# Autonomous driving copilot: Gesture control and autonomous driving system

SIWEN TU, LIN HE, RUILIN MA, CHENYU SHI, and SHUPEI LI, Team: < Pi team >, LIACS

**Abstract**. TBC.

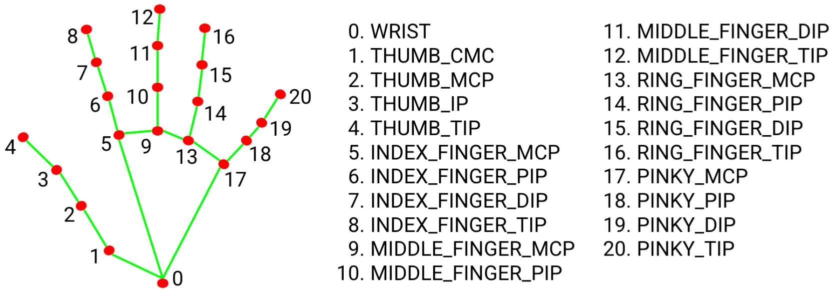
**Key Words**: keywords

## 1 INTRODUCTION

## 2 GESTURE CONTROL

During the gesture control stage, the driver sits in front of the computer, obeserving the road condition through the video stream sent from the raspberry pi mobile car, and controls the movement of the car via gestures. The key part of gesture control is correctly identifying gestures in a short time. Our solution is using the CVZone package's implementation of the MediaPipe Hands model [3].

The MediaPipe Hands model is a pipeline designed for detecting landmarks of hands in images or videos. It is composed of two stages. The first stage is marking a square area where palms are present with a single-shot palm detector. And the second stage is recognizing the handedness and the landmarks of hands with an encoder-decoder-based tracker. Note that the handedness is a flag indicating whether the detected hand is left or right. The landmarks of hands provide critical information for restoring gestures. The MediaPipe Hands model adopts the topology suggested by [2] and returns the 21 landmarks, which are shown in Figure 1. The basic building block of the detector and the tracker is the convolutional neural network. The MediaPipe Hands model has been pre-trained on three large gesture datasets and added as a module in the MediaPipe framework.



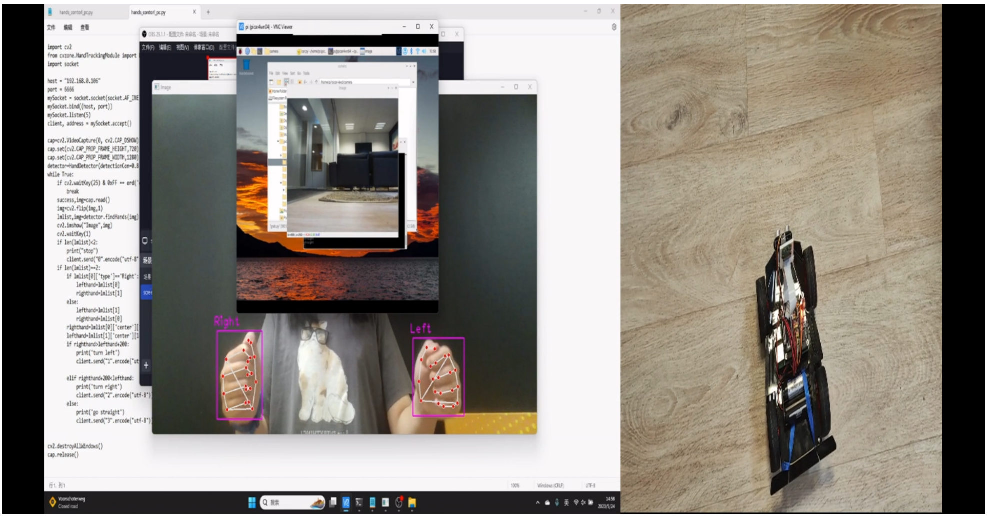
0. WRIST
1. THUMB_CMC
2. THUMB_MCP
3. THUMB_IP
4. THUMB_TIP
5. INDEX_FINGER_MCP
6. INDEX_FINGER_PIP
7. INDEX_FINGER_DIP
8. INDEX_FINGER_TIP
9. MIDDLE_FINGER_MCP
10. MIDDLE_FINGER_PIP
11. MIDDLE_FINGER_DIP
12. MIDDLE_FINGER_TIP
13. RING_FINGER_MCP
14. RING_FINGER_PIP
15. RING_FINGER_DIP
16. RING_FINGER_TIP
17. PINKY_MCP
18. PINKY_PIP
19. PINKY_DIP
20. PINKY_TIP

Fig. 1. The topology of the hand [1].

The control logic of our algorithm mainly utilizes the number and the relative height of hands. CVZone provides APIs that can return a list of detected hands and extract the center coordinates of hands from 21 landmarks. The gesture control program will only be activated if two hands are appearing in front of the computer camera, which simulates the scenario of driving a real car by steering the wheel. When the number of hands is less than two, which indicates that the drivier is not ready to start or wishes to stop, the program will brake the mobile car immediately. Our

*Robotics'23, Master CS, Spring 2023, Leiden, the Netherlands*

program supports three motion commands: turn left, turn right, and go straight. We use the relative height of two hands to determine the intent of the driver. Denote the $y$ coordinate of the center of the left hand as $y_{\text{left}}$ and that of the right hand as $y_{\text{right}}$. Given a threshold $\eta$, we regard the driver wants to turn left if $y_{\text{right}} > y_{\text{left}} + \eta$ or turn right if $y_{\text{left}} > y_{\text{right}} + \eta$. In other cases, we regard the driver wants to go straight. We set the $\eta$ to 200 according to testing results. If the driver wants to stop or switch to the autonomous driving mode, he or she just needs to move hands outside the view of the camera to deactivate the gesture control program. Figure 2 illustrates the effect of the gesture control program in action. Monitor screens of the computer camera and the car's camera are shown on the left side of the screenshot. The right side of the screenshot is the car going straight.



```
import cv2
from cvzone.HandTrackingModule import
import socket

host = "192.168.0.106"
port = 6666
mySocket = socket.socket(socket.AF_INE
mySocket.bind((host, port))
mySocket.listen(5)
client, address = mySocket.accept()

cap=cv2.VideoCapture(0, cv2.CAP_DSHOW)
cap.set(cv2.CAP_PROP_FRAME_HEIGHT,720)
cap.set(cv2.CAP_PROP_FRAME_WIDTH,1280)
detector=HandDetector(detectionCon=0.8
while True:
    if cv2.waitKey(25) & 0xFF == ord('
        break
    success,img=cap.read()
    img=cv2.flip(img,1)
    lmlist,img=detector.findHands(img)
    cv2.imshow("Image",img)
    cv2.waitKey(1)
    if len(lmlist)<2:
        print("stop")
        client.send("0".encode("utf-8"
    if len(lmlist)==2:
        if lmlist[0]['type']=='Right':
            lefthand=lmlist[0]
            righthand=lmlist[1]
        else:
            lefthand=lmlist[1]
            righthand=lmlist[0]
        righthand=lmlist[0]['center'][
        lefthand=lmlist[1]['center'][1
        if righthand>lefthand+200:
            print('turn left')
            client.send("1".encode("ut

        elif righthand+200<lefthand:
            print('turn right')
            client.send("2".encode("utf-8")
        else:
            print('go straight')
            client.send("3".encode("utf-8")


cv2.destroyAllWindows()
cap.release()
```

Fig. 2. Gesture control in action.

## 3 AUTONOMOUS DRIVING

## 4 CONCLUSION

## REFERENCES

[1] Google. 2023. Mediapipe/hands.md at master · google/mediapipe. https://github.com/google/mediapipe/blob/master/docs/solutions/hands.md

[2] Tomas Simon, Hanbyul Joo, Iain Matthews, and Yaser Sheikh. 2017. Hand Keypoint Detection in Single Images using Multiview Bootstrapping. (2017). arXiv:1704.07809 [cs.CV]

[3] Fan Zhang, Valentin Bazarevsky, Andrey Vakunov, Andrei Tkachenka, George Sung, Chuo-Ling Chang, and Matthias Grundmann. 2020. MediaPipe Hands: On-device Real-time Hand Tracking. (2020). arXiv:2006.10214 [cs.CV]