

Đề tài: **FPV SMART CAR**
Tài liệu Thiết kế - Học kỳ III

Giáo viên hướng dẫn: Hoàng Đức Quang

Lớp: C1.2007.E1

Tên nhóm Nhóm 2

STT	Họ và tên	MSSV
1	Nguyễn Đức Huy	JK-ENR-HA-10123
2	Chu Văn Đức	JK-ENR-HA-10128
3	Lâm Thiên Như Quý	JK-ENR-HA-10126

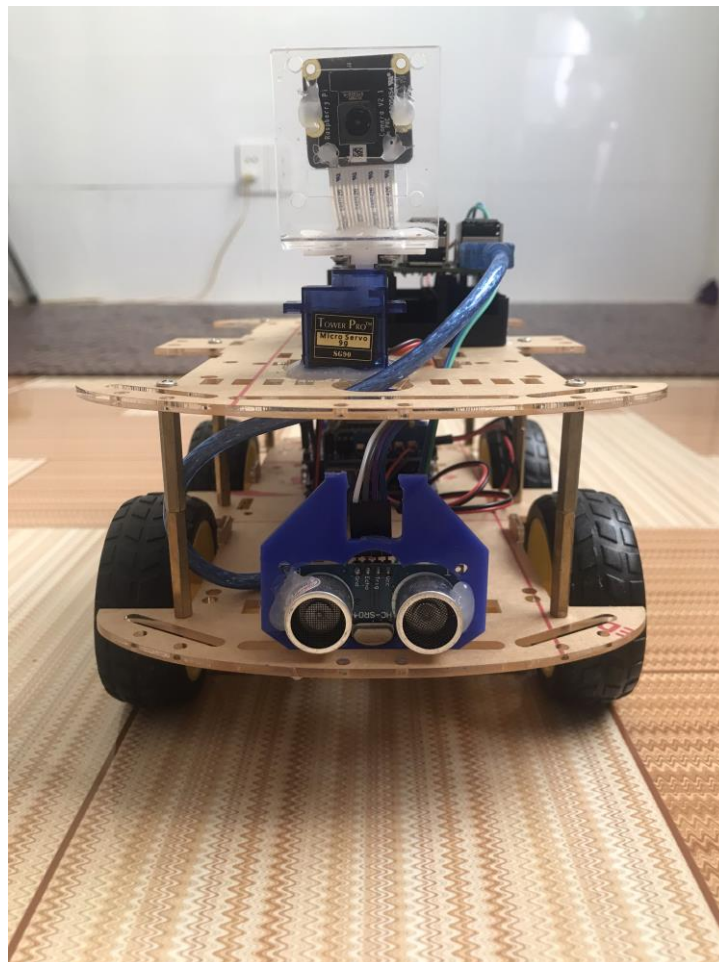
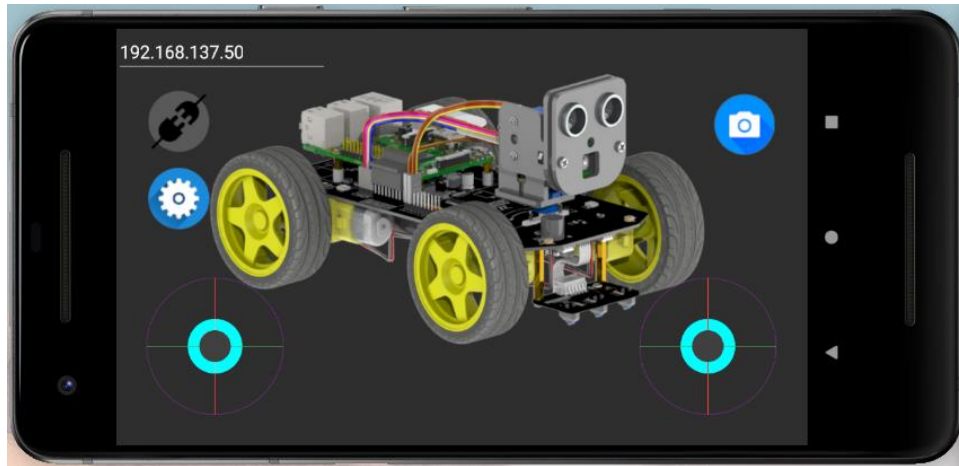
TPHCM, ngày 24 tháng 09 năm 2021

