Lane Detection PS:

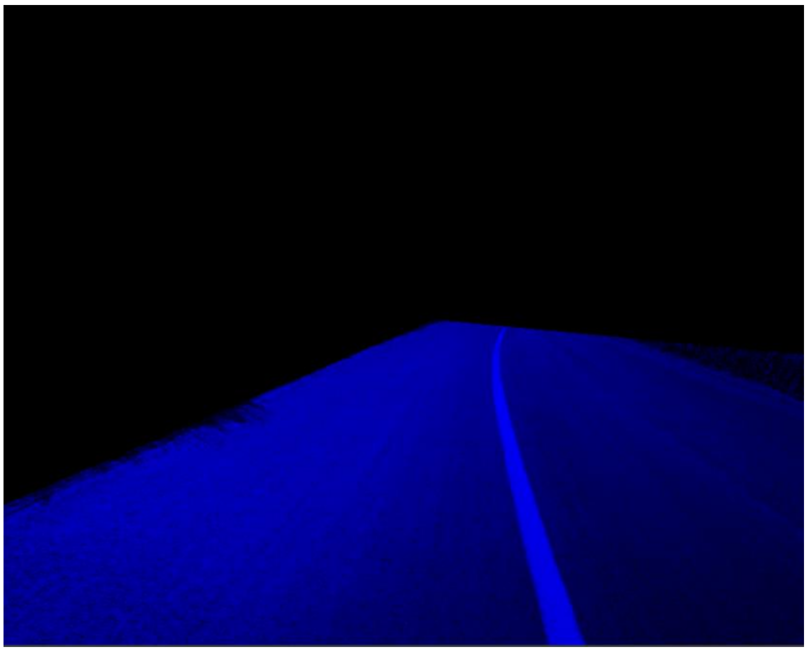
Name: Sarang Jagdish

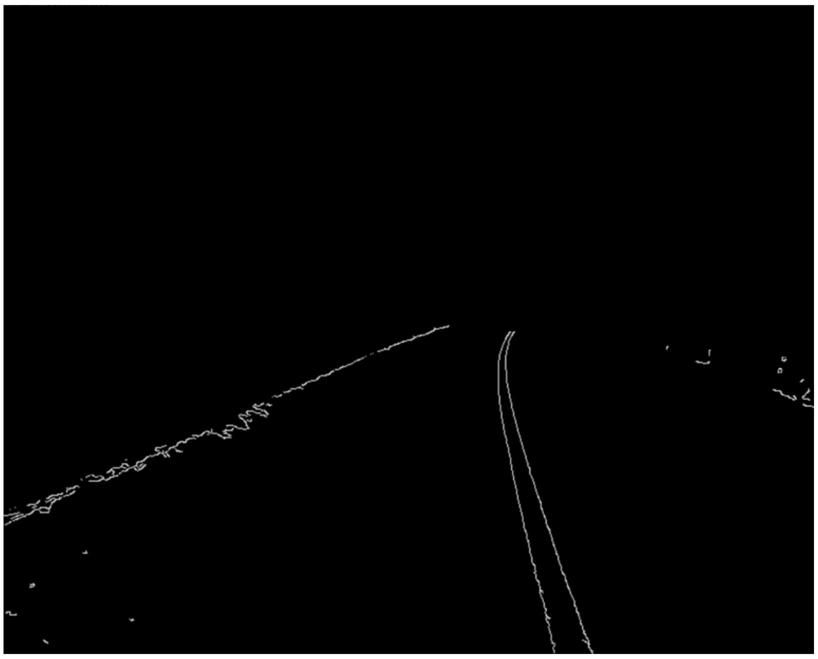
Code:

```
import cv2

import numpy as np


def background_isolation(image):

   height, width = image.shape[:2]

   mask = np.zeros_like(image)

   polygon = np.array([[(0, 640),(0,500),(430,315), (800,350),(800,640)]], np.int32)

   cv2.fillPoly(mask, [polygon],255)

   masked_edges = cv2.bitwise_and(image, mask)

   return masked_edges


image = cv2.imread("G:/My Drive/IIT INDORE BTECH MECH ENGG/ivdc club/autonomy/Lane_2.png")

blur=cv2.GaussianBlur(image, (5,5),0.9)

gray_image = cv2.cvtColor(blur, cv2.COLOR_RGB2GRAY)

cannyed_image = cv2.Canny(gray_image,150, 255)

masked = background_isolation(cannyed_image)

masked2=background_isolation(image)

def draw_lines(img, lines, color=[0, 255, 0], thickness=10):

   if lines is None:

       return

   img = np.copy(img)

   line_img = np.zeros(

     (

       img.shape[0],

       img.shape[1],

       3

     ),
     dtype=np.uint8,
   )
   for line in lines:
     for x1, y1, x2, y2 in line:
       cv2.line(line_img, (x1, y1), (x2, y2), color, thickness)
   img = cv2.addWeighted(img,0.8, line_img, 1.0, 0.0)
   return img
lines = cv2.HoughLinesP(
   masked,
   rho=2,
   theta=np.pi / 180,
   threshold=100,
   lines=np.array([]),
   minLineLength=0.5,
   maxLineGap=200
)
line_image = draw_lines(image, lines)
gray_image = cv2.cvtColor(image, cv2.COLOR_BGR2GRAY)
_, thresh = cv2.threshold(gray_image, 145, 255, cv2.THRESH_BINARY_INV)
contours, _ = cv2.findContours(thresh, cv2.RETR_EXTERNAL, cv2.CHAIN_APPROX_SIMPLE)
filled_image = cv2.fillPoly(image, contours, (0, 0, 0))
bitmap = cv2.bitwise_not(filled_image)
cv2.imshow("bitmap", bitmap)
cv2.imshow("path",line_image)
cv2.imshow("masked",masked)
cv2.imshow("mask2",masked2)
cv2.waitKey(0)
cv2.destroyAllWindows()
```

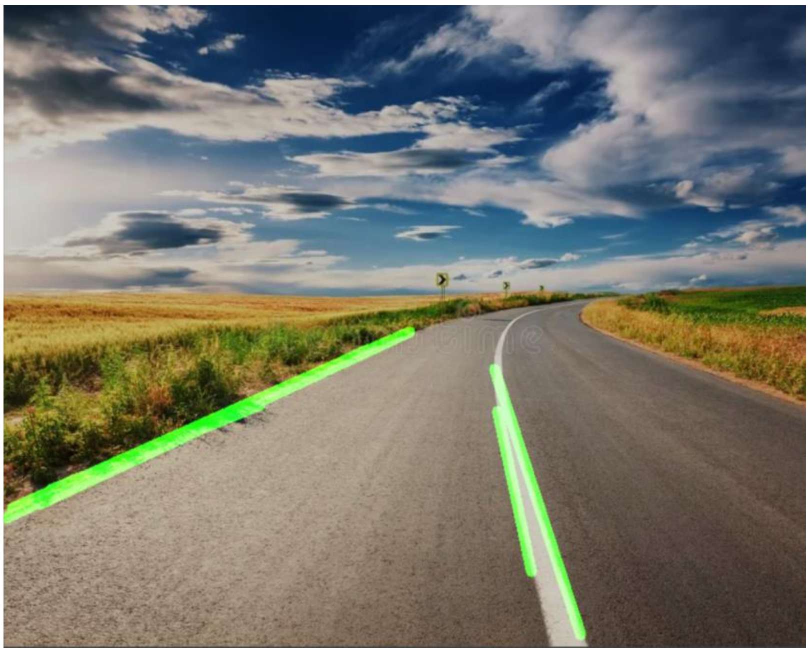
Output:

Background Isolation and Canny edges:





Lane Detection:



Bitmap: