

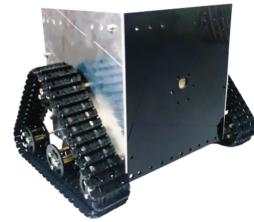
ROS2 Foxy



PS4 Joystick



WALL-E



ESP32 DEVKIT V1



Ubuntu 20.04

 照祥同学

# 17-ESP32\_ROS2\_with\_WALL-E

URL: [https://github.com/ZhaoXiangBox/esp32\\_ros2\\_robot](https://github.com/ZhaoXiangBox/esp32_ros2_robot)

Videos from Bilibili 照祥同学: [第十七节：搭建 ESP32 ROS2 的履带机器人](#)

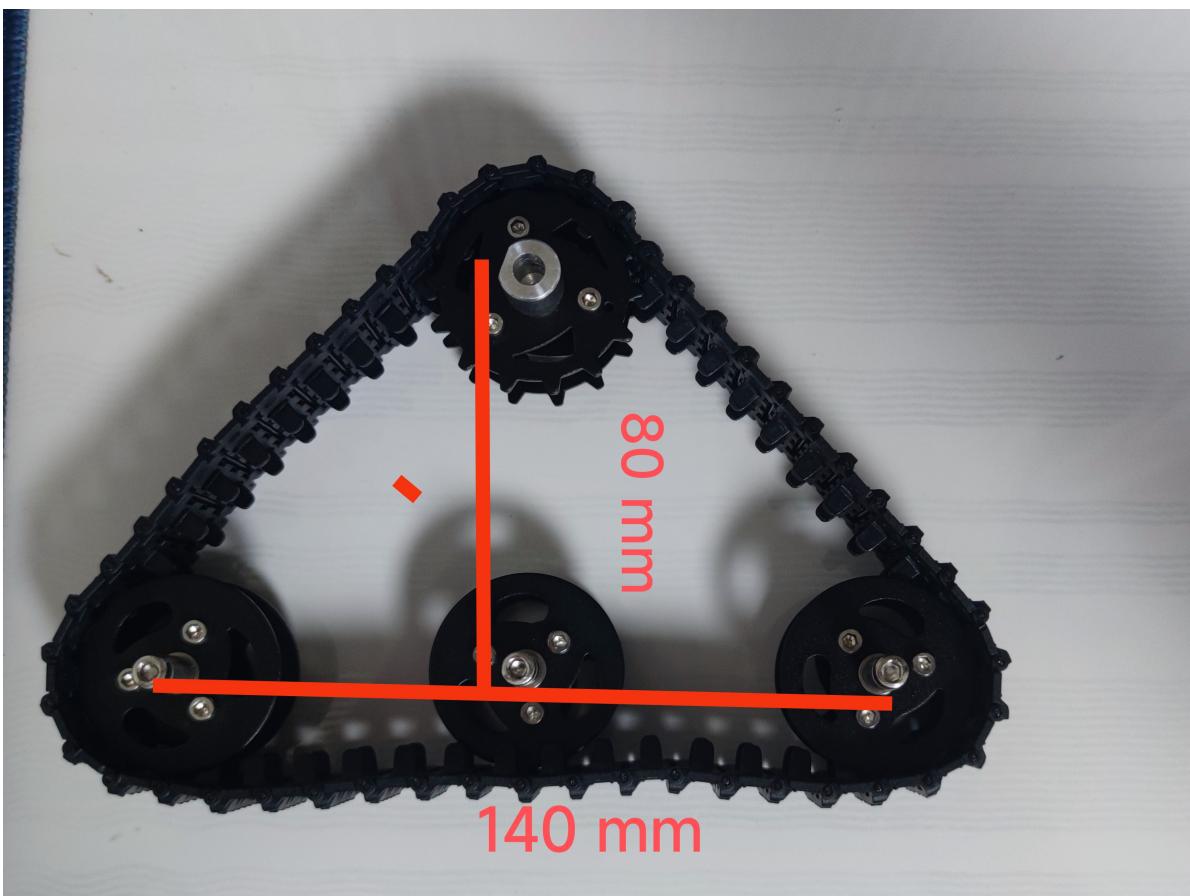
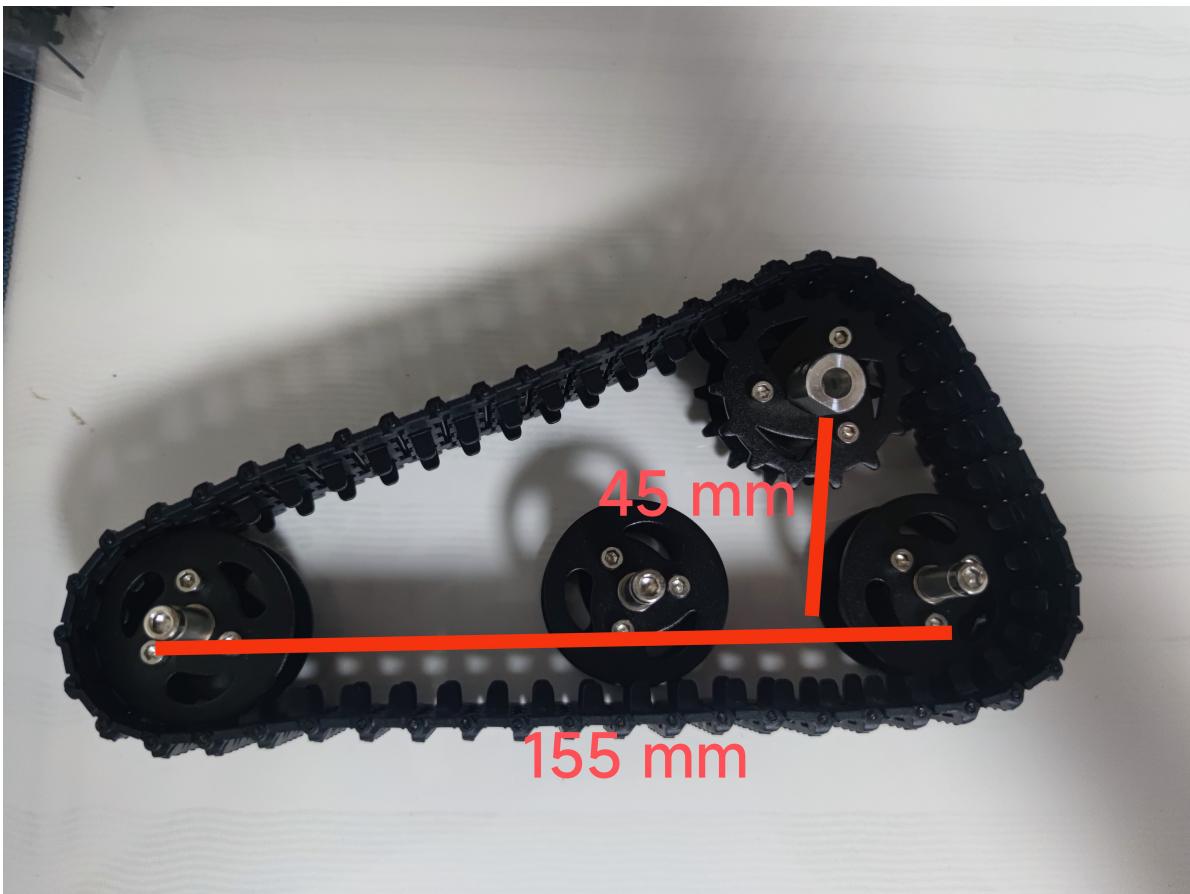
---