MỤC LỤC

MỤC LỤC	2
GIỚI THIỆU.....	3
I. DANH SÁCH THIẾT BỊ	4
II. SƠ ĐỒ HOẠT ĐỘNG	8
III. LẬP TRÌNH NHÚNG ĐIỀU KHIỂN ARDUINO.....	9
1. Tác vụ điều khiển 4 Motor:.....	9
2. Tác vụ điều khiển Servo Camera:	9
3. Tác vụ đọc cảm biến siêu âm HC-SR04:.....	9
IV. XÂY DỰNG SERVER CHO RASBERRY PI.....	11
1. Đọc dữ liệu từ Client(Android):.....	11
2. Phát video stream lên Website:	12
3. Truyền dữ liệu điều khiển và nhận cảnh báo thông qua cổng Serial:	12
4. Hú còi khi có vật cản:.....	12
V. PHÁT TRIỂN ỨNG DỤNG ANDROID.....	13
1. Connect, Disconnect tới Server:	13
2. Chỉnh tốc độ chạy cho xe:	13
3. Chụp ảnh môi trường:	13
4. Phát video stream:	14
5. Nhận diện hình ảnh:	14
VI. DOWNLOAD CODE	15
VII. ƯU ĐIỂM, NHƯỢC ĐIỂM VÀ HƯỚNG PHÁT TRIỂN	16
1. Ưu điểm:.....	16
2. Nhược điểm:	16
3. Hướng phát triển:.....	16
TÀI LIỆU THAM KHẢO	16

GIỚI THIỆU

FPV Smart Car là hệ thống xe robot được điều khiển thông qua Wifi và mạng Internet dựa trên mô hình Socket. Hệ thống được lập trình với ba ngôn ngữ chính: "Java phát triển ứng dụng Android, Python xây dựng Server trên Raspberry Pi, C lập trình nhúng điều khiển Arduino". Ngoài ra, hệ thống còn tích hợp thư viện Tensorflow để nhận diện hình ảnh.

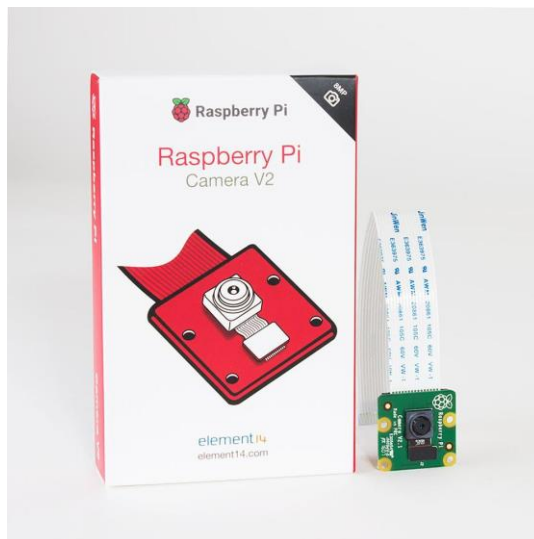


I. DANH SÁCH THIẾT BỊ

Raspberry Pi 3 Model B+



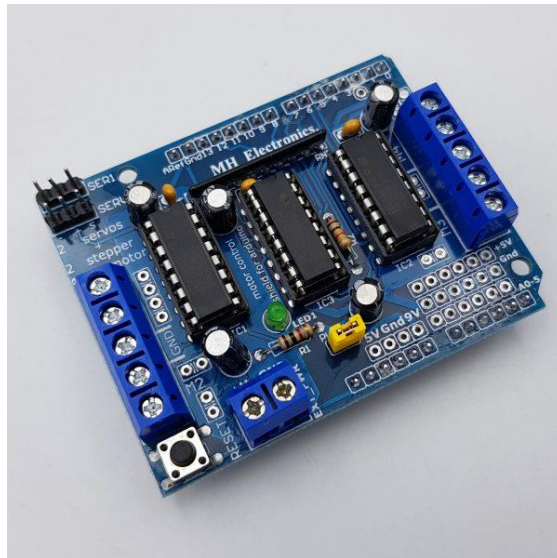
Raspberry Pi Camera Module V2 8MP



Arduino UNO R3



Arduino Motor Shield L293D



Ultrasonic HC-SR04



Động cơ servo SG90 180 độ



Còi Buzzer 5V DC



Pin cell 18650 2600mAh



Hộp để pin 18650 2 cell



Mach 18650 Battery Shield V3 OEM



Khung xe robot car 4 bánh



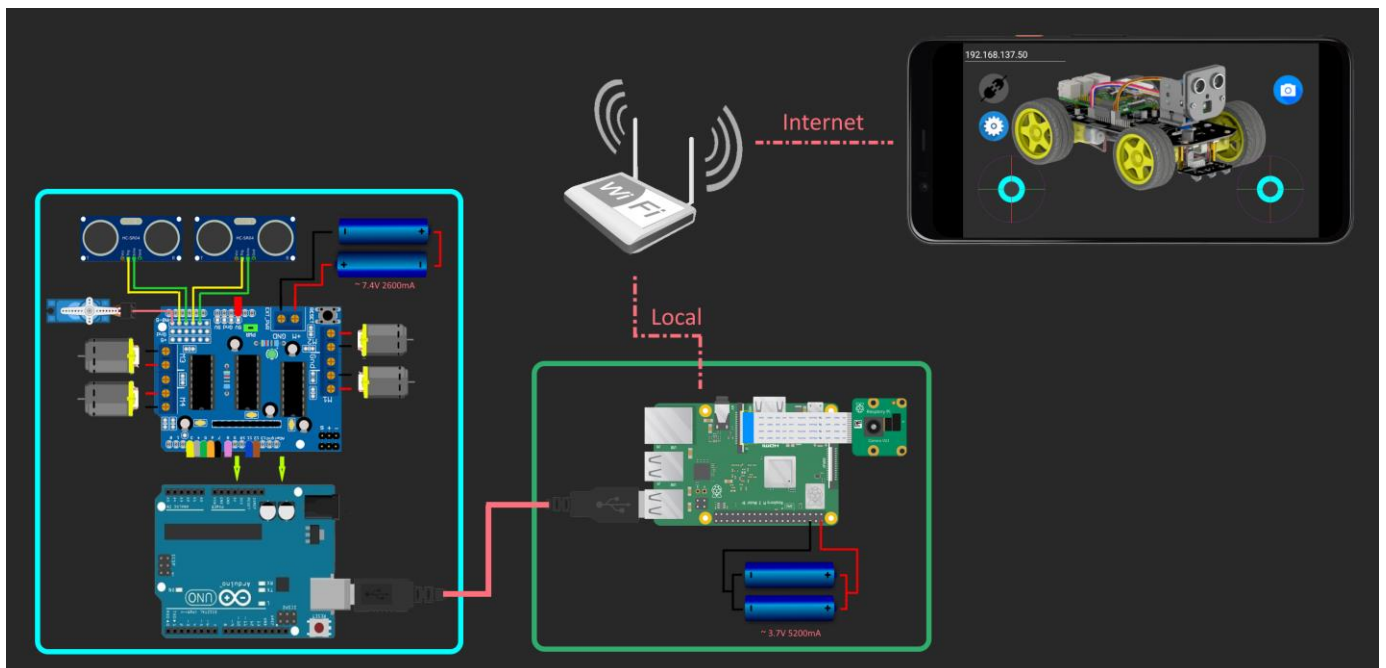
Dây Cắm Breadboard Đực Cái



Smartphone Android



II. SƠ ĐỒ HOẠT ĐỘNG



III. LẬP TRÌNH NHÚNG ĐIỀU KHIỂN ARDUINO

Ứng dụng FreeRTOS, Arduino sẽ thực hiện 3 nhiệm vụ chia thành 3 tác vụ chính:

1. Tác vụ điều khiển 4 Motor:

```
void controlMotor(void *p){
    while (true){
        xSemaphoreTake(motorSemaphore, portMAX_DELAY);
        switch (motor.getDirection()){
            case MIDDLE:
                motor.stop();
                break;
            case RIGHT:
                motor.runRight();
                break;
            case UP:
                motor.runUp();
                break;
            case LEFT:
                motor.runLeft();
                break;
            case DOWN:
                motor.runDown();
                break;
        }
    }
}
```

2. Tác vụ điều khiển Servo Camera:

```
void controlServo(void *p){
    while (true){
        xSemaphoreTake(servoSemaphore, portMAX_DELAY);
        servo.write();
    }
}
```

