

Detectarea benzilor de circulatie

Studenti: Simona Toma, Robert Ursu

Disciplina: Vedere artificiala pentru vehicule

Informatica Aplicata Anul 3

Introducere

Siguranța în trafic devine o necesitate din ce în ce mai mare odată cu creșterea traficului urban. Ieșirea de pe bandă fără respectarea regulilor de circulație este cauza principală a majorității accidentelor de pe străzi din ziua de azi, cele mai multe dintre acestea fiind provocate din cauza neatentiei sau oboselii soferilor.

Respectarea regulilor de circulație pe benzi este crucială pentru siguranța rutieră atât pentru șoferi, cât și pentru pietoni. Detectarea și identificarea liniei benzii sunt foarte utile când vine vorba despre asistența avansată la parcare, menținerea benzii de circulație, avertismentele off-road și schimbarea benzii.

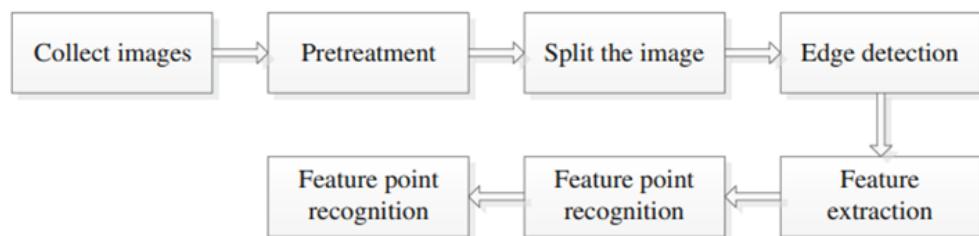
Self-Driving Car este una dintre cele mai revoluționare tehnologii aduse de inteligența artificială. Mașinile autonome utilizează tehnologia de detectare a benzilor pentru a nu le depasi, circulând doar în interiorul acestora și prevenind în acest mod coliziunile din trafic.

Acest tip de sistem inteligent asigură întotdeauna călătoria în siguranță însă, pot apărea diferite obstacole în detectarea corectă și preciză a benzilor, cum ar fi condițiile proaste de drum, cantitatea insuficientă de vopsea utilizată pentru marcarea limitelor benzii, ceata, lipsa iluminatului stradal pe timp de noapte, umbrele copacilor sau alte automobile.

Asadar, proiectul realizat are scopul de a identifica marcajele benzii cu intenția de a obține un mediu sigur de circulație în trafic. Tehnologia folosește o cameră instalată pe mașină pentru a capta vederea din față, apoi aplică câteva proceduri pentru a determina benzile.

Metodologia de realizare

Procesul de detectare și recunoaștere a benzii constă din următoarele componente: preluarea imaginii, procesarea imaginii, segmentarea imaginii, detectarea marginilor, extragerea caracteristicilor, identificarea punctelor caracteristice și recunoașterea liniei benzii (fig. 1).



Librarii utilizate:

1. OpenCV
2. numpy
3. matplotlib

Codul aplicatiei:

```
import cv2 as cv
import numpy as np
from matplotlib import pyplot as plt

def lanesDetection(img):
    # img = cv.imread("./img/road.png")
```

```

# img = cv.cvtColor(img, cv.COLOR_BGR2RGB)

# print(img.shape) # returneaza dimensiunile imaginii

height = img.shape[0]
width = img.shape[1]

region_of_interest_vertices = [
    (200, height), (width/2, height/1.37), (width-300, height)
]

gray_img = cv.cvtColor(img, cv.COLOR_RGB2GRAY) # rgb to gray
edge = cv.Canny(gray_img, 50, 100, apertureSize=3)
cropped_image = region_of_interest(
    edge, np.array([region_of_interest_vertices], np.int32)) # imaginea decupata

lines = cv.HoughLinesP(cropped_image, rho=2, theta=np.pi/180,
                       threshold=50, lines=np.array([]), minLineLength=10,
                       maxLineGap=30) # detecteaza liniile din imaginea decupata

image_with_lines = draw_lines(img, lines)

# plt.imshow(image_with_lines)

# plt.show()

return image_with_lines

# defineste regiunea de interes din imagine

def region_of_interest(img, vertices):
    mask = np.zeros_like(img)

