**Lane Detection (OpenCV)**

# **Using Hough Transform**

* 1. Capturing the Image using **VideoCapture()**.
  2. Applying blur to the Image.
  3. Converting the Image to grayscale.
  4. Masking the image to select white lines.
  5. Using **canny** to detect edges in Image.
  6. Using **Hough Transform** to get the Lane Lines.

Demo Video:

<https://drive.google.com/file/d/1tCZQmn8JnYfudoezUDOOMPIZvwRGaSQf/view?usp=sharing>

# **Using Sliding Window Search**

* 1. Calibrating code to camera for **pixels to meter conversion**.
  2. **Pre-Processing the Image:**
     1. Converting the image to **HLS**(Hue, Lightness and Saturation).
     2. Extracting the white lines from the Image.
     3. Converting the Image to Grayscale and applying Blur.
     4. Using **Canny** to detect edges.
  3. **Converting the Image into Top-Down View**:
     1. Changing the **source and destination points** for conversion.
     2. Using **Perspective Warp** to convert.
  4. Plotting **Histogram to detect the base** of Lane Lines:
     1. **Summing up all the pixels vertically** in the **bottom half**.
     2. Finding the **most intense pixels** in the **left and right halves of the image seperately**.
  5. Applying **Slide Window Search** to find the **estimated location** of lane lines:
     1. The Image gets **divided into a number of windows** (we used 9).
     2. Finding the rough location of the lane in each window.
     3. Fitting an estimated curve representing the lane lines for viewing.
  6. Applying **General Search** to further pinpoint the location of the lane inside each window.
  7. Measure the lane **Curvature, Off-set from the center, and off-set direction**.

Demo Video: <https://drive.google.com/file/d/1zdBfotrWBPCz6kdWUTVV-lYPEYlOCMU9/view?usp=sharing>

**NOTE: The code outputs the following: Deviation,**

**Deviation Direction,**

**Curve Radius,**

**Curve Direction.**