3. Tác vụ đọc cảm biến siêu âm HC-SR04, đọc dữ liệu từ Serial và cảnh báo vật cản:

```
void readData(void* p){
    while(true){
        if(sensor.alertFront() && motor.getDirection() == UP){
            Serial.println("0");
            motor.setDirection(MIDDLE);
            xSemaphoreGive(motorSemaphore);
            vTaskDelay(pdMS_TO_TICKS(50));
        }else if(sensor.alertBottom() && motor.getDirection() == DOWN){
            Serial.println("1");
            motor.setDirection(MIDDLE);
            xSemaphoreGive(motorSemaphore);
            vTaskDelay(pdMS_TO_TICKS(50));
        }

        if (Serial.available() > 0) {
            data_server = Serial.readStringUntil('\n');
            if(data_server.indexOf("MOTOR") != -1){
                str_number = &data_server[data_server.indexOf("#") + 1];
                data_number = str_number.toInt();
                task_delay = 50;
            }
        }
    }
}
```


IV. XÂY DỰNG SERVER CHO RASBERRY PI

Server được xây dựng dựa trên mô hình Python Socket TCP và sẽ đảm nhận các chức năng sau:

1. Đọc dữ liệu từ Client(Android):

```
def readData1(self):
    try:
        try:
            self.connection1,self.client_address1 = self.server_socket1.accept()
            print("Sender1 connection successful!")
        except:
            print("Sender1 connect failed!")
            self.server_socket1.close()
        while True:
            try:
                AllData=self.connection1.recv(1024).decode('utf-8')
                self.alert()
            except:
                break
            if(AllData):
                AllData = AllData[2:]
                if(AllData == "CLIENT#0"):
                    self.reset()
                else:
                    self.tcp_Flag = True
                    print(AllData)
                    self.ser.write(str(AllData + '\n').encode('utf-8'))
                    self.tcp_Flag = False

    except Exception as e:
        print(e)
    print("Stop Server")
    self.stopTcpServer()
```

```
def readData2(self):
    try:
        try:
            self.connection2,self.client_address2 = self.server_socket2.accept()
            print("Sender2 connection successful!")
        except:
            print("Sender2 connect failed!")
            self.server_socket2.close()
        while True:
            try:
                AllData=self.connection2.recv(1024).decode('utf-8')
            except:
                break
            if(AllData):
                AllData = AllData[2:]
                if(AllData == "CLIENT#0"):
                    self.reset()
                else:
                    print(AllData)
                    if(not self.tcp_Flag):
                        self.ser.write(str(AllData + '\n').encode('utf-8'))
            except Exception as e:
                print(e)
    print("Stop Server")
    self.stopTcpServer()
```

2. Phát video stream lên Website:

```
class StreamingHandler(server.BaseHTTPRequestHandler):
    def do_GET(self):
        if self.path == '/':
            self.send_response(301)
            self.send_header('Location', '/index.html')
            self.end_headers()
        elif self.path == '/index.html':
            content = PAGE.encode('utf-8')
            self.send_response(200)
            self.send_header('Content-Type', 'text/html')
            self.send_header('Content-Length', len(content))
            self.end_headers()
            self.wfile.write(content)
        elif self.path == '/stream.mjpg':
            self.send_response(200)
            self.send_header('Age', 0)
            self.send_header('Cache-Control', 'no-cache, private')
            self.send_header('Pragma', 'no-cache')
            self.send_header('Content-Type', 'multipart/x-mixed-replace; boundary=FRAME')
            self.end_headers()
            try:
                while True:
                    with output.condition:
                        output.condition.wait()
                        frame = output.frame
                    self.wfile.write(b'--FRAME\r\n')
                    self.send_header('Content-Type', 'image/jpeg')
                    self.send_header('Content-Length', len(frame))
                    self.end_headers()
                    self.wfile.write(frame)
                    self.wfile.write(b'\r\n')
            except Exception as e:
                logging.warning(
                    'Removed streaming client %s: %s',
```

3. Truyền dữ liệu điều khiển và nhận cảnh báo thông qua cổng Serial:

```
while True:
    try:
        AllData=self.connection1.recv(1024).decode('utf-8')
        self.alert()
    except:
        break
    if(AllData):
        AllData = AllData[2:]
        if(AllData == "CLIENT#0"):
            self.reset()
        else:
            self.tcp_Flag = True
            print(AllData)
            self.ser.write(str(AllData + '\n').encode('utf-8'))
            self.tcp_Flag = False
```

4. Hú còi khi có vật cản:

```
class Buzzer:
    def run(self,command):
        if command:
            GPIO.output(Buzzer_Pin,True)
        else:
            GPIO.output(Buzzer_Pin,False)
```

Start Server:

```
PROBLEMS  OUTPUT  TERMINAL  PORTS  DEBUG CONSOLE
pi@developer:~/Projects/Server $ python3 Server.py
Server running on 192.168.137.50!
█
```

V. PHÁT TRIỂN ỨNG DỤNG ANDROID

Ứng dụng Android được phát triển bằng ngôn ngữ lập trình Java và được tích hợp thêm thư viện Tensorflow để nhận diện hình ảnh. Ứng dụng có các chức năng:

1. Connect, Disconnect tới Server:

```
public void doConnecting(View v){
    final String HOST = txtIPAddress.getText().toString();
    final int PORT_SENDER = 3000;
    final int PORT_READER = 5000;
    final int VIDEO_PORT = 8000;

    if(!isConnecting){
        client = new Client(HOST, PORT_SENDER, PORT_READER, VIDEO_PORT);
        thread = new Thread(client);
        thread.start();
    }else{
        isConnecting = false;
    }
}
```

2. Chỉnh tốc độ chạy cho xe:

```
public void doSetting(View v) {
    showDialog(builder);
}
```

3. Chụp ảnh môi trường:

```
public void doCamera(View v){
    verifyStoragePermission( activity, this);
    if(takeScreenShot(mWebView, fileName: "smartcar")!=null){
        Toast.makeText( context: this, text: "Save Image", Toast.LENGTH_SHORT).show();
    }
}
```

4. Phát video stream:

```
public void start(){
    try {
        sender1 = new Socket(host, port_sender);
        dos1 = new DataOutputStream(sender1.getOutputStream());
        sender2 = new Socket(host, port_reader);
        dos2 = new DataOutputStream(sender2.getOutputStream());
        handler.post(new Runnable() {
            @Override
            public void run() {
                String url = "http://" + host + ":" + video_port + "/index.html";
                mWebView.loadUrl(url);
            }
        });
        isConnecting = true;
    } catch (UnknownHostException e) {
        e.printStackTrace();
    } catch (IOException e) {
        e.printStackTrace();
    } catch (Exception e){
        e.printStackTrace();
    }
}
```

5. Nhận diện hình ảnh:

```
protected void processImage(Bitmap rgbFrameBitmap) {
    MainActivity.computingDetection = true;

    trackingOverlay.postInvalidate();
    final Canvas canvas = new Canvas(croppedBitmap);
    canvas.drawBitmap(rgbFrameBitmap, frameToCropTransform, paint: null);

    TensorImage image = TensorImage.fromBitmap(croppedBitmap);
    Outputs outputs = model.process(image);

    final List<DetectionResult> results = outputs.getDetectionResultList();

    final Paint paint = new Paint();
    paint.setColor(Color.RED);
    paint.setStyle(Style.STROKE);
    paint.setStrokeWidth(2.0f);

    final List<DetectionResult> mappedRecognitions =
        new ArrayList<>();
```

```
    for (final DetectionResult result : results) {
        final RectF location = result.getLocationAsRectF();
        if (location != null && result.getScoreAsFloat() >= MINIMUM_CONFIDENCE_TF_OD_API) {
            canvas.drawRect(location, paint);

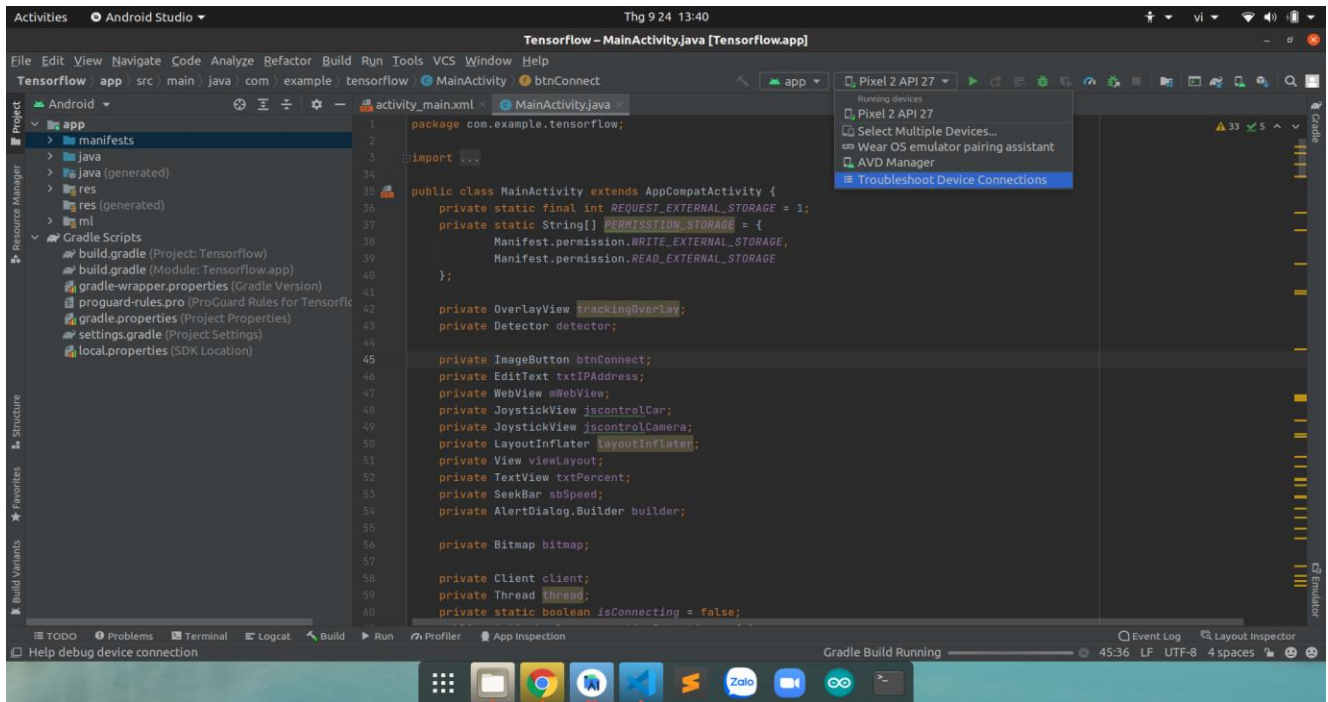
            cropToFrameTransform.mapRect(location);

            mappedRecognitions.add(result);
        }
    }

    tracker.trackResults(mappedRecognitions);
    trackingOverlay.postInvalidate();

    MainActivity.computingDetection = false;
}
```


Cài đặt ứng dụng cho android:




VI. DOWNLOAD CODE

Open terminal: `git clone https://github.com/ChuDuc1997/FPVSmartCar.git`

```
duc@developer:~/Documents$ git clone https://github.com/ChuDuc1997/Project.git
Cloning into 'Project'...
remote: Enumerating objects: 100, done.
remote: Counting objects: 100% (100/100), done.
remote: Compressing objects: 100% (76/76), done.
remote: Total 100 (delta 4), reused 100 (delta 4), pack-reused 0
Receiving objects: 100% (100/100), 3.24 MiB | 1.73 MiB/s, done.
Resolving deltas: 100% (4/4), done.
```


Android


Arduino


Server

VII. ƯU ĐIỂM, NHƯỢC ĐIỂM VÀ HƯỚNG PHÁT TRIỂN

1. Ưu điểm:

- Điều khiển real time tốt
- Có thể phân tích được đối tượng
- Có thể làm robot thăm dò, thu thập dữ liệu môi trường

2. Nhược điểm:

- Chỉ hoạt động được trong phạm vi mạng cục bộ
- Chưa tận dụng được dữ liệu video, image
- Mô hình tf lite còn ít dữ liệu
- Bảo mật kém

3. Hướng phát triển:

- Lưu trữ dữ liệu video, image lên Cloud
- Xây dựng mô hình tf lite riêng từ dữ liệu video, image
- Xây dựng Server trung gian để truyền nhận dữ liệu hiệu quả và quy mô hơn

TÀI LIỆU THAM KHẢO

1. <https://github.com/tensorflow/examples.git>
2. https://github.com/Freenove/Freenove_4WD_Smart_Car_Kit_for_Raspberry_Pi.git
3. https://hub.tensorflow.google.cn/tensorflow/lite-model/ssd_mobilenet_v1/1/metadata/2
4. <https://codelabs.developers.google.com/codelabs/recognize-flowers-with-tensorflow-on-android/#0>