

### Camera Calibration

Camera lenses distort incoming light to focus it onto the camera sensor, often introducing minor inaccuracies in light distortion. This can lead to measurement errors in computer vision tasks. Camera calibration mitigates this by generating a distortion model that corrects for lens distortions using a known reference object, such as an asymmetric checkerboard *Figure 1: Camera Calibration - checkboard*.

### Image preprocessing

Conversion to the LAB color space and enhancement of the L-channel using CLAHE to improve visibility of lanes. In day time shadow removal is applied before CLAHE to increase edges contrast. Masking of white lane pixels using HSV color space. Adaptive thresholding to account for different lighting conditions. Gaussian blurring to reduce noise. *Figure 2: Left - distorted lanes, Right - undistorted lanes*

### Edge Detection

Canny edge detection method to identify edges in the blurred image. The intensity thresholds for edge detection are dynamically calculated based on the median pixel intensity of the image. This makes the algorithm adaptive to varying lighting conditions, which is crucial for robustness in day and night scenarios.

### Perspective Warp

Detecting curved lanes directly in the camera’s perspective can be challenging. By applying a perspective transformation, we obtain a bird’s-eye view of the lanes, simplifying the task. Assuming the lane exists on a flat 2D surface, a polynomial curve can then be fitted to accurately represent the lane boundaries. This module handles both perspective warping and ROI extraction. *Figure 3: Left - Edges after Image Processing and Canny, Right – Edges after Perspective Warp and ROI Cropping*

*Figure 6: Region of Interest Visualization for debug*

### Lane Separator

After acquiring the birds eye view of the ROI of the edges we use the sliding window algorithm to differentiate between the left and right lane boundaries so that we can fit two different curves representing the lane boundaries.

The algorithm itself is as follows: Starting from the initial position, 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 insufficient pixels are detected, the next window starts in the same lateral position. This continues until the windows reach the other edge of the image.

### Curve Fit

After acquiring the lanes, this part is straight forward, we apply polynomial regression to the left and right lane pixels individually using of degree 2 following the logic that when the equation becomes a good estimation of a straight line and when we get a curve.

### Temporal Analysis

The temporal median is calculated over a fixed history of grayscale frames to estimate overall scene brightness and determine day or night conditions.

Temporal median filtering smooths out noise across frames, making the system more robust to transient artifacts.

### Lane Change Detection

Lane change detection is implemented by monitoring significant deviations in the fitted curves over time, with thresholds set for flagging changes.

*Figure 4: Temporal visualization to determine the correct threshold for lane change detection*

*Figure 5: Temporal Median Grayscale Intensity to understand the range of shadowed regions for shadow removal*

