本科生课程考核试卷

（适用于课程论文、提交报告）

课　程：机器人基础　　课程代码：*ED21603

姓　名：莫湘渝　　学　号：20232373

专　业：明月科创实验班

授课教师：柏龙、孙园喜、陈晓红

授课时间：2025 学年 第一学期（第一周至第十七周）

学 生 成 绩：

课 程 成 绩

阅卷教师 (签名)_____

重庆大学国家卓越工程师学院 制

一、移动机器人概括（包含但不限于：应用领域总结分析；移动机器人的共性技术总结）

应用领域总结分析：

移动机器人的应用领域广泛，涉及工业制造、物流配送、服务行业、医疗辅助、农业自动化等多个方面。在工业领域，移动机器人主要负责自动化生产线的物料搬运、组件装配等任务。物流配送领域中，它们通过自动化的货物分拣和运输，提高效率并减少人力成本。服务行业中，移动机器人如服务机器人在酒店、餐厅等场所提供导引、送餐等服务。医疗辅助领域中，移动机器人协助进行手术或药品配送。农业自动化方面，移动机器人用于精准施肥、喷药等作业。

共性技术总结：

移动机器人的共性技术包括传感器技术、导航技术、控制技术等。传感器技术作为机器人的感知系统，负责收集环境信息。导航技术涉及路径规划和避障，而控制技术则涉及电机控制和运动控制。这些技术的发展，使得移动机器人能够更好地感知环境、分析数据并做出自主决策。

二、移动机器人控制技术分析（包含但不限于：移动底盘分析；电机特性分析；电机控制策略以及 pid 特性分析；嵌入式控制系统总结分析；传感系统总结分析）

移动底盘分析：

移动机器人的底盘设计对其性能至关重要。常见的底盘类型包括轮式、履带式和腿式等。轮式底盘具有结构简单、运动平稳的特点，适用于平坦地面履带式底盘则具有较好的越障能力和抓地力，适合复杂地形腿式底盘则模仿生物的行走方式，具有高度的灵活性和适应性。选择合适的底盘类型需要根据机器人的应用场景和任务需求来决定。轮式底盘因其灵活性和成本效益被广泛应用，而履带式底盘则适用于复杂地形如泥泞或不平地面。本次课程使用的麦克纳姆轮底盘，这是一种特殊的全方位移动底盘，由四个麦克纳姆轮组成，每个轮子由轮毂和多个沿一定角度排布的小辊子构成，这些小辊子与轮毂轴的夹角为 45°。这种设计

使得麦克纳姆轮底盘能够在水平面实现任意方向的移动，包括前后左右移动以及原地旋转，具有极高的灵活性。麦克纳姆轮的工作原理基于每个轮子产生的运动分解。当麦克纳姆轮转动时，每个轮子都会产生一个速度向量，这个速度向量可以分解为沿 X 轴和 Y 轴的分量，以及一个绕轮子和地面接触点的旋转分量。通过独立控制四个轮子的转速和转向，可以实现底盘在 X 轴和 Y 轴方向上的平移以及绕 Z 轴的旋转。例如以下为我们小组的逆解算，由平移旋转量得电机执行量：

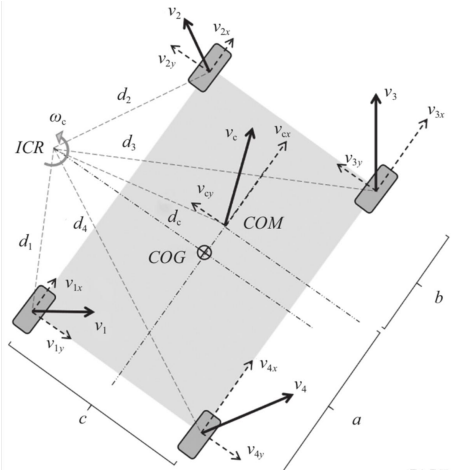
motor_six_speed = x_speed + y_speed + w_speed * RadiusofMotor

motor_seven_speed = x_speed - y_speed - w_speed * RadiusofMotor

motor_eight_speed = x_speed - y_speed + w_speed * RadiusofMotor

motor_nine_speed = x_speed + y_speed - w_speed * RadiusofMotor

麦克纳姆轮底盘的控制策略通常涉及到速度环和电流环的串级控制。通过精确控制电机的转速和转向，可以实现底盘的精确移动和旋转。在实际应用中，这种控制策略需要结合传感器反馈和 PID 控制算法来实现精确的运动控制。

