

ROAD LANE DETECTION

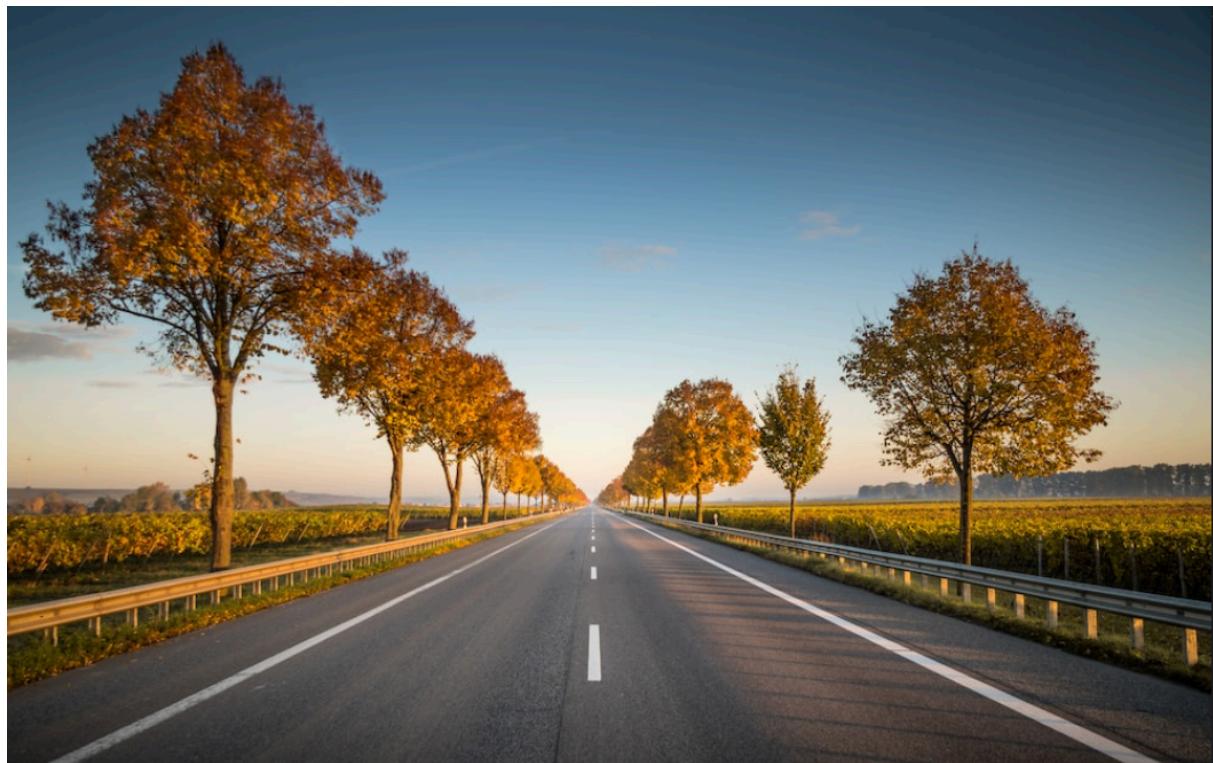
(CORIZO PROJECT)

LIBRARIES USED:

OpenCV - pip install opencv-contrib-python

Numpy - pip install numpy

Files Used: Road.png



IN THIS FILE I WILL DETECT ROAD LINE

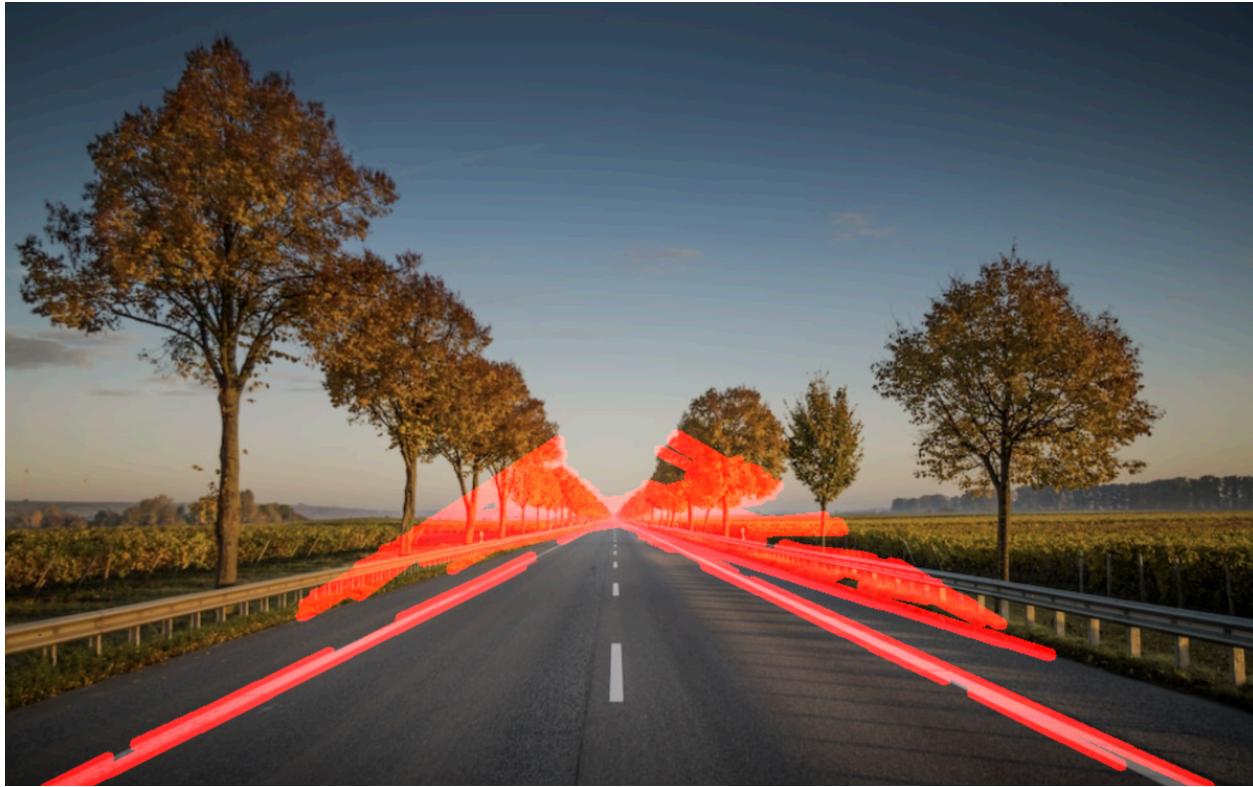
(Without video capturing)