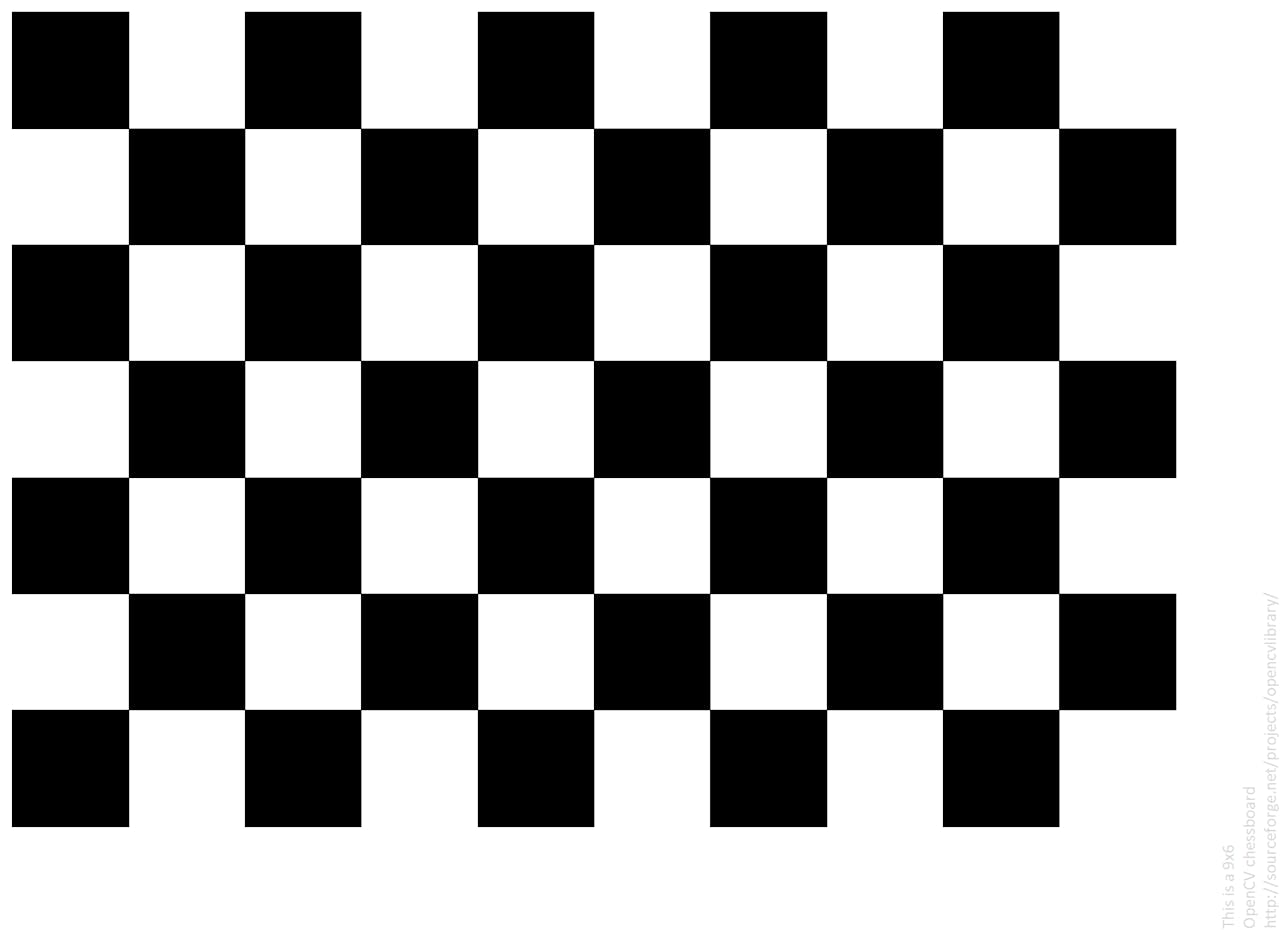
Results

Figure 1: Camera Calibration - checkboard

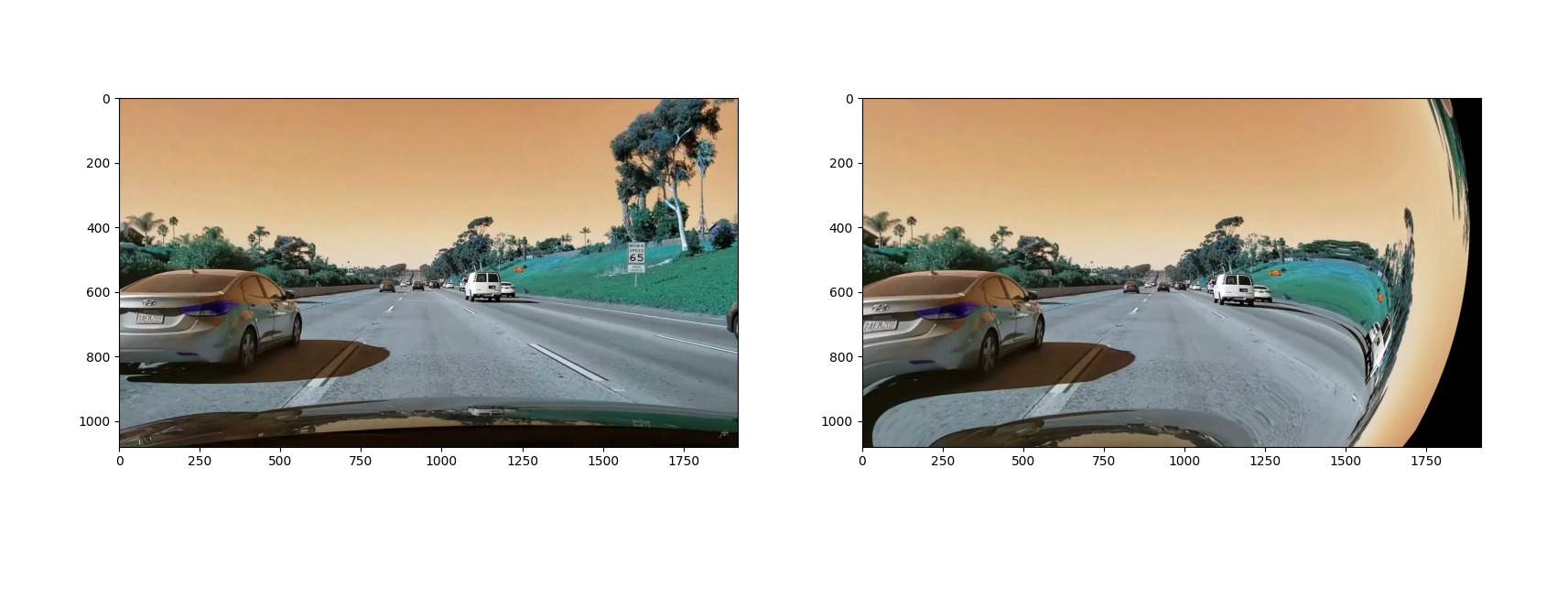


Figure 2: Left - distorted lanes, Right - undistorted lanes

