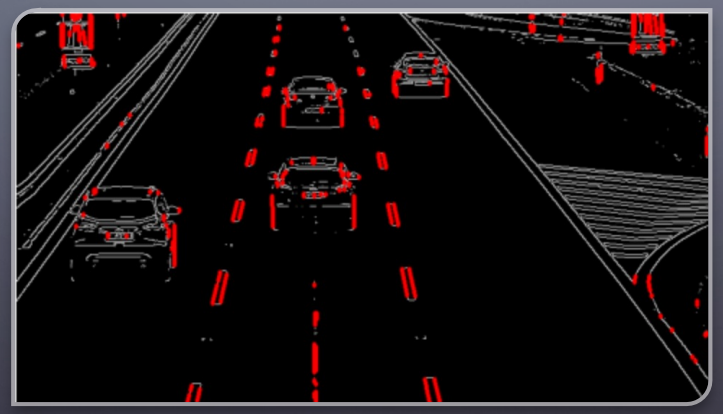
FINAL PROJECT

BY: NOORA NOOR AND CHRISTINA CHUM

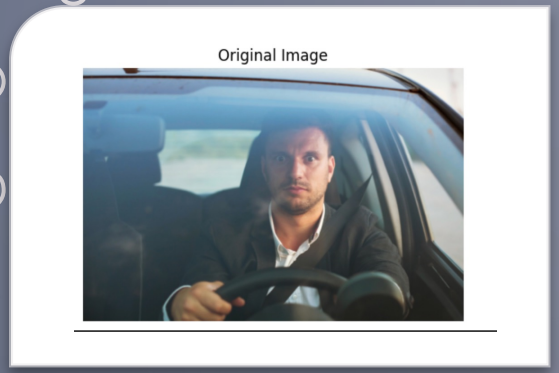
ABOUT THE PROJECT

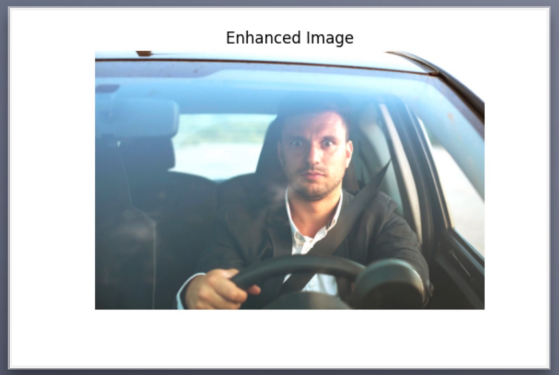
- Lane detection
- Enhance or make quality of photo better
- Face recognition
- Pedestrian recognition
- License plate recognition

LANE DETECTION



- Probabilistic hough line perform lane detection on gray scaled image
- Apply canny edge to highlight potential lane lines
- Hough transform identifies lines corresponding to lanes
- Filtered lines recognized lane lines based on orientation drawn in red
- Allow identification and visualization markings on road image

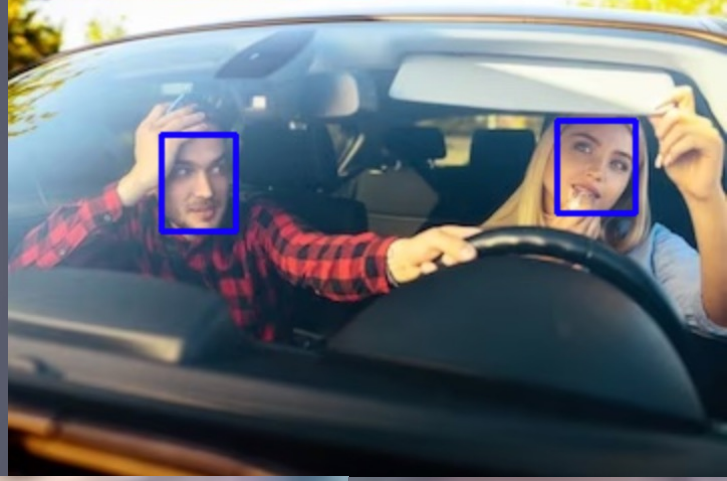




ENHANCING OR MAKING IMAGE BETTER

- First photo convert image from BGR to RGB
- Adjust the contract and brightness
- Second photo use bilateral to reduce the noise
- Gaussian blur to remove blur by deconvolution using the unsharp mask filter
- Also converted photo from BRG to RGB
- Enhanced contrast and brightness

FACE RECOGNITION



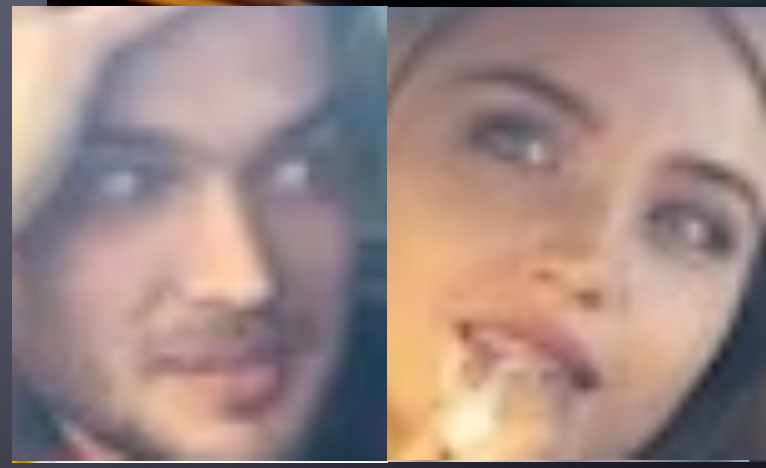
- Use a face detection opencv
- Zoom in code for face

```
for idx, face in enumerate(faces):
x, y, w, h = face['box']
```

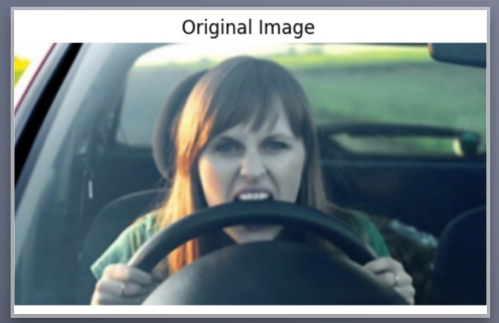
- Resize ROI by 2
- Scale induvial face by 5
- Make rectangle on the face

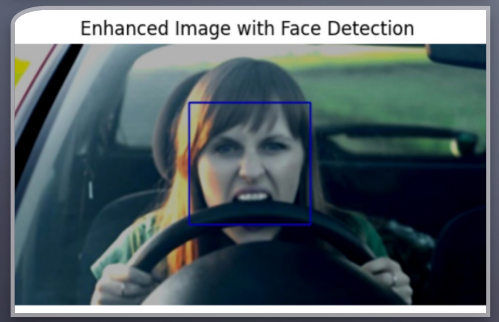
```
for face in faces:
x, y, w, h = face['box']
cv2.rectangle(image, (x, y), (x+w,
y+h), (255, 0, 0), 3)
```

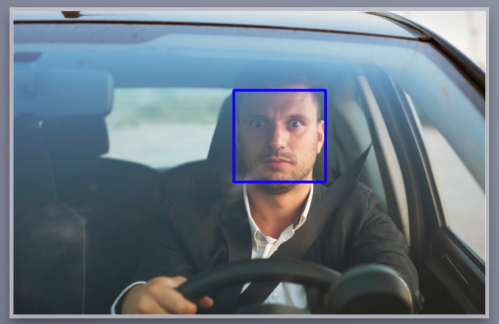
- Convert BGR2RGB to display photo

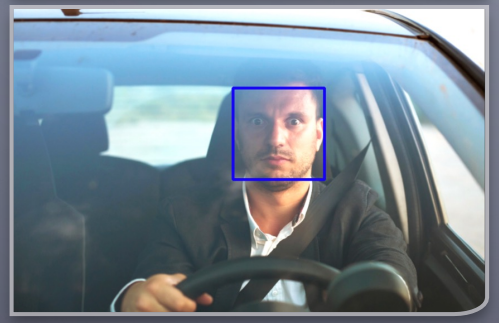


FACE RECOGNITION COMBINED WITH ENHANCING OR MAKING PHOTO BETTER









- Haarcaascaade use for face classification
- Converting image from BGR2GRAY
- Face detect open cv to detect the face in the image
- Code use to make rectangle on face

```
for face in faces:
x, y, w, h = face['box']
cv2.rectangle(image, (x, y), (x+w, y+h), (255, 0, 0), 3)
```

- Enhance photo by changing the contrast and brightness

PEDESTRIAN RECOGNITION

- HOG decipher used to detect person
- Use this code to make rectangle around pedestrian `for (x, y, w, h) in pedestrians:`

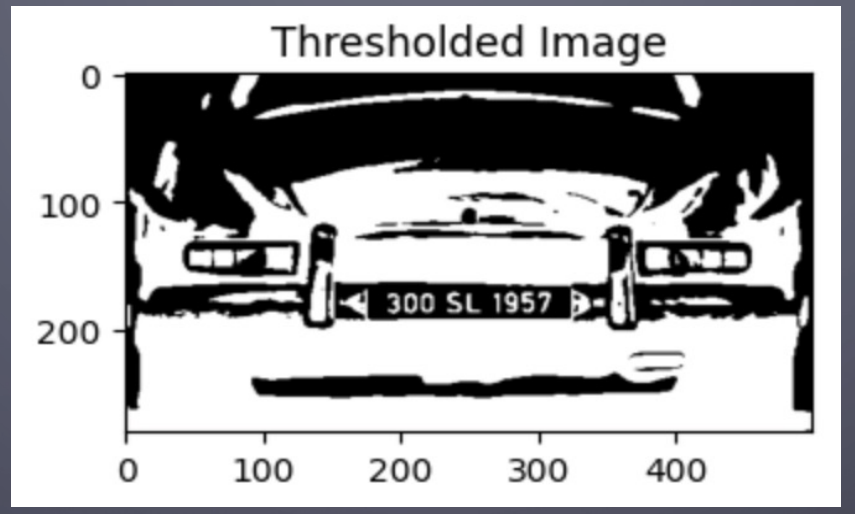
```
cv2.rectangle(mask, (x, y), (x+w, y+h), (255, 255, 255), thickness=cv2.FILLED)
```

