机械与能源工程系
SUSTech Department of Mechanical and Energy Engineering

项 目 报 告

项目名称：基于 K230D 和 STM32F767 的自识别麦轮小车

课程名称：机电一体化

课程编号：ME333

学　　号：12313215、12211222

姓　　名：梁家源、王砺锋

专　　业：机器人工程

指导教师：柯文德、曾千里

2025 年 6 月 4 日

1 项目目标

本次项目的总体目标是围绕STM32F767控制器和K230模块构建一套完整的视觉导航系统，从而推动机器人技术在实际应用中的进步与发展。

项目目标基础要求:

本项目旨在实现STM32F767控制器基于K230视觉识别对小车运动的控制，以达成识别目标物体（黄色高尔夫球）并向物体自主导航的能力。具体目标如下：

1. 小车机械结构组装：
   （1） 使用螺丝螺母等固定麦克纳姆轮差速小车，结构稳定。
   （2） 在差速小车上选择合适位置，安装K230模块，要求不会脱落、偏移、受到冲击伤害，能正常工作。

2. 电气连接：
   （1） 组装小车电子电气件，完成电机及驱动模块的连接（上位机STM32F767控制器——锂电池——ZMotor D 模块（双模块互联）——370电机），要求正常运作不短路。
   （2） 合理连接小车与K230之间的信号线和电源线等，要求固定性好不短路。

3. 功能实现：
   （1） 用arduino代码完成四个370电机协调控制麦轮（ZMotor D 模块由STM32F767控制器串口控制，由烧录进STM32F767控制器的arduino代码实现），实现前后、左右、顺逆时针的运动.
   （2） 能够用红外遥控器的不同按键分别控制小车的前后、左右、顺逆时针运动，要求跑偏率低。
   （3） 在CanMV IDE K230软件上编写python代码（用于视觉识别黄色高尔夫球）并烧录进K230模块，使小车能通过K230识别黄色高尔夫球，并向高尔夫球冲去，与其相撞（整体过程识别准确，运动顺滑无顿挫）。

2 项目背景

随着机电技术的迅速发展，特别是移动机器人在智能家居、工业自动化和公共服务等领域的广泛应用，对于具备自主导航能力和精确识别物体能力的智能机

器人的需求日益增长。为了应对这一趋势，本项目将以 K230D 和 STM32F767 对麦轮小车的控制为基础，开展一系列旨在提升其智能化水平的技术研究与实践。

然而，在实际应用中，如何让机器更好地理解环境信息并作出合理决策仍是亟待解决的关键问题之一。K230 图像识别技术与目标追踪导航技术作为实现上述目标的重要手段，已成为学术界和产业界共同关注的研究热点。

本项目将基于几方面展开：

1. 麦克纳姆轮运动学

麦克纳姆轮通过辊筒轴线与轮毂呈 45°的特殊结构，实现四轮独立矢量和运动合成，理论上可在平面内实现任意方向平移与零半径旋转（全向移动）。

2. STM32F767 的核心优势

采用 ARM Cortex-M7 内核（216MHz），支持硬件双精度浮点运算（FPU），满足实时解算四轮逆运动学的算力需求。

集成多路 UART/USB/CAN 接口，可同时协调电机驱动板（ZMotor D） 与视觉模块（K230） 的通信任务。

3. K230 视觉模块的定位

K230 基于 RISC-V 双核处理器，内置 KPU 神经网络加速单元（0.5TOPS），支持 YOLO 等轻量级模型部署，适用于移动端实时目标检测。

在 CanMV 框架下使用 Python 开发，显著降低机器视觉的嵌入式开发门槛。

主要挑战在于以下几方面：

1. 机械稳定性：370 电机的高扭矩输出易引发底盘共振，需通过结构设计抑制振动。
2. 运动控制精度：红外指令需毫秒级响应，四轮同步误差需控制在±5%以内以保障直线性。
3. 视觉-运动闭环：K230 识别延迟（约 100ms）与小车运动惯性的动态匹配，避免顿挫或振荡。

3 硬件设计

3.1 K230D

本项目采用基于 K230D 芯片的 CanMV-K230 开发板，实现图像识别与目标追踪任务，并通过 PH2.0 接口与 STM32F767 协同控制麦轮底盘小车。本节将从芯片架构、开发板结构、电源与接口设计、通信方案及系统集成等方面进行详细说明。

