**Project Synopsis Report: Road Lane Line Detection – Computer Vision Project in Python**

**1. Introduction**

Lane line detection is essential for self-driving cars and computer vision applications. It ensures vehicles stay within their designated lanes, reducing the risk of accidents. This project aims to detect lane lines in real-time using Python and OpenCV.

**2. Objectives**

* Develop a machine learning model to detect lane lines on roads.
* Utilize computer vision techniques for real-time lane detection.
* Apply frame masking and Hough Line Transformation to identify lane markings.

**3. System Requirements**

**Hardware Requirements:**

* Processor: Intel Core i5 or higher
* RAM: 8GB or more
* Storage: 50GB free space
* OS: Windows 10 / macOS / Linux

**Software Requirements:**

* Python 3.6 or higher
* Jupyter Lab
* Libraries: numpy, opencv-python

**4. Key Concepts and Tools**

**Lane Line Detection:** The process of identifying lane markings on roads to guide autonomous vehicles, ensuring they stay within their lanes and avoid accidents.

**Frame Masking:** A technique to isolate relevant parts of an image by setting unnecessary pixels to zero, simplifying the detection process.

**Hough Line Transformation:** A method to detect geometric shapes such as lines, circles, and rectangles in an image by transforming image pixels into a parameter space.

**5. Project Steps**

**Step 1: Setup Environment**

1. Install Python and Jupyter Lab.
2. Install required libraries:

bash

1. pip install numpy opencv-python

**Step 2: Load and Preprocess Data**

1. Capture or load a video frame using OpenCV.
2. Convert the frame to grayscale for easier processing.

**Step 3: Frame Masking**

1. Define a region of interest (ROI) to focus on lane lines.
2. Apply a mask to the ROI to isolate the lane markings.

**Step 4: Hough Line Transformation**

1. Use the Canny edge detector to find edges in the frame.
2. Apply the Hough Line Transformation to detect lane lines.

**Step 5: Display Results**

1. Draw the detected lane lines on the original frame.
2. Display the result in real-time using OpenCV.

**6. Code Implementation**

**Loading and Preprocessing Data:**

python

import cv2

import numpy as np

# Load video

cap = cv2.VideoCapture('road\_video.mp4')

def preprocess\_frame(frame):

# Convert to grayscale

gray = cv2.cvtColor(frame, cv2.COLOR\_BGR2GRAY)

# Apply Gaussian blur

blur = cv2.GaussianBlur(gray, (5, 5), 0)

# Edge detection

edges = cv2.Canny(blur, 50, 150)

return edges

**Frame Masking:**

python

def region\_of\_interest(edges):

height, width = edges.shape

mask = np.zeros\_like(edges)

# Define a polygon to mask

polygon = np.array([[

(0, height),

(width, height),

(width, height//2),

(0, height//2),

]], np.int32)

cv2.fillPoly(mask, polygon, 255)

masked\_edges = cv2.bitwise\_and(edges, mask)

return masked\_edges

**Hough Line Transformation:**

python

def detect\_lines(masked\_edges):

# Detect lines using Hough Transformation

lines = cv2.HoughLinesP(masked\_edges, 1, np.pi/180, 50, minLineLength=100, maxLineGap=50)

return lines

def display\_lines(frame, lines):

line\_image = np.zeros\_like(frame)

if lines is not None:

for line in lines:

x1, y1, x2, y2 = line[0]

cv2.line(line\_image, (x1, y1), (x2, y2), (255, 0, 0), 10)

combined\_image = cv2.addWeighted(frame, 0.8, line\_image, 1, 1)

return combined\_image

**Real-Time Lane Line Detection:**

python

while cap.isOpened():

ret, frame = cap.read()

if not ret:

break

edges = preprocess\_frame(frame)

masked\_edges = region\_of\_interest(edges)

lines = detect\_lines(masked\_edges)

result = display\_lines(frame, lines)

cv2.imshow('Lane Line Detection', result)

if cv2.waitKey(1) & 0xFF == ord('q'):

break

cap.release()

cv2.destroyAllWindows()

**7. Conclusion**

This project demonstrates the use of computer vision techniques to detect road lane lines in real-time. By implementing frame masking and Hough Line Transformation, the project ensures accurate detection of lane markings, which is crucial for the safe operation of autonomous vehicles. Completing this project enhances your skills in image processing and real-time video analysis using Python and OpenCV.