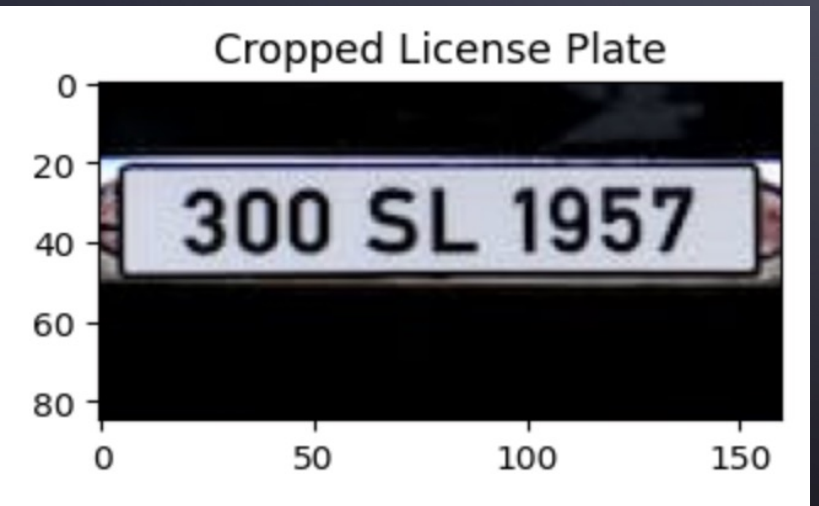
- Guassian blur used to blur out background

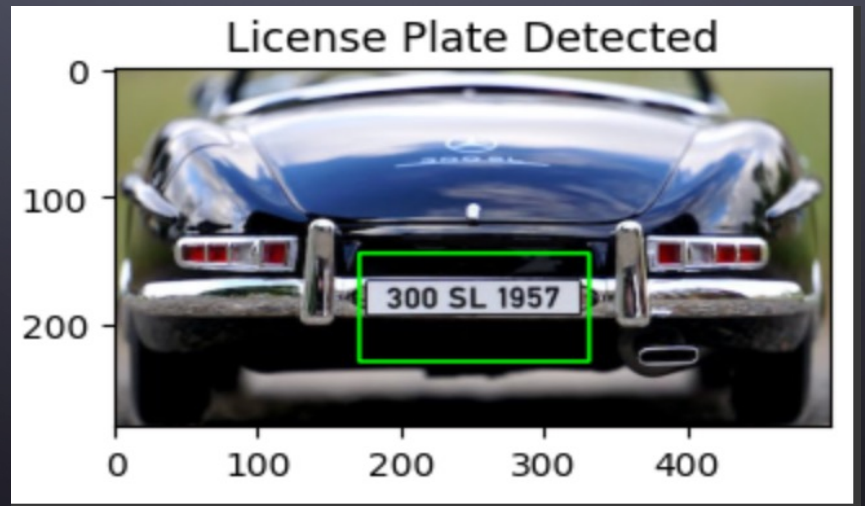
Use for loop to zoom in on pedestrian.

```
for (x, y, w, h) in pedestrians:
# Calculate the zoomed-in region
zoomed_x = max(0, x - int((zoom_factor - 1) * w / 2))
zoomed_y = max(0, y - int((zoom_factor - 1) * h / 2))
zoomed_w = min(image.shape[1], int(w * zoom_factor))
zoomed_h = min(image.shape[0], int(h * zoom_factor))
```

LICENSE PLATE RECOGNITION







- Convert image from BGR to GRAY
- Used Gaussin blur to reduce the noise and improve accuracy
- Made Thresholding to create binary image
- Contour threshold image
- Masked the detected license plate
- Calculate the coordinate for the plate.
- Crop the plate.

LICENSE PLATE RECOGNITION CONTINUED

Same codes as the previous slide to get outcome

TROUBLES THAT WE HAD

- The lane detection was hard to do
- Just detecting the lanes with the codes for hough did not work so well
- Enhancing the image was hard
- Hard to tweak code a lot to enhance or make the image better
- Some photos did not work well with the codes

ALL DONE!!!.... WAIT IS THAT OJ.....