```

```

match_mask_color = (255)

cv.fillPoly(mask, vertices, match_mask_color)

masked_image = cv.bitwise_and(img, mask)

return masked_image


# traseaza liniile pe imaginea originala

def draw_lines(img, lines):

    img = np.copy(img)

    blank_image = np.zeros((img.shape[0], img.shape[1], 3), np.uint8)

    for line in lines:

        for x1, y1, x2, y2 in line:

            cv.line(blank_image, (x1, y1), (x2, y2), (0, 255, 0), 2)

    img = cv.addWeighted(img, 0.8, blank_image, 1, 0.0)

    return img


def videoLanes():

    cap = cv.VideoCapture('./img/video.mp4') # este preluat videoclipul din folder

    while(cap.isOpened()):

        ret, frame = cap.read() # memoreaza frame-ul

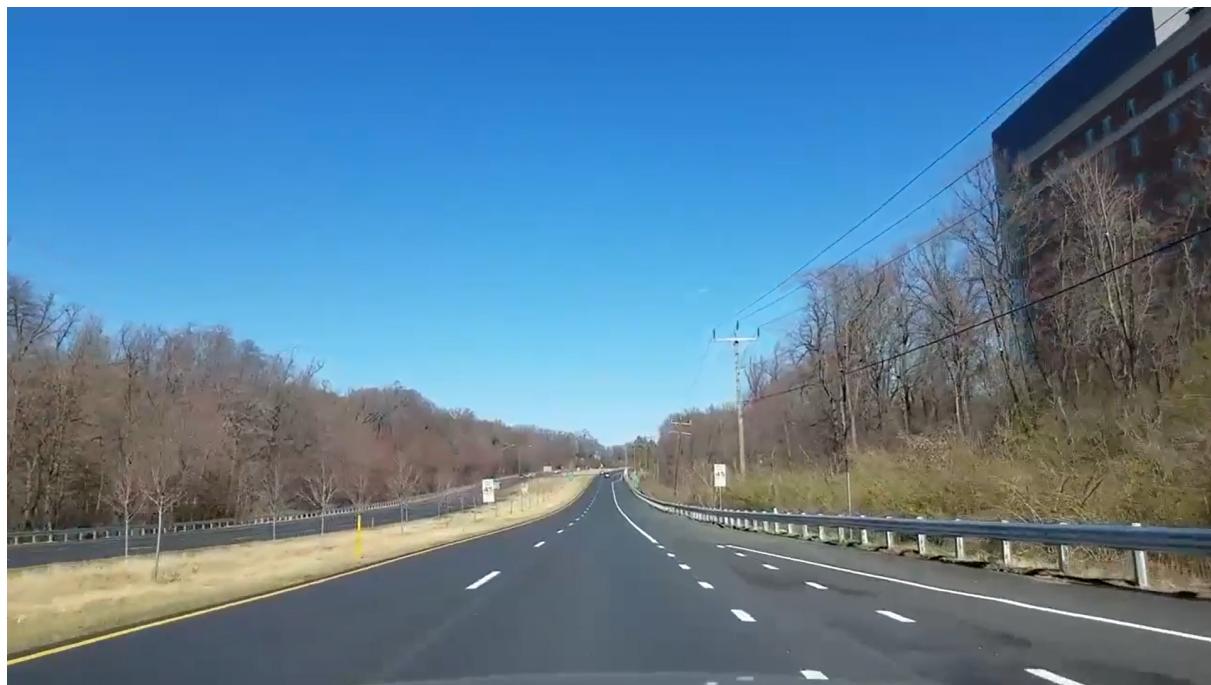
        frame = lanesDetection(frame) # apeleaza functia avand ca parametru frame-ul
        respectiv

        cv.imshow('Lanes Detection', frame) # afiseaza frame-urile

```

```
if cv.waitKey(1) & 0xFF == ord('q'): # se inchide programul prin apasarea tastei  
Q  
  
break  
  
cap.release()  
  
cv.destroyAllWindows()  
  
if __name__ == "__main__":  
    videoLanes()
```

Imaginea originală:



OUTPUT:

