# **ROAD LANE DETECTION**

(CORIZO PROJECT)

## **LIBRARIES USED:**

OpenCV - pip install opency-contrib-python Numpy - pip install numpy

Files Used: Road.png



# IN THIS FILE I WILL DETECT ROAD LINE

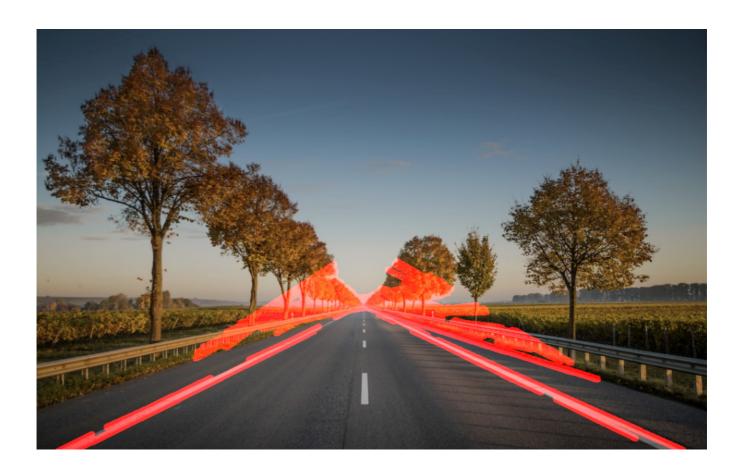
(Without video capturing)

# **Code-1( without live video capturing)**

#### **INPUT:**

```
1 import cv2
 2 import numpy as np
 3 img = cv2.imread('road.png')
 4 gray = cv2.cvtColor(img, cv2.COLOR_BGR2GRAY)
 5 blur = cv2.GaussianBlur(gray, (5, 5), 0)
 6 \text{ edges} = \text{cv2.Canny(blur, 50, 150)}
 7 mask = np.zeros_like(edges)
 8 height, width = mask.shape
 9 polygon = np.array([[
       (0, height),
10
11
       (width, height),
12
       (width // 2, height // 2)
13 ]])
14 cv2.fillPoly(mask, polygon, 255)
15 masked_edges = cv2.bitwise_and(edges, mask)
17 lines = cv2.HoughLinesP(masked_edges, rho=6, theta=np.pi/60, threshold=160,
   lines=np.array([]), minLineLength=40, maxLineGap=25)
18
19 line_img = np.zeros_like(img)
20 for line in lines:
21
       x1, y1, x2, y2 = line[0]
22
       cv2.line(line_img, (x1, y1), (x2, y2), (0, 0, 255), 10)
23
24 result = cv2.addWeighted(img, 0.8, line_img, 1.0, 0.0)
26 cv2.imshow('Result', result)
27 cv2.waitKey(0)
28
29
30
31
```