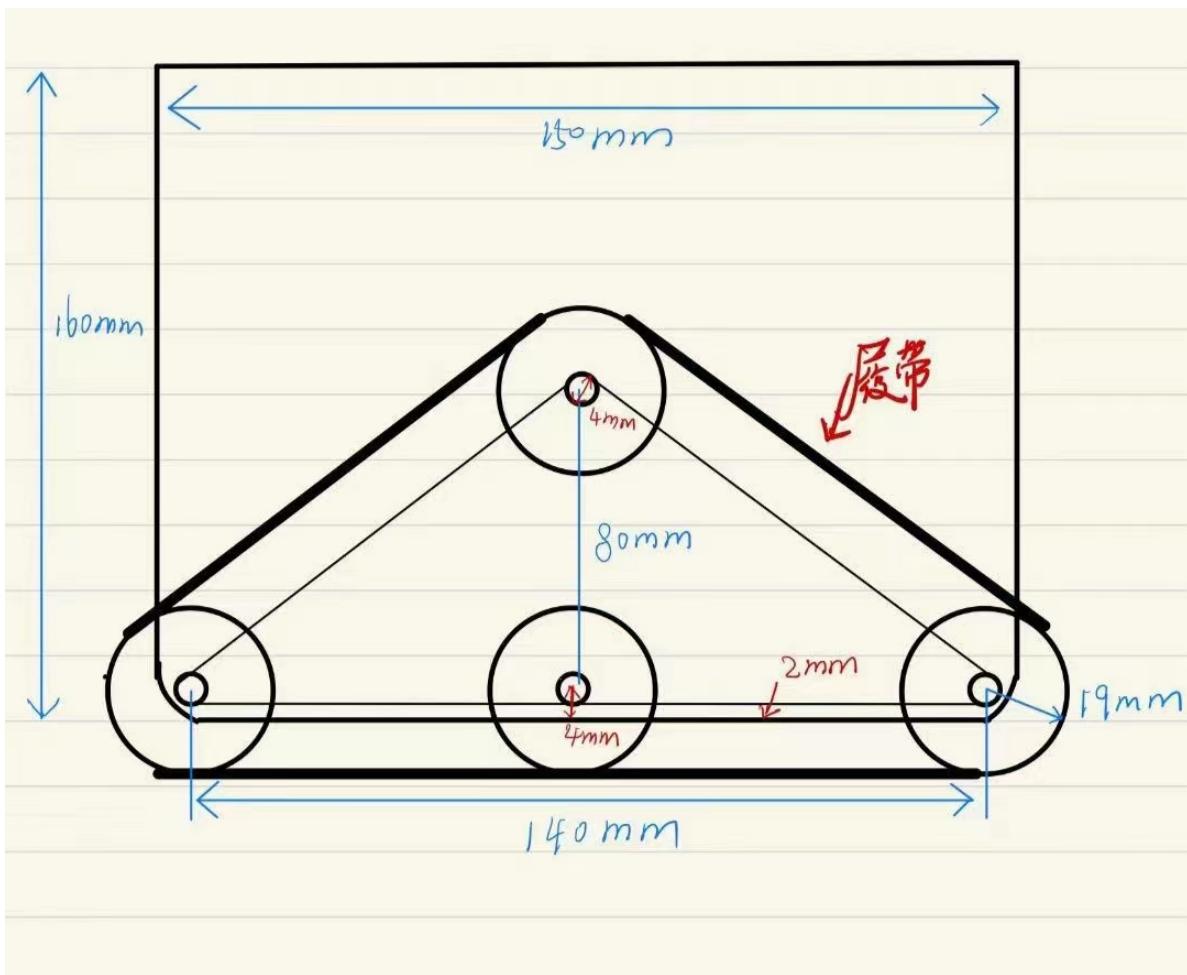
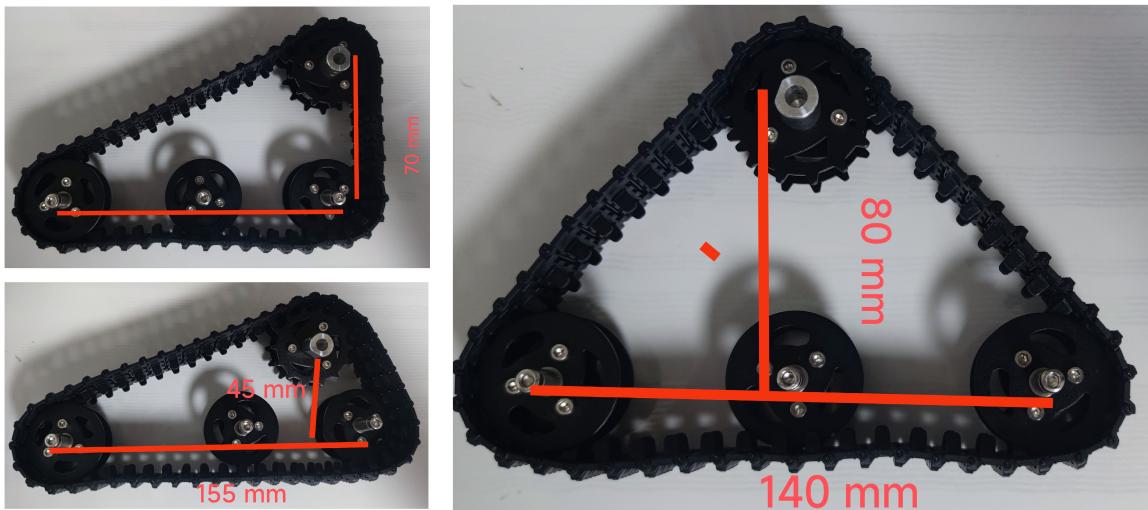
目录:

1. 装配一辆差速履带车;
  2. 差速底盘运动学解算;
- 

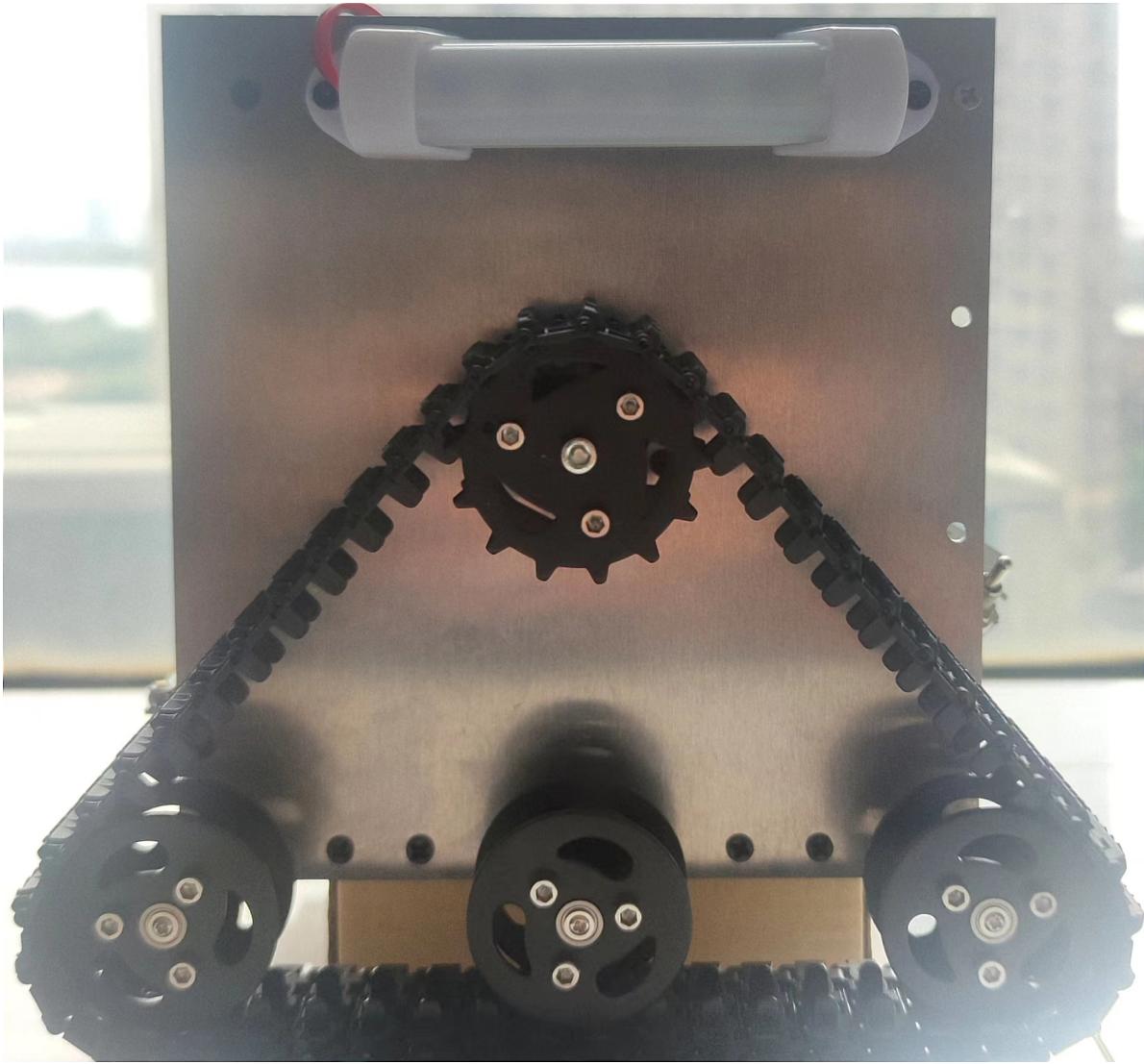
## 1. 装配一辆差速履带车

作者计算机专业，非机械设计科班，为了做这个外形像WALL-E的履带机器人，参考了很多设计并结合成本考虑，在平板中绘制了构想图：





后找淘宝商家制作，商家要求提供 .dwg 格式的设计文件，遂去下载学习使用fusion 360绘制相关图纸，最后的成品如下：



因为履带轮向内侧受力，所以选择的是 2mm 厚度的铝板，强度刚好满足要求，形变小，履带不掉下来，价格实惠。两块履带模组通过6根直径3mm的铜柱连接固定，就完成了WALL-E机器人的主题框架。



当前章节只讨论 `ESP32_ROS2_Robot` 控制的实现，所以未有展示WALL-E的 外框架、眼睛、手臂的实现细节，后续会放到一个新的专栏中逐步讲解；

## 2. 差速底盘运动学解算

通过机器人运动形式来看，本章节是差速运动，通过控制两个驱动电机转速实现机器人前进或者转弯；相关通讯细节仍然是采用的本专栏中使用的：`micro_ros Arduino ESP32` 固件为基础开发的，`ESP32` 通过串口或者 `WIFI` 的方式和安装有 `ROS2 Foxy` 环境的电脑进行通讯，实现 `ROS2` 层面的控制；

因此，我们仅需要将 `ROS2` 层面基于质点发送的2D线速度和角速度解算为两个电机的速度即可。

电机参数：

```
***** 车轮配置 *****  
采用中断方式捕获电机的霍尔脉冲，并且使用的是边沿触发方式，所以电机的编码值计算方法如下：  
encoder = (边沿触发) 2 × 霍尔编码器相数量 (如: 2) × 霍尔编码器线束 (如 13 ) × 电机减速比 (如: 30)  
  
更新车轮间距、车轮直径等参数后，需要自行核对  
*****  
*****/  
double encoderresolution = 1320.0; // 编码器输出脉冲数/圈 2*2*11*30 = 1320  
double wheel_diameter = 0.044; // 轮胎直径 m  
double D = 0.205 ; // 左右履带间距 205 mm
```

ROS2 层面的速度话题订阅：

```
// 订阅速度话题 cmd_vel 的回调函数。
void subs_Twist_callback(const void *msgin) {
    const geometry_msgs__msg__Twist *msg = (const geometry_msgs__msg__Twist
*)msgin;
    double Velocity = msg->linear.x;
    double Angular = msg->angular.z;

    Twist_to_pluse(Velocity,Angular);
    // Serial.println("Recv data");
    lastMotorCommand = millis(); // 记录每一次订阅到 topic 的时间。
}
```

差速运动的速度解算：

```
// 车辆运动学中心的线角速度 -----> 每个电机和舵机需要执行的目标脉冲数
void Twist_to_pluse(double linear_vel, double angular_vel)
{
    // 差速底盘运动学解算： Twist topic -----> each wheel velocity.
    double V_left = linear_vel - angular_vel * ( D / 2.0);
    double V_right = linear_vel + angular_vel * ( D / 2.0);

    // 每个电机在 PID 执行一次所期望的脉冲数
    Twist_temp.left_pulse_per_interval = (int) ( V_left * (encoderresolution /
                                                        (PI * wheel_diameter)) ) /
    pid_rate;
    Twist_temp.right_pulse_per_interval = (int) ( V_right * (encoderresolution /
                                                               (PI * wheel_diameter)) ) /
    pid_rate;
    // 交给PID控制器去计算并执行
    setTargetTicksPerFrame(Twist_temp.left_pulse_per_interval ,
                           Twist_temp.right_pulse_per_interval);
}
```

---

update by zhaoxiangli 2023.10.06