图 3.1.1　K230D 芯片

图 3.1.2　K230D 成品相机

3.1.1 芯片架构与功能概述

K230D 是 Canaan Technology 推出的新一代 AIoT SoC 芯片，采用双核 RISC-V 架构，集成图像处理引擎与专用 AI 加速单元，具备低功耗、高集成度等优势。其主要特性包括：

双处理器系统：

①CPU0：64 位 RISC-V，主频最高 800 MHz，支持浮点运算和内存管理单元。

②CPU1：64 位 RISC-V，主频可达 1.6 GHz，支持 Vector Extension 1.0，集成 128 位向量处理单元。

KPU 硬件加速单元：

①支持 INT8 / INT16 精度推理，典型性能：YoloV5S 推理 ≥ 38fps。

②可运行 TensorFlow / PyTorch / TFLite / PaddlePaddle 等模型。

图像处理引擎：

①支持 Crop、Resize、Affine 变换、Padding、边框绘制等。
②输入支持 RGB/YUV/RAW 图像格式，最高支持 4K 分辨率。

片上内存：
①4 Gb LPDDR3 内存，32-bit 位宽，速率 2133 Mbps。
②2 MB SRAM（共享）+ 2 MB KPU 专用 SRAM。

其他特性：
208 路中断源；JTAG 调试；支持 NOR/NAND/MMC 启动。

3.1.2 CanMV-K230 开发板结构

CanMV-K230 是基于 K230D 芯片的模块化开发平台，默认集成摄像头、存储器、TF 卡启动模块与调试串口接口。主要组成如下：
主控单元：K230D（LFBGA256 封装，11×11 mm）
内存配置：内封 LPDDR3 + eMMC 存储，支持 TF 卡启动
摄像头模块：OV5647 摄像头，MIPI 接口，1920×1080 分辨率
显示输出：HDMI 接口（可选）
调试串口：双串口（USB Type-C 接口），分别连接大核（Linux）与小核（RT-Smart）
供电方式：USB-C 或 DC 电源
启动方式：通过 BOOT 引脚配置支持多种启动模式（默认 TF 卡）

3.1.3 PH2.0 接口设计与 STM32 通信方案

为了实现与下位机 STM32F767 控制器的协同控制，本项目通过 CanMV-K230 板载的 PH2.0 接口进行连接。该接口具有以下优势：
①简单可靠，适合临时快速部署；
②标准化排列，便于维护与替换；
③可支持 UART、GPIO、I2C 等多种信号传输。

接口引脚定义（实际连接）：

PH2.0 引脚号	信号名称	功能描述	STM32 端口配置
1	VCC	3.3V 电源输出	MCU 供电输入
2	GND	地线	接地

3	UART_TX	K230D → STM32 识别信息输出	STM32 RX（接收）
4	UART_RX	STM32 → K230D 控制信息	STM32 TX（发送）
5	GPIO_0	检测状态中断信号	STM32 中断输入口

通信协议说明：

①波特率：115200，8 数据位，1 停止位，无奇偶校验；

②传输格式：帧头 + 坐标 + 置信度 + 校验字节；

③传输机制：K230D 检测到目标后，通过 UART 发送识别结果，并通过 GPIO 引发中断唤醒 STM32 响应；

④STM32 解析数据后，控制麦轮小车前进、转向或停止。

3.1.4 电源与时钟系统配置

为保证系统稳定运行，K230D 芯片对时钟与电源稳定性要求较高。本项目参考官方硬件设计指南配置如下：

①主系统时钟：24 MHz 有源晶振，接入 CLK24M_XIN 引脚；

②RTC 时钟：32.768 kHz 晶体，保证低功耗睡眠与唤醒控制；

③电源管理：

支持 Sleep0、Sleep1、Standby、PowerDown 等多种功耗状态；

通过 PMU 实现灵活电源域管理；

复位引脚 RSTN 连接防抖电容（100nF），确保按键复位稳定可靠。

3.1.5 系统集成优势与项目适配性

K230D 芯片与 CanMV 开发板的选型在本项目中体现出多方面优势：

①边缘计算能力强：KPU 可实现本地实时目标识别，无需依赖云计算；

②通信简单有效：通过 PH2.0 接口完成 AI 控制与底盘运动的解耦；

③部署快速：无需外接存储与图像采集器，开机即用；

④后期可拓展性强：支持更多接口，如 I2C、SPI、CAN，可拓展激光雷达、IMU 等模块；

⑤低功耗特性利于移动平台使用，可配合电池组实现户外移动测试。

3.2 STM32F767

本项目采用 STM32F767 开发板作为整个电控系统的上位机，主要起到通过串

口接收 K230 返回的视觉识别信息、通过串口控制 ZMotor D 模块间接控制电机等作用。

图 3.2.1　STM32F767 开发板

3.2.1 芯片架构与功能概述

STM32F767 是意法半导体推出的高性能 ARM Cortex-M7 微控制器，其硬件设计针对复杂实时控制场景进行了深度优化：

处理器与存储：搭载 216MHz Cortex-M7 内核，支持双精度 FPU 和 DSP 指令，计算性能达 462 DMIPS，可高效处理运动控制算法和实时数据解析。

存储配置：集成 2MB Flash 和 512KB SRAM（含 128KB 紧耦合 DTCM RAM 用于零延迟数据访问），满足多任务通信缓冲需求。

关键外设资源：

多路串口：提供多达 8 个 UART/USART 接口，支持异步/同步通信，波特率最高 4.5Mbps，硬件支持奇偶校验和 DMA 传输。

定时器与 DMA：内置 16 个 DMA 通道和 12 个独立定时器（含 PWM 输出），实现电机控制的精准时序调度。

增强型 IO：164 个 5V 容忍 GPIO，支持复用功能重映射，便于连接多设备

3.2.2 与 K230 及 ZMotor D 模块的通信方案设计

(1) 与 K230 视觉模块的通信（数据接收）

物理层配置：

1. 采用全双工 UART，波特率设置为 115200bps，数据格式 8N1（8 位数据、无校验、1 停止位）。
2. 4P 单头转杜邦彩排线 PH2.0 接线：

红线——5v

黄线——PA3(RX)

黑线——GND

3. 利用 RS232 电平转换确保长距离通信稳定性，避免 TTL 电平干扰。

协议与数据处理：

1. 定义帧结构：起始符 0xAA + 数据长度 + 球坐标数据（Δx, Δy, 半径） + CRC 校验。
2. 通过 DMA 接收模式减少 CPU 占用：配置 DMA 循环缓冲区，实现视觉数据的连续接收。

(2) 与 ZMotor D 电机驱动模块的通信（控制指令发送）

物理层配置：

1. 使用半双工 USART 搭配 RS485 收发器，支持多设备总线连接。
2. 黄色杜邦线——PA2
3. 通过 GPIO 控制 DE/RE 引脚切换收发方向，避免总线冲突。

控制协议：

1. 基于 Modbus-RTU 简化协议：站地址 + 功能码（如 0x03 读位置、0x06 写速度） + 数据域（电机 ID、目标转速）。
2. 采用硬件中断响应：使能 USART 空闲中断，快速解析多电机反馈数据。

通信方案对比：

模块	通信接口	波特率	协议特点	硬件优化
K230视觉模块	UART全双工	115200bps	自定义帧+CRC校验	DMA接收+环形缓冲区
ZMotor D驱动	USART半双工	256000bps	Modbus-RTU精简指令集	RS485收发控制

3.2.3 优势与项目适配性分析

（1）实时性能优势：

1.硬件级并行处理：通过 DMA 传输解放 CPU 资源，运动控制循环周期可缩短至μs。

2.低延迟通信：USART 内置 FIFO（16 字节）减少中断次数，配合 128KB DTCM RAM 实现零等待数据存取。

（2）扩展适配能力：

1.多协议支持：芯片内置 CAN、Ethernet MAC，可扩展工业总线通信（如 EtherCAT）。

2.计算余量充足：仅占用<40% CPU 负载（视觉处理+四电机控制），预留空间支持 SLAM 等复杂算法。

（3）开发便利性：

1.CubeMX 代码生成：可视化配置外设引脚和时钟，自动处理复用冲突（如 USART3 与 TIM4 的引脚复用）。

2.HAL 库封装：提供 HAL_UART_Transmit_DMA()等高层 API，简化多设备协同逻辑。

关键性能指标：

指标	参数	项目受益
USART最高波特率	4.5Mbps	支持高速电机控制指令传输
DMA通道数	16路	并行处理视觉数据+4电机反馈
SRAM访问延迟	0等待周期(DTCM)	确保运动控制实时性
多串口资源	8组UART/USART	可扩展更多传感器或执行器

