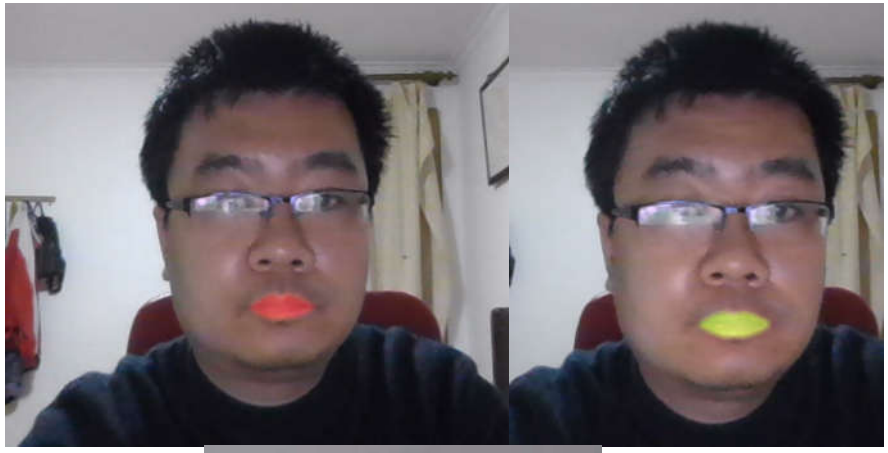
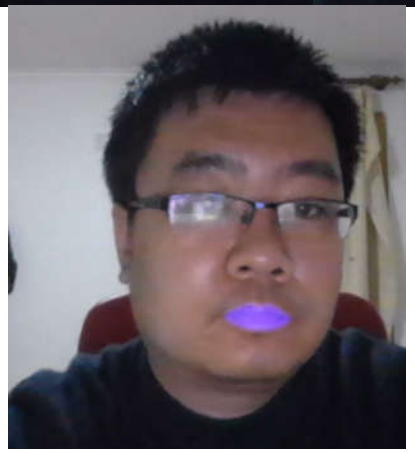
Python编程与人工智能实践



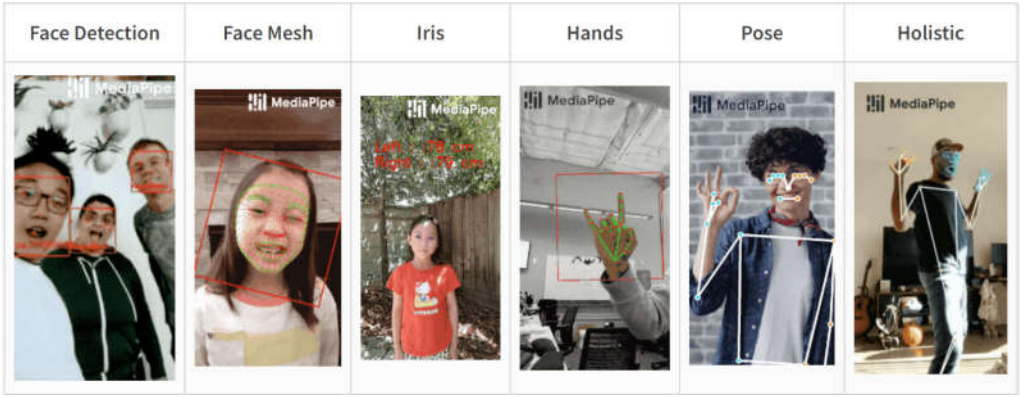


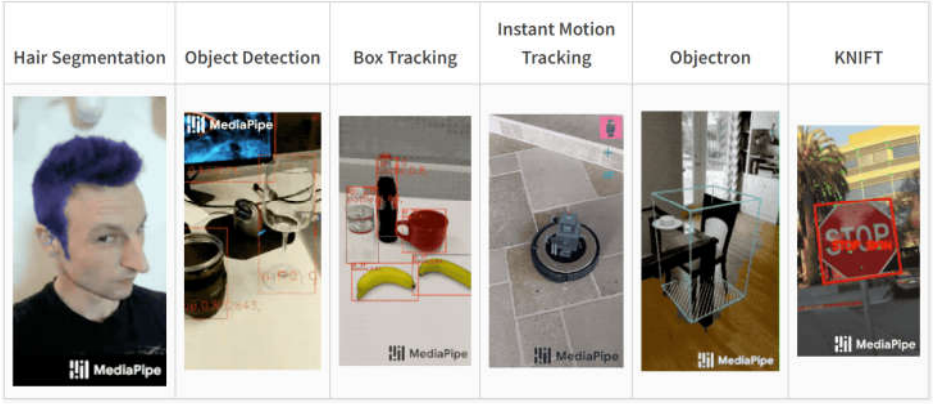
应用篇：
基于mediapipe的人脸关键点检测
魔幻口红特效

于泓
鲁东大学
信息与电气工程学院
2022.3.23

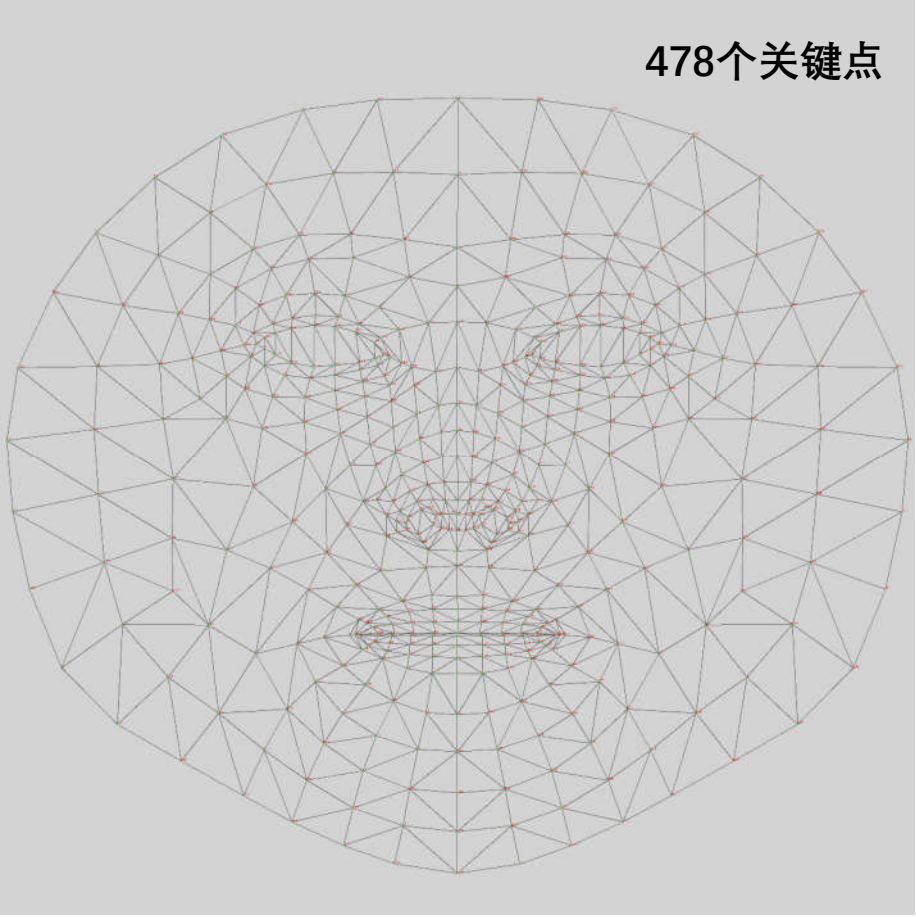
MediaPipe：Google Research 开源的跨平台多媒体机器学习模型应用框架

作为一款跨平台框架，MediaPipe 不仅可以被部署在服务器端，更可以在多个移动端（安卓和苹果 iOS）和嵌入式平台（Google Coral 和树莓派）中作为设备端机器学习推理（On-device Machine Learning Inference）框架。





Face-mesh



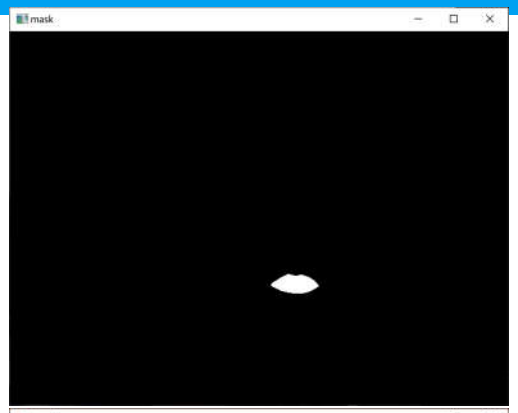
上嘴唇

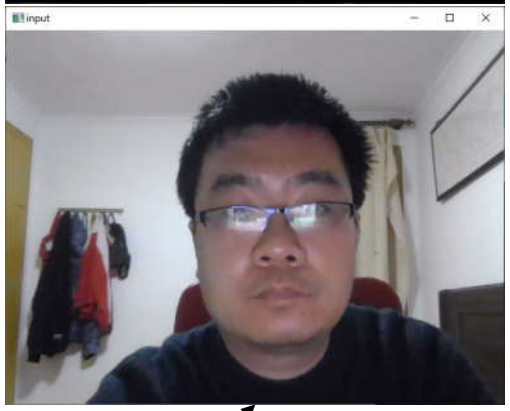
[61, 185, 40, 39, 37,0, 267, 269, 270, 409,
291,308,415,310,311,312,13,82,80,191,78,61]

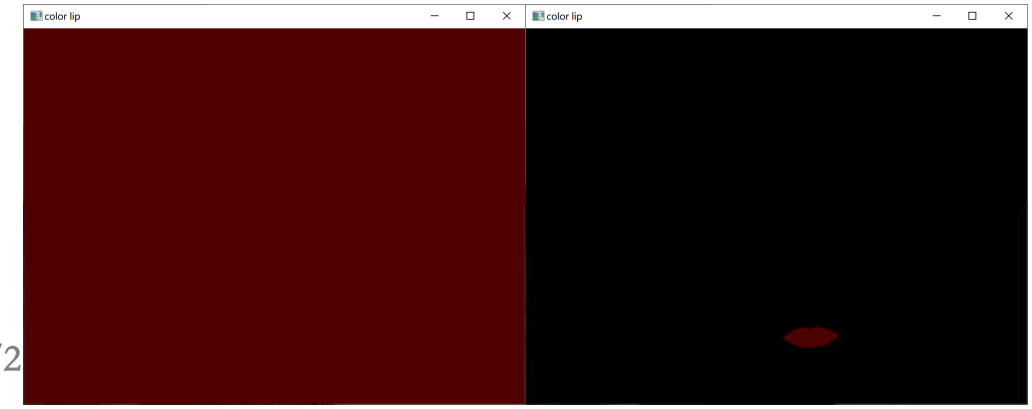
下嘴唇

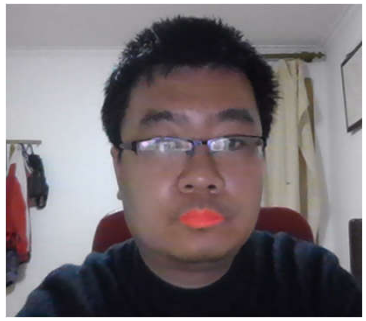
[78, 95, 88, 178, 87, 14, 317, 402, 318, 324,
308,291,375,321,405,314,17,84,181,91,146,61,78]

```python
def change_color_lip(img,list_lms,index_lip_up,index_lip_down,color):
    # cv2.imshow("input",img)

    mask = np.zeros_like(img)
    points_lip_up = list_lms[index_lip_up,:]
    mask = cv2.fillPoly(mask,[points_lip_up],(255,255,255))

    points_lip_down = list_lms[index_lip_down,:]
    mask = cv2.fillPoly(mask,[points_lip_down],(255,255,255))

    # cv2.imshow("mask",mask)

    img_color_lip = np.zeros_like(img)
    img_color_lip[:] = color
    img_color_lip = cv2.bitwise_and(mask,img_color_lip)
    # cv2.imshow("color lip",img_color_lip)

    img_color_lip = cv2.GaussianBlur(img_color_lip,(7,7),10)
    img_color_lip = cv2.addWeighted(img,1,img_color_lip,0.8,0)

    return img_color_lip
```









C=AX1
+BX0.8
+0

A

B

+

C

```python
def empty(a):
    pass


if __name__ == "__main__":

    # 创建人脸关键点检测对象
    mp_face_mesh = mp.solutions.face_mesh
    face_mesh = mp_face_mesh.FaceMesh(static_image_mode=False,
                                      max_num_faces=1,
                                      refine_landmarks=True,
                                      min_detection_confidence=0.5,
                                      min_tracking_confidence=0.5)

    # 口红颜色调节
    cv2.namedWindow("BGR")
    cv2.resizeWindow("BGR",640,240)
    cv2.createTrackbar("Blue","BGR",0,255,empty)
    cv2.createTrackbar("Green","BGR",0,255,empty)
    cv2.createTrackbar("Red","BGR",0,255,empty)
```

```
while True:

    # 读取一帧图像
    success, img = cap.read()
    if not success:
        continue

    # 获取宽度和高低
    image_height, image_width, _ = np.shape(img)

    # BGR 转 RGB
    img_RGB = cv2.cvtColor(img, cv2.COLOR_BGR2RGB)
    results = face_mesh.process(img_RGB)
    list_lms = []
    if  results.multi_face_landmarks:
        face_landmarks = results.multi_face_landmarks[0]

        for i in range(478):
            pos_x = int(face_landmarks.landmark[i].x * image_width)
            pos_y = int(face_landmarks.landmark[i].y * image_height)
            list_lms.append((pos_x,pos_y))

        list_lms = np.array(list_lms,dtype=np.int32)
        index_lip_up = [61, 185, 40, 39, 37,0, 267, 269, 270, 409, 291,308,415,310,311,312,13,82,80,191,78,61]
        index_lip_down = [78, 95, 88, 178, 87, 14, 317, 402, 318, 324, 308,291,375,321,405,314,17,84,181,91,146,61,78]

        # 获取口红颜色
        b = cv2.getTrackbarPos("Blue","BGR")
        g = cv2.getTrackbarPos("Green","BGR")
        r = cv2.getTrackbarPos("Red","BGR")
```

获取所有关键点坐标

```
        color = (b,g,r)
        img = change_color_lip(img,list_lms,index_lip_up,index_lip_down,color)


    ## 灰度图像 彩色嘴唇
    # img_gray = cv2.cvtColor(img, cv2.COLOR_BGR2GRAY)
    # img_gray = cv2.cvtColor(img_gray, cv2.COLOR_GRAY2BGR)

    # img_show = change_color_lip(img_gray,list_lms,index_lip_up,index_lip_down,(30,30,213))

    cv2.imshow("BGR",img)

    key =  cv2.waitKey(1) & 0xFF

    # 按键 "q" 退出
    if key ==  ord('q'):
        break
cap.release()
```