#### **OUTPUT:**



# **Screenshot of the code:**

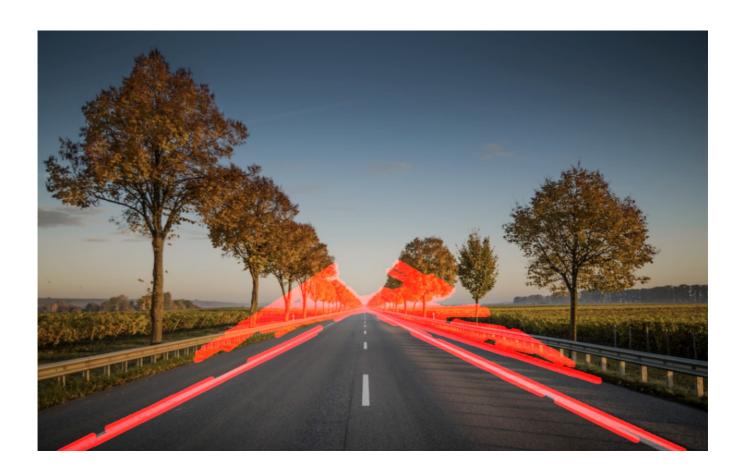
```
import cv2
import numpy as np
img = cv2.imread('road.png')
gray = cv2.cvtColor(img, cv2.COLOR_BGR2GRAY)
blur = cv2.GaussianBlur(gray, (5, 5), 0)
edges = cv2.Canny(blur, 50, 150)
mask = np.zeros_like(edges)
basek = np.zeros_like(edges)
         height, width = mask.shape
         polygon = np.array([[
               (0, height),
(width, height),
(width // 2, height // 2)
10
13
        cv2.fillPoly(mask, polygon, 255)
masked_edges = cv2.bitwise_and(edges, mask)
14
15
16
17
         lines = cv2.HoughLinesP(masked_edges, rho=6, theta=np.pi/60, threshold=160, lines=np.array([]), minLineLength=40, maxLineGap=25)
18
         line_img = np.zeros_like(img)
for line in lines:
19
20
21
22
23
               x1, y1, x2, y2 = line[0]
cv2.line(line_img, (x1, y1), (x2, y2), (0, 0, 255), 10)
24
         result = cv2.addWeighted(img, 0.8, line_img, 1.0, 0.0)
25
26
27
         cv2.imshow('Result', result)
         cv2.waitKey(0)
28
```

# **Code-2( with live video capturing)**

#### **INPUT:**

```
1 import cv2
 2 import numpy as np
 4 cap=cv2.VideoCapture(0)
 5 cap.set(3,640)
 6 cap.set(4,480)
 8 ret,img = cap.read()
gray = cv2.cvtColor(img, cv2.COLOR_BGR2GRAY)
10 blur = cv2.GaussianBlur(gray, (5, 5), 0)
11 edges = cv2.Canny(blur, 50, 150)
12 mask = np.zeros_like(edges)
13 height, width = mask.shape
14 polygon = np.array([[
       (0, height),
15
16
       (width, height),
17
       (width // 2, height // 2)
18]])
19 cv2.fillPoly(mask, polygon, 255)
20 masked_edges = cv2.bitwise_and(edges, mask)
22 lines = cv2.HoughLinesP(masked_edges, rho=6, theta=np.pi/60, threshold=160,
   lines=np.array([]), minLineLength=40, maxLineGap=25)
24 line_img = np.zeros_like(img)
25 for line in lines:
26
       x1, y1, x2, y2 = line[0]
27
       cv2.line(line_img, (x1, y1), (x2, y2), (0, 0, 255), 10)
28
29 result = cv2.addWeighted(img, 0.8, line_img, 1.0, 0.0)
31 cv2.imshow('Result', result)
32 cv2.waitKey(1)
33
```

#### **OUTPUT:**



## Screenshot of the code:

```
import numpy as np
        cap=cv2.VideoCapture(0)
        cap.set(3,640)
        cap.set(4,480)
       ret,img = cap.read()
gray = cv2.cvtColor(img, cv2.COLOR_BGR2GRAY)
       gray = cv2.cvtcotor(img, cv2.CoLoR_BGR2Gf
blur = cv2.GaussianBlur(gray, (5, 5), 0)
edges = cv2.Canny(blur, 50, 150)
mask = np.zeros_like(edges)
height, width = mask.shape
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27
28
29
30
        polygon = np.array([[
              (0, height),
              (width, height),
(width // 2, height // 2)
        cv2.fillPoly(mask, polygon, 255)
masked_edges = cv2.bitwise_and(edges, mask)
        lines = cv2.HoughLinesP(masked_edges, rho=6, theta=np.pi/60, threshold=160, lines=np.array([]), minLineLength=40, maxLineGap=25)
        line_img = np.zeros_like(img)
        for line in lines:
              x1, y1, x2, y2 = line[0]
cv2.line(line_img, (x1, y1), (x2, y2), (0, 0, 255), 10)
        result = cv2.addWeighted(img, 0.8, line_img, 1.0, 0.0)
31
        cv2.imshow('Result', result)
        cv2.waitKey(1)
```

# THANK YOU!

**SUBMITTED BY:** 

Ananya Bisht