```
import cv2 import time

import RPi.GPIO as GPIO #

Setup GPIO pins


GPIO.setmode(GPIO.BCM)

LEFT_RELAY_PIN = 17     #Pin for left relay

RIGHT_RELAY_PIN = 17     #Pin for right relay

GPIO.setup (LEFT_RELAY_PIN, GPIO.OUT)

GPIO.setup (RIGHT_RELAY_PIN, GPIO.OUT)

GPIO.setup (LEFT_RELAY_PIN, GPIO.LOW)

GPIO.setup (RIGHT_RELAY_PIN, GPIO.LOW)

#Threshold to detect object

Thres = 0.45

Threshold_time = 10

#Load pre.trained model and classes classNames = [ ] with

open(‘coco.names’ , ‘r’) as f :          classNames =

f.read().rstrip(‘\n’).split(‘\n’) configPath =

‘ssd_mobilenet_v3_large_coco_2020_01_14.pbtx weightsPath =

‘frozen_interference_graph.pb’



net = cv2.dnn_DetectionModel(weightsPath,configPath) net.setInputSize(320,320)
net.setInputScale(1.0/127.5) net.setInputMean((127.5,127.5,127.5)) net.setInputSwapRB(True)

#Main function Def main() :       global cap

cap = cv2,VideoCapture(0)

cap.set(3,1280)     #Width of the frame
cap.set(4, 720)  # Height of the frame

cap.set(10, 70)

cap.set(cv2.CAP_PROP_EXPOSURE, -12)


start_time = time.time()


# Define midpoint for left and right distinction

frame_width = int(cap.get(3)) midpoint =

frame_width // 2



# Run the main loop while True:     success, img =

cap.read()     classIds, confs, bbox = net.detect(img,

confThres)     # Draw the red separation line in the

middle of the frame     cv2.line(img, (midpoint, 0),

(midpoint, int(cap.ge cv2.line(img, (midpoint, 0),

(midpoint, int(cap.g


# Initialize relay states

left_detected = False

right_detected = False
```

```
if len(classIds) != 0:     for classId, confidence, box

in zip(classIds, c        if classId == 1:  # Check if

detected obj            x, y, w, h = box

cv2.rectangle(img, box, color=(0, 0,
```

```
            # Check if detected object is on the

if x + w // 2 < midpoint:

left_detected = True            else:


                right_detected = True cv2.line(img,

(midpoint, 0), (midpoint, int(cap.g
```

```
# Initialize relay states
left_detected = False

right_detected = False
```

```
if len(classIds) != 0:     for classId, confidence, box

in zip(classIds, c        if classId == 1:  # Check if

detected obj            x, y, w, h = box

cv2.rectangle(img, box, color=(0, 0,
        # Check if detected object is on the

if x + w // 2 < midpoint:

left_detected = True            else:


           right_detected = True
```