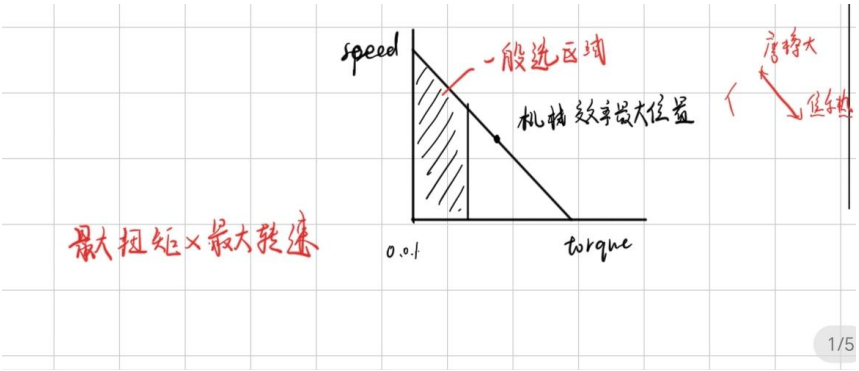


腿式

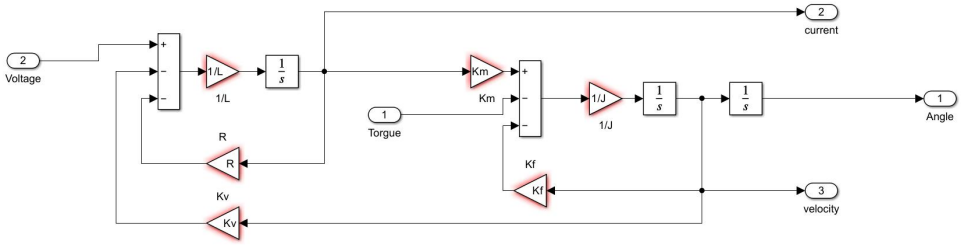
履带式

电机特性分析：

移动机器人电机的选型是一个涉及多个因素的决策过程，包括机器人的负载、速度、加速度、效率、控制需求以及成本等。电机的选择需要根据机器人的具体应用和负载要求来确定。电机的机械特性曲线描述了电机在不同负载下的转速和转矩关系，对于电机的控制和性能至关重要。通常先考虑电机是否满足最大转速，再看扭矩是否满足。关于电机的转矩、转速和效率特性，以下是一些关键点：电机的转矩和转速之间存在反比关系，即在功率一定的情况下，转速越高，转矩越低；转速越低，转矩越高 $T = \frac{9550 \times P}{n}$ 其中 T 是转矩（单位：N·m），P 是功率（单位：kW），n 是转速（单位：r/min）。启动转矩：转矩转速曲线中的起点即为启动转矩，通常是电机额定转矩的 1.5 倍左右。最大转矩：转矩转速曲线上的最高点即为最大转矩，通常是电机额定转矩的 2 倍左右。零转矩转速：当电机负载为零时，转速达到的最大值即为零转矩转速。额定转矩：转矩转速曲线上的水平线即为额定转矩，是电机能够持续输出的最大转矩。在负载电流较小时效率较低，输入的功率大部分消耗在空载损耗上；当负载电流增大时效率也增大，输入的功率大部分消耗在机械负载上；但当负载电流大到一定程度时铜损快速增大，此时效率又开始变小。



电机仿真模型 $V_m = iR + L\frac{di}{dt} + V_{emf}$



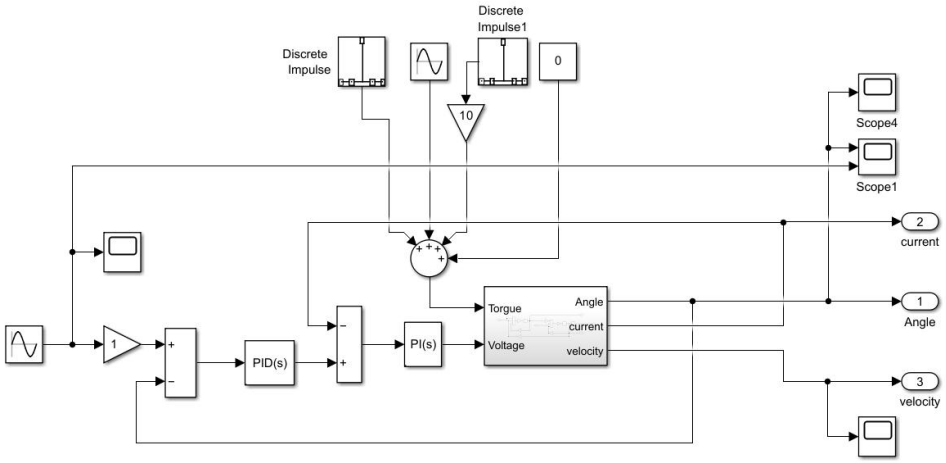
电机建模

本次课程使用为直流有刷电机，搭配减速箱，使用 arduino 引脚输出 pwm 波控制占空比输出进而控制电机。

电机控制策略以及 PID 特性分析：

电机控制策略中，PID 控制是一种经典的闭环控制算法，具有原理简单、易于实现、适用面广等优点。PID 控制器通过比例、积分、微分三个环节来调整电机的输入，以达到预期的输出。PID 参数的整定通常采用试凑法，根据系统响应调整比例、积分、微分系数以达到最佳控制效果。本次课程使用直流有刷电机，不含编码器，因此在项目中的反馈主要靠视觉模块的位置环差量控制输出，具体为操作为对于地面黑线，利用阈值二值化后，将所有阈值内的像素进行线性拟合，得到线性对象与竖直的角度，和距中线的距离，将其作为差量，分别计算得到旋转输出量和平移输出量，带入前面麦轮底盘逆解算公式，分别得到各个电机的理论输出量，该算法优势在于，即使电机没有输出理论输出量，也能通过计算，使运动收敛。我们使用增量式 PID，在调节 PID 控制器各个参数比例（P）、积分（I）和微分（D）时，我们了解到比例参数（Kp）决定了控制器对当前误差的响应强度，它能够快速减少误差，但过大的 Kp 可能导致系统振荡。积分参数（Ki）用于消除稳态误差，通过累积误差随时间的积分来调整控制输出，从而提高系统的精度，然而，过大的 Ki 可能会引起系统超调。微分参数（Kd）则是基于误差的变化率来调整控制输出，有助于减少系统的振荡和过冲，但过高的 Kd 可能会使系统对噪声过于敏感。在 PID 控制器的参数整定过程中，我们采取策略为先调整比例参数，然后是积分参数，最后是微分参数。这个过程根据小车系统的弯道响应来观察每个参数对小车性能的影响，并进行相应的调整。通过试凑法、临界比例法，我们先将比例参数 Kp 由小到大调到小车过弯后会快速振荡修正路线，然后适当降低 Kp，再调节 Kd 和 Ki，在项目中我们较多使用了 PD 控制器，Ki 参数很小，对于项目的应用环境来说足够了，实现了丝滑过弯以及局部飘移。此外，PID 参数的整定还需要考虑系统的采样时间，以确保控制器能够适应系统的动态特性。一般默认为最大采样频率，这里不做赘述。PID 控制器的调节目标是

实现快速响应、高精度和稳定性。一个良好的 PID 控制系统应该具有适当的衰减比，稳态误差趋近于 0，并且系统响应迅速。在其他应用中，PID 控制器的参数还需要根据具体系统的动态特性进行调整，以达到最佳的控制效果。



PID 仿真模型

嵌入式控制系统总结分析：

嵌入式控制系统是为特定设备量身打造的计算机系统，它们深植于设备内部，负责执行控制任务。这些系统以微处理器为核心，配备存储器、输入输出接口和专用软件，共同实现对设备的精确控制。它们能够在严格的时间限制内响应，确保操作的及时性和系统的安全性。在资源有限的条件下，如有限的处理器速度和内存，嵌入式控制系统通过精心设计，实现高效运行，同时保持高可靠性和稳定性，以适应各种环境挑战。

这些系统针对特定任务进行了优化，使得它们在执行特定功能时表现出高效率。软件与硬件的紧密结合，使得嵌入式控制系统能够在资源受限的情况下，实现复杂的控制逻辑。随着技术的进步，这些系统现在越来越多地具备网络连接功能，允许远程监控和控制，同时也面临着安全性的挑战，尤其是在关键应用领域。

良好的嵌入式系统设计注重维护和升级的便利性，通常采用模块化设计，以便于单个组件的更换或升级。开发这些系统需要专业的工具和环境，以及严格的测试和验证流程，确保它们在各种操作条件下都能稳定工作。嵌入式控制系统的广泛应用，如智能家居、物联网和自动驾驶汽车，展示了跨学科知识在设计和实现过程中的重要性。

嵌入式通讯是嵌入式系统中实现设备间数据交互的关键技术，涵盖了多种通

信协议和方式。常见的串行通信协议包括 UART，它是一种简单且成本低廉的异步通信方式，广泛应用于设备间的近距离通信。RS-232 和 RS-485 则是两种常用的串行通信标准，RS-232 支持双向通信，常用于 PC 与外部设备的连接，而 RS-485 支持多点通信，适用于长距离和高速数据传输。同步串行通信协议如 SPI 和 I2C 在嵌入式系统内部芯片间通信中应用广泛，SPI 具有高速传输的特点，而 I2C 支持多主设备通信，线缆少且传输速率高。总线通信协议如 CAN 总线在汽车和工业控制领域具有高可靠性和实时性。无线通信技术如 Zigbee、Wi-Fi 和蓝牙为嵌入式系统提供了灵活的通信方式，Zigbee 适用于智能家居和工业自动化，Wi-Fi 用于互联网接入和局域网建设，蓝牙则以高速传输和低功耗著称。网络通信协议如 TCP/IP 是互联网的基础，用于实现设备间的网络通信，而 Modbus 则广泛应用于工业控制系统中。不同的通信协议和技术各有其特点和适用场景，选择合适的协议需要综合考虑通信速度、传输距离、系统复杂度和成本等因素，以满足具体的应用需求.

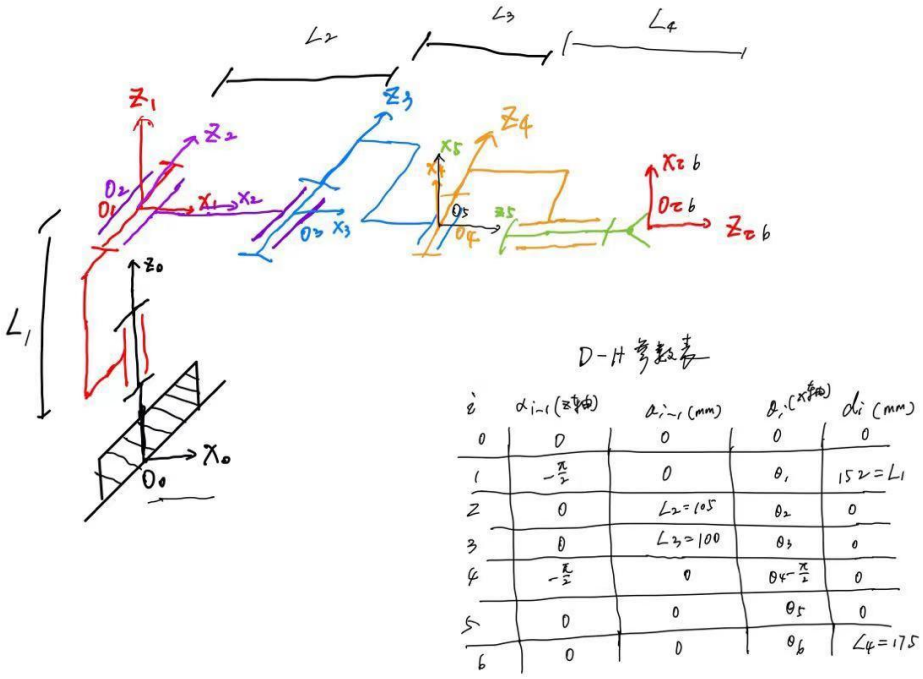
本次项目我们小组的嵌入式控制系统采用 arduino+openmv，openmv 不仅仅只作为视觉传感器，它同样能够控制底盘，在控制策略中，二者类似于大小脑，arduino 控制机械臂运动，读取超声波传感器返回距离，通过 PID 控制器计算后控制底盘前后运动，读取 openmv 控制信号，决策底盘控制权式 arduino 还是 openmv；openmv 读取 arduino 控制信号，当底盘控制权在 openmv 时识别路线，直接通过串口控制底盘巡线，同时，它在读取信号后知道何时开始识别物块，识别物料台，并传回相印信号。

传感系统总结分析：

传感系统是移动机器人感知环境的关键，包括视觉系统、深度学习、触觉传感器，距离传感器等。这些传感器为机器人提供关于环境的多维信息，是机器人进行导航和避障的基础。通过传感器数据，机器人能够实现对环境的精确感知和智能决策。视觉系统由光信号发生器、传感器、图像采集卡等组成，涉及图像采集、压缩编码及传输、图像增强、边缘检测、阈值分割、目标识别、三维重建等，几乎覆盖机器视觉的各个方面。超声波传感器因其独特的工作原理，在距离测量和物体检测方面表现出色。它们通过发射高频声波并接收反射波来计算与物体的距离，这一过程不受光线条件的影响，且能够在真空中工作，具有穿透性。然而，超声波传感器在软质材料的感测上可能存在精度问题，且对温度变化敏感，可能需要温度补偿来提高测量精度。本次项目主要运用的传感器为视觉，超声波。

三、执行机构分析（基于课上的机械臂传动方式与尺寸，包含但不限于：机械臂结构分析（机构简图）；机械臂正运动学模型；机械臂逆运动学求解）

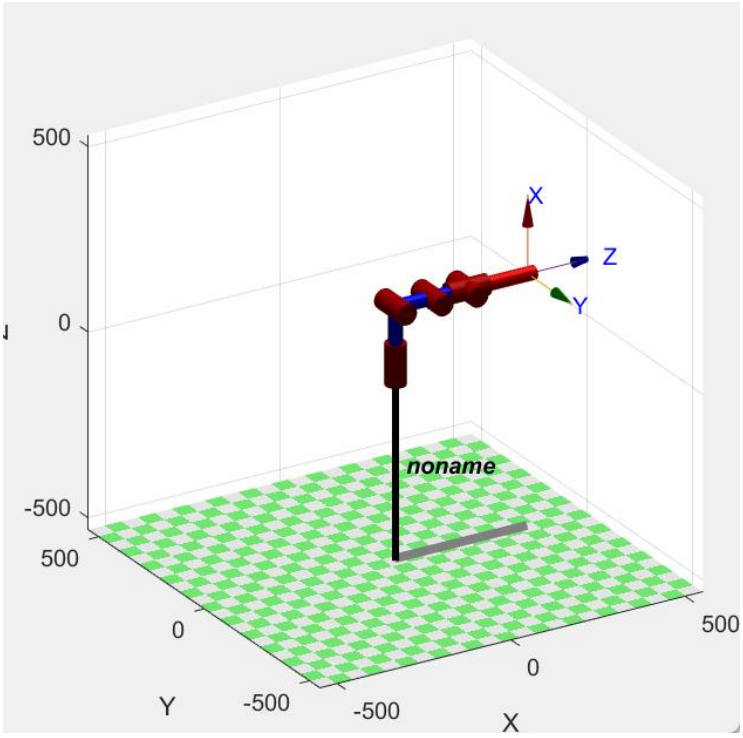
机构简图：



机构简图以及关节参数

正运动学模型：

```
L(1)=Link([ 0   d1   a1   alpha1],'qlim', jt_lts(1, :));%theta关节角 d
%aa=L(1).A(pi/6)%得对应关节变换齐次矩阵
L(2)=Link([ 0   d2   a2   alpha2],'qlim', jt_lts(2, :));
L(3)=Link([ 0   d3   a3   alpha3],'qlim', jt_lts(3, :));
L(4)=Link([ -90* radian1   d4   a4   alpha4],'qlim', jt_lts(4, :));
L(5)=Link([ 0   d5   a5   alpha5],'qlim', jt_lts(5, :));
L(6)=Link([ 0   d6   a6   alpha6],'qlim', jt_lts(6, :));

Six_link=SerialLink([L(1),L(2),L(3),L(4),L(5),L(6)]);
%Six_link=SerialLink([L(1),L(2),L(3),L(4),L(5)]);

Six_link.n;%转轴数量
Six_link.d;%连杆偏距
Six_link.links;%显示连杆信息的函数
%Six_link.plot([0,0,0,0,0,0]);%初始弧度
Six_link.plot([0,0,0,-90 * radian1,0,0]);%初始弧度
```



Robotool 简要建模

$$T_0^1=\begin{bmatrix}\cos\theta_1 & -\sin\theta_1 & 0 & 0\\ \sin\theta_1 & \cos\theta_1 & 0 & 0\\ 0 & 0 & 1 & 153\\ 0 & 0 & 0 & 1\end{bmatrix} \qquad T_1^2=\begin{bmatrix}\cos\theta_2 & 0 & -\sin\theta_2 & 0\\ \sin\theta_2 & 0 & \cos\theta_2 & 0\\ 0 & -1 & 0 & 0\\ 0 & 0 & 0 & 1\end{bmatrix}$$

$$T_2^3=\begin{bmatrix}\cos\theta_3 & -\sin\theta_3 & 0 & 105\cos\theta_3\\ \sin\theta_3 & \cos\theta_3 & 0 & 105\sin\theta_3\\ 0 & 0 & 1 & 0\\ 0 & 0 & 0 & 1\end{bmatrix} \qquad T_3^4=\begin{bmatrix}\cos\theta_4 & 0 & -\sin\theta_4 & 100\cos\theta_4\\ \sin\theta_4 & 0 & \cos\theta_4 & 100\sin\theta_4\\ 0 & -1 & 0 & 0\\ 0 & 0 & 0 & 1\end{bmatrix}$$

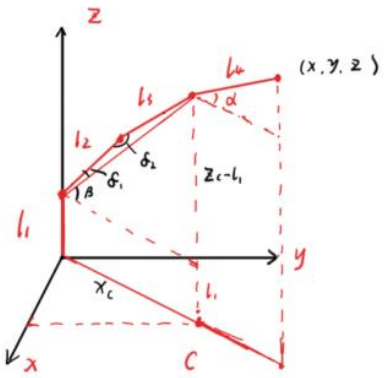
$$T_4^5=\begin{bmatrix}\cos\theta_5 & 0 & \sin\theta_5 & 0\\ \sin\theta_5 & 0 & -\cos\theta_5 & 0\\ 0 & 1 & 0 & 0\\ 0 & 0 & 0 & 1\end{bmatrix} \qquad T_5^6=\begin{bmatrix}\cos\theta_T & -\sin\theta_T & 0 & 0\\ \sin\theta_T & \cos\theta_T & 0 & 0\\ 0 & 0 & 1 & 172\\ 0 & 0 & 0 & 1\end{bmatrix}$$

变换矩阵

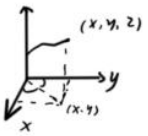
Matlab 建模如下：

```
T1=[cos(theta(1)), -sin(theta(1)), 0, a(1);
cos(alpha(1))*sin(theta(1)), cos(alpha(1))*cos(theta(1)), -sin(alpha(1)), -d(1)*sin(alpha(1));
sin(alpha(1))*sin(theta(1)), sin(alpha(1))*cos(theta(1)), cos(alpha(1)), d(1)*cos(alpha(1));
0, 0, 0, 1];
T2=[cos(theta(2)), -sin(theta(2)), 0, a(2);
cos(alpha(2))*sin(theta(2)), cos(alpha(2))*cos(theta(2)), -sin(alpha(2)), -d(2)*sin(alpha(2));
sin(alpha(2))*sin(theta(2)), sin(alpha(2))*cos(theta(2)), cos(alpha(2)), d(2)*cos(alpha(2));
0, 0, 0, 1];
T3=[cos(theta(3)), -sin(theta(3)), 0, a(3);
cos(alpha(3))*sin(theta(3)), cos(alpha(3))*cos(theta(3)), -sin(alpha(3)), -d(3)*sin(alpha(3));
sin(alpha(3))*sin(theta(3)), sin(alpha(3))*cos(theta(3)), cos(alpha(3)), d(3)*cos(alpha(3));
0, 0, 0, 1];
T4=[cos(theta(4)), -sin(theta(4)), 0, a(4);
cos(alpha(4))*sin(theta(4)), cos(alpha(4))*cos(theta(4)), -sin(alpha(4)), -d(4)*sin(alpha(4));
sin(alpha(4))*sin(theta(4)), sin(alpha(4))*cos(theta(4)), cos(alpha(4)), d(4)*cos(alpha(4));
0, 0, 0, 1];
T5=[cos(theta(5)), -sin(theta(5)), 0, a(5);
cos(alpha(5))*sin(theta(5)), cos(alpha(5))*cos(theta(5)), -sin(alpha(5)), -d(5)*sin(alpha(5));
sin(alpha(5))*sin(theta(5)), sin(alpha(5))*cos(theta(5)), cos(alpha(5)), d(5)*cos(alpha(5));
0, 0, 0, 1];
T6=[cos(theta(6)), -sin(theta(6)), 0, a(6);
cos(alpha(6))*sin(theta(6)), cos(alpha(6))*cos(theta(6)), -sin(alpha(6)), -d(6)*sin(alpha(6));
sin(alpha(6))*sin(theta(6)), sin(alpha(6))*cos(theta(6)), cos(alpha(6)), d(6)*cos(alpha(6));
0, 0, 0, 1];
T=T1*T2*T3*T4*T5*T6;
```

逆运动学解算，主要运用几何法进行逆运动学解算：



对于 θ_1，利用机械臂投影



$\tan\theta_1 = \frac{y}{x}$，$\theta_1 = \arctan\theta_1$

对于投影半径 r

$r = \sqrt{x^2+y^2}$

$x_c = r - \cos\alpha \cdot l_4$

$z_c = z - \sin\alpha \cdot l_4$

$\alpha \in (-90°, 90°)$

$(l_2-l_3)^2 < x_c^2 + (z_c-l_1)^2 < (l_2+l_3)^2$ ①

当 α 满足 ① 时；（代码中由 $-90°$～$90°$ 步进 $1°$ 检验）

将 x_c, z_c 代入下面得 θ_2, θ_3, θ_4

利用余弦公式

$$\cos\delta_1 = \frac{l_2^2 + [x_c^2 + (z_c-l_1)^2] - l_3^2}{2 \cdot l_2 \cdot \sqrt{x_c^2+(z_c-l_1)^2}}, \quad \delta_1 = \arccos\delta_1$$

$$\tan\beta = \frac{z_c-l_1}{x_c}, \quad \beta = \arctan\beta$$

$$\cos\delta_2 = \frac{l_2^2 + l_3^2 - \sqrt{x_c^2+(z_c-l_1)^2}}{2 \cdot l_2 \cdot l_3}, \quad \delta_2 = \arccos\delta_2$$

$\theta_2 = -(\beta + \delta_1)$

$\theta_3 = \pi - \delta_2$

$\theta_4 = -\theta_2 - \theta_3 - \alpha$

最后需要确保 θ_2, θ_3, θ_4 为实数

如下为代码编写：（含详细注释）

```
function result = calc_jt_agl(x, y, z)
    % 机械臂关节长度
    l1 = 152;
    l2 = 105;
    l3 = 100;
    l4 = 175;

    % 计算 theta1，通过反正切函数 atan2 根据末端位置的 x、y 坐标来确定
    theta1 = atan2(y, x);

    % 在 x-y 平面投影中的半径
    r = sqrt(x^2 + y^2);
    % 初始化找到解的标志
    found_solution = false;

    % 循环尝试不同的 a 值
    for a_deg = -90:90
        a = deg2rad(a_deg);
        % 计算 xc 和 zc
        xc = r - cos(a) * l4;
        zc = z - sin(a) * l4;

        % 计算 lac_sq
        lac_sq = xc^2 + (zc - l1)^2;

        % 检查是否在范围内
        if lac_sq > (l2 + l3)^2 || lac_sq < (l2 - l3)^2
            continue;

          end

          % 计算 jbac 和 jcac_prime
          jbac = acos((l2^2 + lac_sq - l3^2) / (2 * l2 * sqrt(lac_sq)));
          jcac_prime = atan2(zc - l1, xc);

          % 计算 theta2
          theta2 = -jbac - jcac_prime;

          % 计算 theta3
          theta3 = pi - acos((l2^2 + l3^2 - lac_sq) / (2 * l2 * l3));

          % 检查是否有虚数部分
          if isreal(theta2) && isreal(theta3)
              % 计算 theta4
              theta4 = - theta2 - theta3 - a;

              % 标记已找到解
              found_solution = true;

              break; % 结束循环
          end
      end

      if found_solution
          % 返回关节角度数组
          result = [0, rad2deg(theta1), rad2deg(theta2), rad2deg(theta3), rad2deg(theta4)-90, 0, 0];
```

```
        else
            disp('找不到合适的逆解');
            result = NaN(1, 7); % 返回 NaN 表示未找到解
        end
    end

    function angle=plt_calc(stPt,edPt)
        numPoints = 11;
        % 使用 linspace 生成插值点
        xValues = linspace(stPt(1), edPt(1), numPoints);
        yValues = linspace(stPt(2), edPt(2), numPoints);
        zValues = linspace(stPt(3), edPt(3), numPoints);

        angle = zeros(numPoints, 7);

        % 循环计算每个插值点的关节角度
        for i = 1:numPoints
            % 调用函数计算关节角度
            angle(i, :) = calc_jt_agl(xValues(i), yValues(i), zValues(i));
        end

    end
```

可以计算出最优逆解，各个关节的执行角度

四、综合实践环节报告

1．任务分解

根据综合实践环节项目的要求，我们将任务分解为一下几点：

1. 小车循迹任务 2.视觉模块识别物块 3.抓取物块

4.放置物块 5.设计机械臂执笔装置 6.机械臂正逆解算写字

2．任务分工

我们小组的任务分工如下：

文哲浩：主要负责 Arduino 控制板代码开发与编写，通信问题的解决、小车拾取物块之后寻找地点的算法问题等，找到解决策略。根据任务要求进行整体代码框架搭建，同时在开发过程中进行硬件调试，寻找并解决问题。并且设计了机械臂的执笔装置。

莫湘渝：主要负责 OpenMV 视觉模块的代码开发，视觉代码执行框架的搭建，以及通过机械臂逆解算确定机械臂拾取、放置物块的关节舵机位置，解算控制地盘，基于视觉解决小车循迹、物块识别以及地点识别等问题，在开发过程中进行硬件调试，寻找并解决问题，同时优化了机械臂执笔装置。

王彦斌：主要负责机械臂写字任务以及机械臂执笔装置的安装，负责写字部分机械臂运动学解算，调整关节参数，实现写字。参与 openmv 循迹部分 pid 调试，参与整体硬件部分接线及问题排查，在开发过程优化小车各项任务的执行效果。

3. 实现过程（包含但不限于：任务描述；技术路径与策略；核心程序逻辑；实现的实际效果）

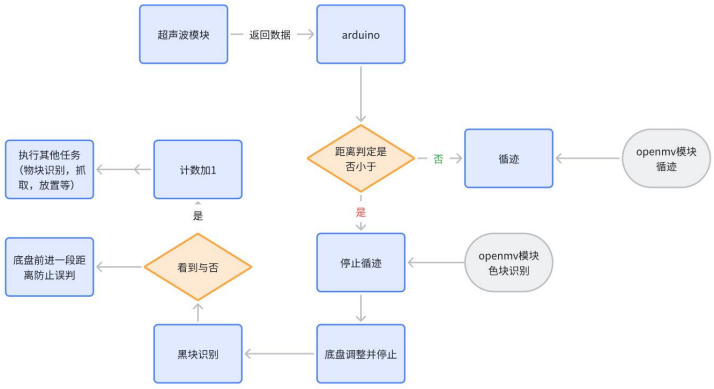
任务描述：

该任务综合性很高，底盘控制包含有寻迹，械臂控制包含有物块抓取和写字，综合运用了多种传感器，例如视觉，超声波测距，红外传感。小组需要先进行寻迹到达指定位置识别台，在识别台利用视觉识别物块颜色，并将物块抓取，最后沿着既定轨迹寻迹到相应的放置台放置物块，识别台和放置台前有障碍物，可利用超声波进行小车定位。完成三次不定顺序，不同颜色物块放置后，回到识别台，走入另一个轨迹，到达写字台终点，控制机械臂在写字台上写字，写字内容对应不同颜色物块顺序，写字完成后任务完成。

技术路径与策略以及程序逻辑：

本次项目我们小组的采用 arduino+openmv 控制策略，openmv 不仅仅只作为视觉传感器，它同样能够控制底盘，在控制策略中，二者类似于大小脑，arduino 控制机械臂运动，读取超声波传感器返回距离，决策控制信号，当超声波测距后达到某一阈值，则通过 PID 控制器控制底盘前后运动，使其停在指定位置（识别台，放置台，写字台前）。同时读取 openmv 控制信号，决策底盘控制权交给 arduino 还是 openmv；openmv 读取 arduino 控制信号，当底盘控制权在 openmv 时识别路线，直接通过串口控制底盘巡线，同时，它在读取信号后知道何时开始识别物块，识别物料台，并传回相应信号。完整逻辑为当开启电源时，超声波同时开启，距离未达到限制前，arduino 给 openmv 相应引脚高电平信号，底盘控制权交给 openmv，定速巡线，直至到达识别台前，超声波第一次检测到障碍物时，底盘控制权回给 arduino，PID 控制小车停的位置处于正中央识别台或者放置台前面，开启另一种巡线（本次巡线无前进速度，只调整小车是否在轨迹中间）执行完后，arduino 给 openmv 相应引脚高电平信号，openmv 开启物块识别模式，同时 arduino 控制机械臂按照给定关节角度执行，将机械臂旋转到正对物料台，并微调角度直到色块在画幅中心，openmv 返回识别到的物块颜色以及色块在画幅中心与否，满足条件在中心一定范围时，openmv 传递电平信号使 arduino 执行

相印抓取程序，然后又再次开启巡线，直到下次障碍物，注意当停止但未识别到黑色物块，则前进一段固定距离防止误识别同一个障碍，再次开启巡线。对于放置台的识别，我们组采取了最稳妥的颜色识别加计数统计（当检测到障碍物，并且机械臂旋转到相应位置能识别到黑色放置台则计数加一），在程序里有相应放置台的计数差别，比如蓝色是 1，红色是 3，以此知道小车位于什么颜色放置台前，完成放置任务（放置任务调整小车底盘与前文提到停在识别台前逻辑一致），每完成一次任务，也会有相应的计数，当完成三种颜色识别抓取放置后，回到识别台，arduino 控制底盘执行原地旋转，再开启巡线到写字台前，最后超声波控制底盘距离，机械臂执行写字程序（由最后一个抓取任务决定写字内容）。一共会进行 3 圈的物块抓取和放置。效率较低但比较稳定。（arduino 源码过长，且逻辑已讲清楚，不放源码）



大概流程图

```python
def object_recognition_state_machine(img):


    blobs_red = img.find_blobs([red_threshold], x_stride=50, y_stride=50, merge=True)
    blobs_green = img.find_blobs([green_threshold], x_stride=50, y_stride=50, merge=True)
    blobs_blue = img.find_blobs([blue_threshold], x_stride=50, y_stride=50, merge=True)

    # 重置引脚状态
    center_pin.high()
    color_pin_1.high()#1-red,2-green,都为低则bluew
    color_pin_2.high()

    # 检测红色
    if blobs_red:
        largest_blob_red = max(blobs_red, key=lambda b: b.pixels())
        img.draw_rectangle(largest_blob_red.rect(),color=(255, 0, 0))
        # img.draw_cross(largest_blob_red.cx(), largest_blob_red.cy())
        if largest_blob_red.pixels() > 600:
            # color_pin_1.low()  # 红色指示
            p9.low()
            p8.high()
        # 计算x，y差量
        x_diff_red = largest_blob_red.cx() - img.width() // 2

        if abs(x_diff_red) <= 16:
            center_pin.low()  # 物块在中心偏差8以内
```

```
if blobs_green:
    largest_blob_green = max(blobs_green, key=lambda b: b.pixels())
    img.draw_rectangle(largest_blob_green.rect(), color=(0, 255, 0))
    # img.draw_cross(largest_blob_green.cx(), largest_blob_green.cy(), color=(0, 255, 0))
    if largest_blob_green.pixels() > 600:
        img.draw_rectangle(largest_blob_green.rect(), color=(0, 255, 0))
        # color_pin_2.low()  # 绿色指示
        # color_pin_1.high()
        p9.high()
        p8.low()
    # 计算x, y差量
    x_diff_green = largest_blob_green.cx() - img.width() // 2

    if abs(x_diff_green) <= 16:
        center_pin.low()  # 物块在中心偏差8以内

# 检测蓝色
if blobs_blue:
    blob_blue1=max(blobs_blue, key=lambda b: b.pixels())
    img.draw_rectangle(blob_blue1.rect(), color=(0, 0, 255))
    # img.draw_cross(blob_blue1.cx(), blob_blue1.cy(), color=(0, 0, 255))
    # 计算x, y差量
    if blob_blue1.pixels() > 600:
        # color_pin_1.low()
        # color_pin_2.low()
        p9.low()
        p8.low()
    x_diff_blue = blob_blue1.cx() - img.width() // 2
    if abs(x_diff_blue) <= 16   :
        center_pin.low()  # 物块在中心偏差8以内
```

物块识别代码

对于视觉循迹：视觉模块的位置环差量控制输出，具体为操作为对于地面黑线，利用阈值二值化后，将所有阈值内的像素进行线性拟合，得到线性对象与竖直的角度，和距中线的距离，将其作为差量，分别计算得到旋转输出量和平移输出量，带入前面麦轮底盘逆解算公式，分别得到各个电机的理论输出量。

```
def line_following_state_machine(img):
    move=400
    rho_output = 0  # 初始化rho_output
    theta_output = 0  # 初始化theta_output
    img = img.binary([line_THRESHOLD])
    line = img.get_regression([(100, 100)], robust=True)
    if (line):
        rho_err = abs(line.rho()) - img.width() / 2
        # center=(line.x1())-img.width() / 2
        if line.theta() > 90:
            theta_err = line.theta() - 180
        else:
            theta_err = line.theta()
        img.draw_line(line.line(), color=127)
        # print(rho_err, line.magnitude(), rho_err)
        if line.magnitude() > 8:
            if abs(theta_err)>25:
                # move=350
                if theta_err>25:
                    theta_err=25
                if theta_err<-25:
                    theta_err=-25
            # else: move=350
            rho_output =round( -rho_pid.get_pid(rho_err, 1))
            theta_output = round(theta_pid.get_pid(theta_err, 1))
            # output = rho_output + theta_output

        else:
            # output=50
            theta_output=100
    else:
        # output=100
        theta_output=150
        pass
    FullDirection(move,rho_output ,theta_output)
```

对于机械臂逆解算：根据给定写字的内容，进行笔画拆分，独立出三个笔画，利用前面的程序分别进行逆解算，利用直线插补法，结合舵机执行时的执行速度，选取五个点作为插值个数，得到计算结果之后，将其与舵机执行角度进行转换，然后不断调整角度，最后，根据不同的情况组合笔画，封装成不同的函数，供执行。抓取任务和放置任务，根据中线到识别台距离和高度确定机械臂的末位，逆解算后，再根据实际情况调整，抓取，放置效果较好。

对于增量式 PID 公式如下：

$$\Delta u(k) = u(k) - u(k-1)$$

$$= K_P[e(k) - e(k-1)] + K_I e(k) + K_D[e(k) - 2e(k-1) + e(k-2)]$$

```
from pyb import millis
from math import pi, isnan

class PID:
    _kp = _ki = _kd = _integrator = _imax = 0
    _last_error = _last_derivative = _last_t = 0
    _RC = 1/(2 * pi * 20)
    def __init__(self, p=0, i=0, d=0, imax=0):
        self._kp = float(p)
        self._ki = float(i)
        self._kd = float(d)
        self._imax = abs(imax)
        self._last_derivative = float('nan')

    def get_pid(self, error, scaler):
        tnow = millis()
        dt = tnow - self._last_t
        output = 0
        if self._last_t == 0 or dt > 1000:
            dt = 0
            self.reset_I()
        self._last_t = tnow
        delta_time = float(dt) / float(1000)
        output += error * self._kp
        if abs(self._kd) > 0 and dt > 0:
            if isnan(self._last_derivative):
                derivative = 0
                self._last_derivative = 0
```

```
        else:
            derivative = (error - self._last_error) / delta_time
        derivative = self._last_derivative + \
                                ((delta_time / (self._RC + delta_time)) * \
                                    (derivative - self._last_derivative))
        self._last_error = error
        self._last_derivative = derivative
        output += self._kd * derivative
    output *= scaler
    if abs(self._ki) > 0 and dt > 0:
        self._integrator += (error * self._ki) * scaler * delta_time
        if self._integrator < -self._imax: self._integrator = -self._imax
        elif self._integrator > self._imax: self._integrator = self._imax
        output += self._integrator
    return output
def reset_I(self):
    self._integrator = 0
    self._last_derivative = float('nan')
```

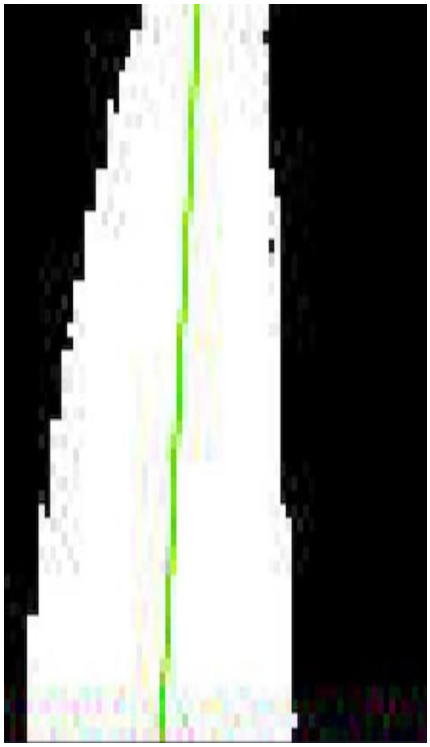
PID 代码编写

前文已经介绍该项目使用 PID 情况，分别为超声波控制距离，小车视觉循迹，这里不做赘述。

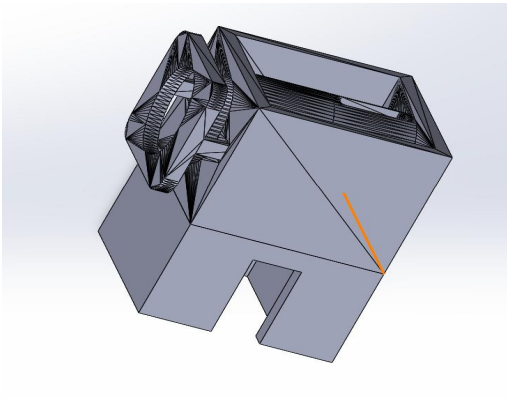
实现效果：

最终小车完成所有任务，包括巡线，物块抓取与放置，机械臂写字（完成一半）。用时较长，但实现效果较好，所有物块都精确放置再放置台，循迹能漂移，转直角弯，物块识别准确，整体失误少。（由于未来得急拍部分任务视频，只放一些任务图片。）

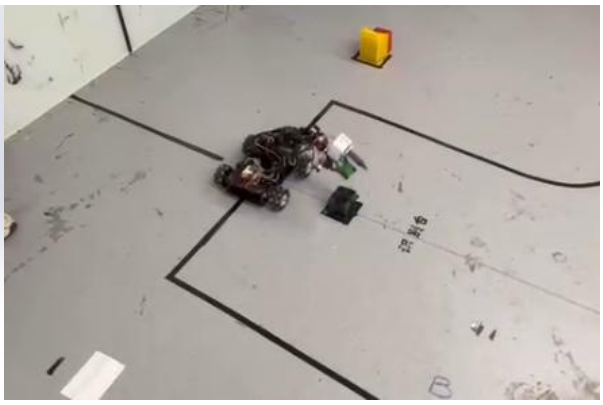
```
def line_following_state_machine(img):
    move=400
    rho_output = 0  # 初始化rho_output
    theta_output = 0  # 初始化theta_output
    img = img.binary([line_THRESHOLD])
    line = img.get_regression([(100, 100)], robust=True)
    if (line):
        rho_err = abs(line.rho()) - img.width() / 2
        # center=(line.x1())-img.width() / 2
        if line.theta() > 90:
            theta_err = line.theta() - 180
        else:
            theta_err = line.theta()
        img.draw_line(line.line(), color=127)
        # print(rho_err, line.magnitude(), rho_err)
        if line.magnitude() > 8:
            if abs(theta_err)>25:
                # move=350
                if theta_err>25:
                    theta_err=25
                if theta_err<-25:
                    theta_err=-25
            # else: move=350
            rho_output =round( -rho_pid.get_pid(rho_err, 1))
            theta_output = round(theta_pid.get_pid(theta_err, 1))
            # output = rho_output + theta_output

        else:
            # output=50
            theta_output=100
    else:
        # output=100
        theta_output=150
        pass
    FullDirection(move,rho_output ,theta_output)
```



（路线拟合效果）



优化笔架



抓取效果

循迹效果

4. 心得体会

在参与本次机器人综合实践环节的过程中，我深刻体会到了跨学科知识在解决实际问题中的重要性和团队协作的力量。通过将课堂上学到的理论知识应用到实际的机器人项目中，我更加深刻地理解了移动机器人的工作原理和控制策略。实际操作中的调试和问题解决过程，让我对理论知识有了更加直观和深刻的认识。在项目中，我们小组成员各司其职，发挥各自的专长。我负责 OpenMV 视觉模块的开发，这让我意识到，一个成功的项目需要团队成员之间的密切合作和有效沟通。每个人的工作都是项目成功的关键一环。在项目实施过程中，我们遇到了诸多挑战，如循迹算法的优化、机械臂的精确控制等。通过不断尝试和调整，我学会了如何分析问题、提出解决方案，并在实践中不断优化和完善。在编程和硬件调试中，我认识到了对技术细节的关注的重要性。一个小小的代码错误或硬件连接不当，都可能导致整个系统的失败。因此，细心和耐心是工程师必备的品质。在设计机械臂执笔装置和优化循迹算法时，我尝试了多种方法，并最终找到了有效的解决方案。这个过程锻炼了我的创新思维，让我学会了在面对困难时，如何跳出传统思维模式，寻找新的解决途径。通过参与项目

的整体规划和时间管理，我学会了如何合理分配时间和资源，确保项目按时完成。这对我的未来职业生涯无疑是宝贵的经验。技术在不断进步，新的问题和挑战也在不断出现。通过这次实践，我更加坚定了持续学习、不断探索新知识的决心。

总之，这次综合实践环节不仅让我在技术和工程实践方面得到了提升，也让我在团队协作、问题解决和创新思维等方面有了显著的成长。我相信这些经验和体会将对我的未来学习和工作产生深远的影响。