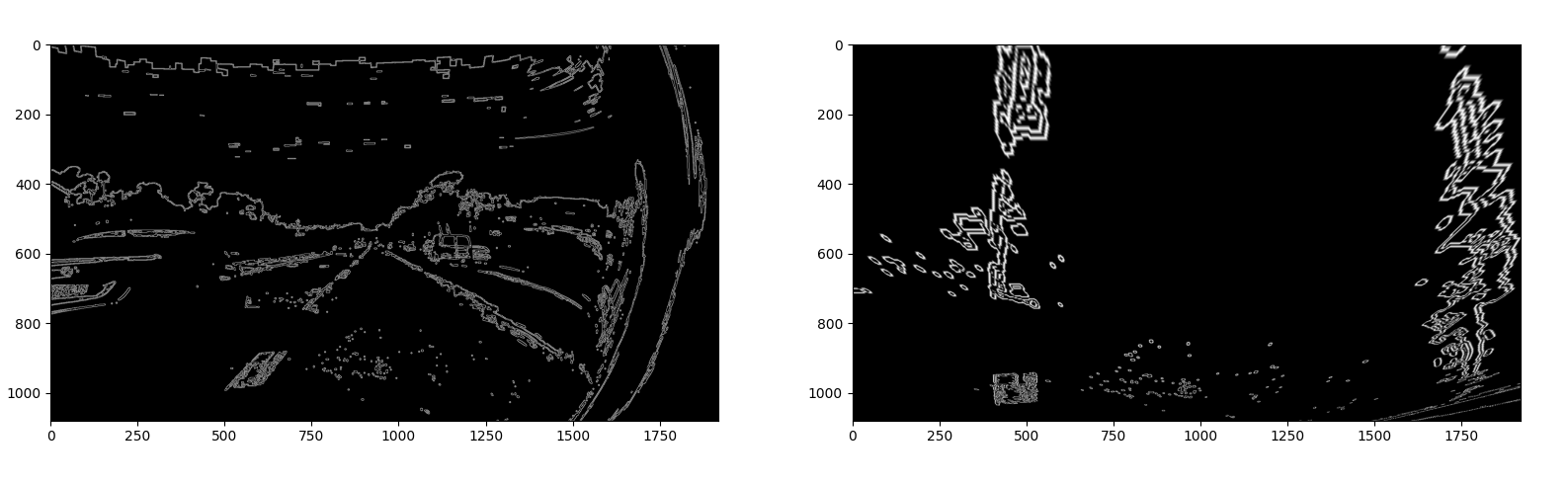


Figure 3: Left - Edges after Image Processing and Canny, Right – Edges after Perspective Warp and ROI Cropping

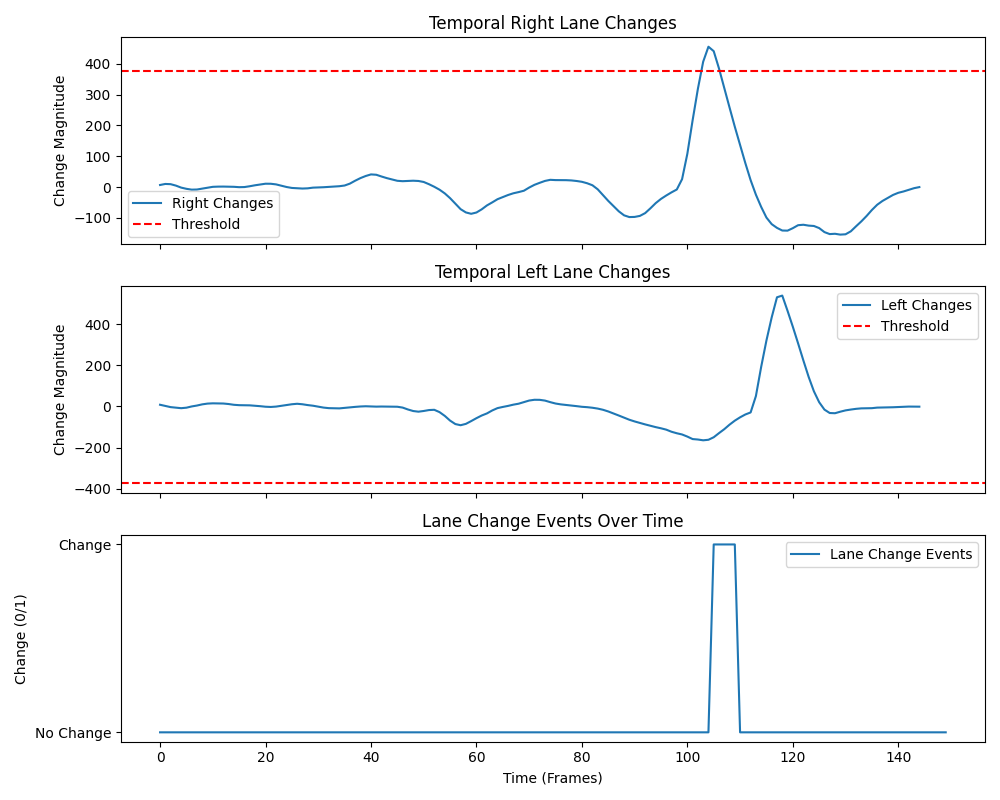


Figure 4: Temporal visualization to determine the correct threshold for lane change detection

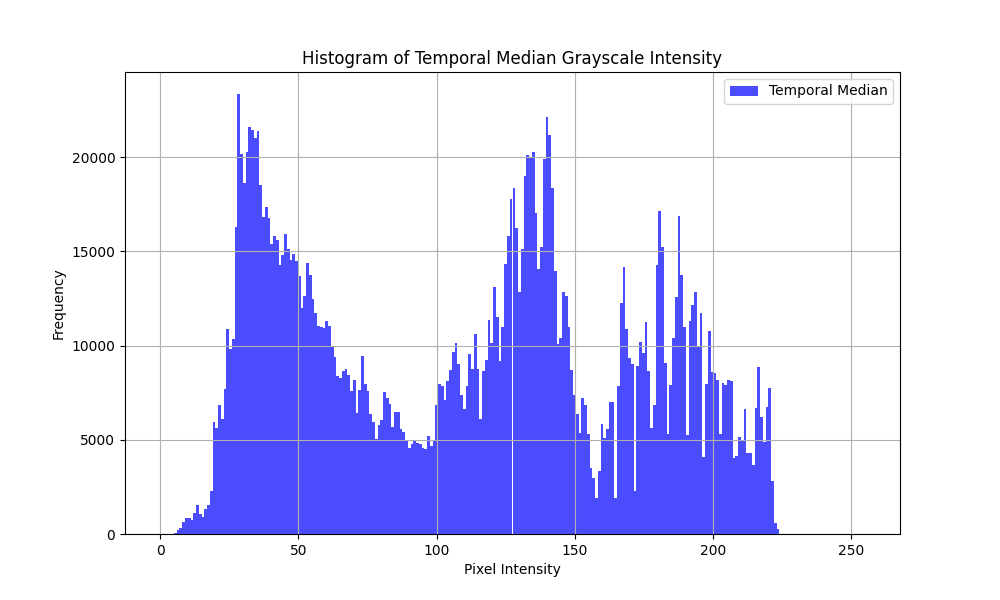


Figure 5: Temporal Median Grayscale Intensity to understand the range of shadowed regions for shadow removal

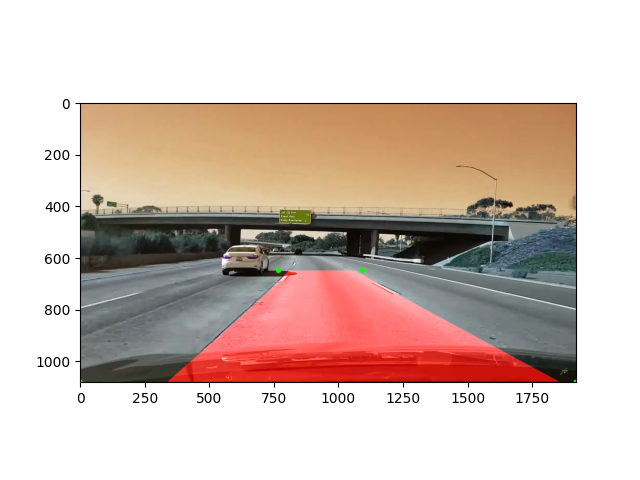


Figure 6: Region of Interest Visualization for debug