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 edges = cv2.Canny(blur, 50, 150)
7 mask = np.zeros_like(edges)
8 height, width = mask.shape
9 polygon = np.array([[
10     (0, height),
11     (width, height),
12     (width // 2, height // 2)
13 ]])
14 cv2.fillPoly(mask, polygon, 255)
15 masked_edges = cv2.bitwise_and(edges, mask)
16
17 lines = cv2.HoughLinesP(masked_edges, rho=6, theta=np.pi/60, threshold=160,
18                         lines=np.array([]), minLineLength=40, maxLineGap=25)
19
20 line_img = np.zeros_like(img)
21 for line in lines:
22     x1, y1, x2, y2 = line[0]
23     cv2.line(line_img, (x1, y1), (x2, y2), (0, 0, 255), 10)
24
25 result = cv2.addWeighted(img, 0.8, line_img, 1.0, 0.0)
26
27 cv2.imshow('Result', result)
28
29
30
31
```

OUTPUT:



Screenshot of the code:

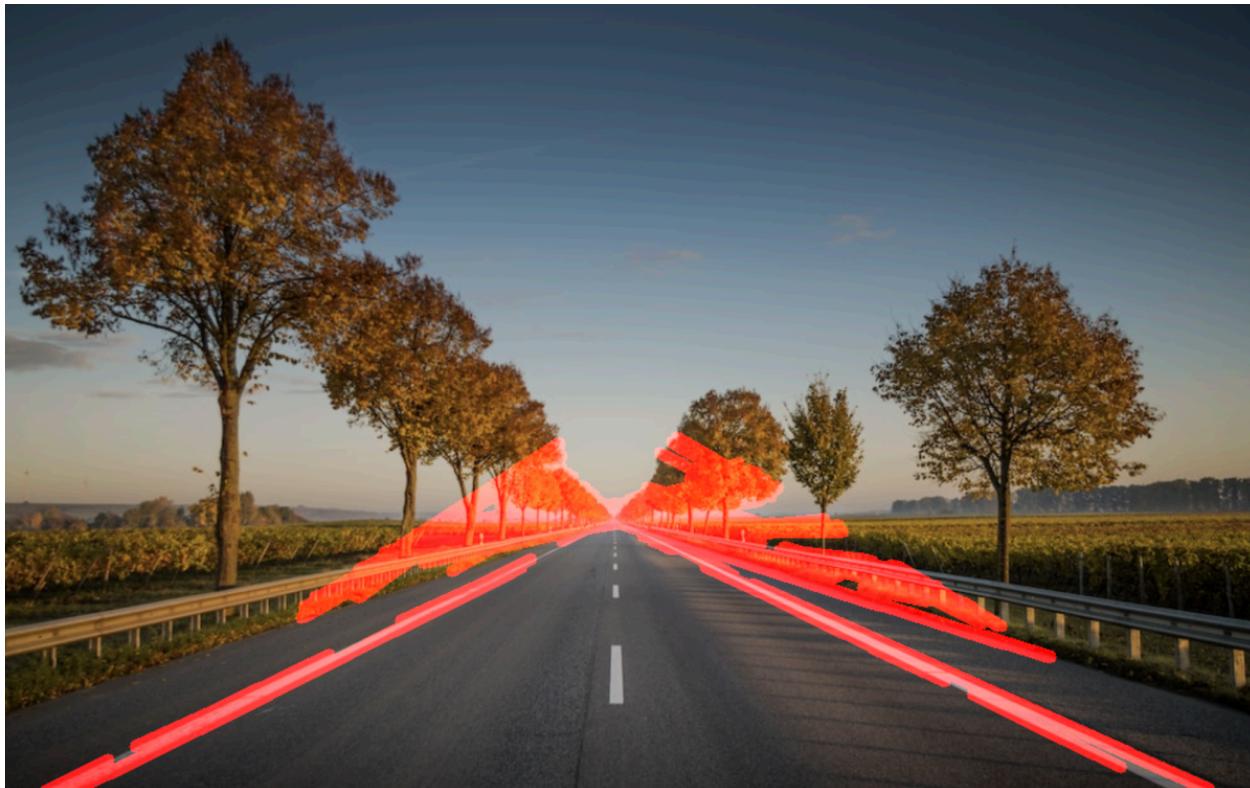
```
1 import cv2
2 import numpy as np
3 img = cv2.imread('road.png')
4 gray = cv2.cvtColor(img, cv2.COLOR_BGR2GRAY)
5 blur = cv2.GaussianBlur(gray, (5, 5), 0)
6 edges = cv2.Canny(blur, 50, 150)
7 mask = np.zeros_like(edges)
8 height, width = mask.shape
9 polygon = np.array([[0, height],
10 | (width,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

Code-2(with live video capturing)

INPUT:

```
1 import cv2
2 import numpy as np
3
4 cap=cv2.VideoCapture(0)
5 cap.set(3,640)
6 cap.set(4,480)
7
8 ret,img = cap.read()
9 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([[[
15     (0, height),
16     (width, height),
17     (width // 2, height // 2)
18 ]])
19 cv2.fillPoly(mask, polygon, 255)
20 masked_edges = cv2.bitwise_and(edges, mask)
21
22 lines = cv2.HoughLinesP(masked_edges, rho=6, theta=np.pi/60, threshold=160,
23 lines=np.array([]), minLineLength=40, maxLineGap=25)
24
25 line_img = np.zeros_like(img)
26 for line in lines:
27     x1, y1, x2, y2 = line[0]
28     cv2.line(line_img, (x1, y1), (x2, y2), (0, 0, 255), 10)
29
30 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:

```
1 import cv2
2 import numpy as np
3
4 cap=cv2.VideoCapture(0)
5 cap.set(3,640)
6 cap.set(4,480)
7
8 ret,img = cap.read()
9 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([[ 
15     (0, height),
16     (width, height),
17     (width // 2, height // 2)
18 ]])
19 cv2.fillPoly(mask, polygon, 255)
20 masked_edges = cv2.bitwise_and(edges, mask)
21
22 lines = cv2.HoughLinesP(masked_edges, rho=6, theta=np.pi/60, threshold=160, lines=np.array([]), minLineLength=40, maxLineGap=25)
23
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)
30
31 cv2.imshow('Result', result)
32 cv2.waitKey(1)
```