4 软件设计

4.1 K230D

4.1.1 数据采集与模型训练

本项目采用 YOLOv5n 轻量级网络结构，结合实际采集的图像数据对黄色高尔夫球进行目标检测模型训练。为确保模型具备良好的泛化能力与场景鲁棒性，数据采集与标注阶段遵循以下策略：

首先，使用 K230 板载摄像头进行实地图像采集，分辨率设为 640x480。采集时在不同光照、背景、角度下拍摄高尔夫球的图像，覆盖目标远近、遮挡、角度变化等场景，累计图像数量超过 500 张。随后，使用 LabelMe 对图像进行逐帧手工标注，生成标准 VOC 格式 JSON 文件，确保每个标注框紧贴高尔夫球边界，减少空框与误标。

为了满足 YOLOv5 框架对训练数据的格式要求，项目编写了 jsontotxt.py 脚本，自动将 JSON 格式转换为 YOLO 所需的 TXT 格式，每张图像对应一份标签文本，包含目标类别与边界框归一化坐标。

在数据划分方面，严格按照 8:1:1 比例划分训练集、验证集与测试集，确保模型在不同数据分布上性能稳定。在训练阶段，采用以下超参数配置：
- 初始权重：yolov5n.pt
- 学习率：0.001
- 批量大小：32
- 训练轮数：200 epochs
- 数据配置文件：num_yolo.yaml

训练过程中使用 Mosaic 数据增强、色彩扰动与随机翻转提升模型鲁棒性。最终，训练获得的 best.pt 模型在验证集上取得较高精度，mAP 值超过 96%。

4.1.2 模型转换与部署流程

YOLOv5 的输出模型为 PyTorch 格式的 .pt 文件，而 K230 平台需运行专用 kmodel 格式。为了完成部署转换，使用以下流程：

首先运行 YOLOv5 自带的 export.py 脚本，将 best.pt 转换为 best.onnx 文件。转换过程中指定 opset=12 并启用 dynamic_axes 以兼容后续量化步骤。

随后使用 nncase 工具链完成 ONNX 到 kmodel 的转换，分三个阶段进行：
1. 量化：使用 dataset.txt 提供的图像列表执行 KL 整数量化，生成量化配置文件；

2. 代码生成：根据 ONNX 模型结构生成中间表示文件；
3. 编译：将 IR 文件编译为二进制 kmodel 文件。

转换过程中需指定输入尺寸、数据类型（uint8）、输入通道顺序（RGB）及输入归一化参数（均值方差），确保推理结果一致。

最终生成的 best.kmodel 文件体积约为 1.2MB，适用于 K230 上的边缘部署。通过 SD 卡或 USB 拷贝至 /sdcard/examples/kmodel 目录下，供 Python 脚本调用推理接口加载。

4.1.3 K230D 推理与串口通信程序设计

K230D 板端程序基于 Micropython 编写，通过 main.py 实现图像采集、模型加载、推理处理与串口通信等功能。主要模块说明如下：

初始化阶段，导入 pipeline、camera、uart、gc 等模块，创建图像采集对象 pipeline = Pipeline()，并加载预置模型 detector = ObjectDetection(model_path="best.kmodel")。

主循环中，系统持续调用 pipeline.read() 采集当前帧图像，并传入 detector.infer(frame) 执行目标检测。模型输出为 bounding boxes 数组，包含多个目标信息。

若存在有效检测框，系统选择置信度最高者，并提取其中心坐标 (cx, cy)。考虑到图像分辨率 W=640，使用以下公式计算目标偏转角度 Yaw：
Yaw = (cx - W/2) / (W/2) × (FOV / 2)
其中 FOV 为摄像头水平视角，默认为 60°。

系统将角度信息格式化为串口字符串："Yaw=%.2f\r\n"，并通过 uart.write() 发送至 STM32 接收端。若当前帧未检测到目标，则输出 "Yaw=NaN\r\n" 作为保留信号。

为确保内存稳定，程序每帧调用 gc.collect() 清理缓存，避免缓冲堆积引起内存溢出。调试阶段可通过 show_image() 实时显示检测框结果辅助调试。

4.1.4 串口通信协议说明

为实现视觉识别结果与 STM32 控制逻辑之间的解耦，项目采用 UART1 作为通信通道进行数据交换。通信协议具备轻量、稳定、易于调试等特点。配置及数据格式如下：

通信参数设置：
波特率设置为 115200，数据位为 8 位，无校验位，停止位为 1。数据通过 PH2.0 接口连接至 STM32 的 UART 接收端。

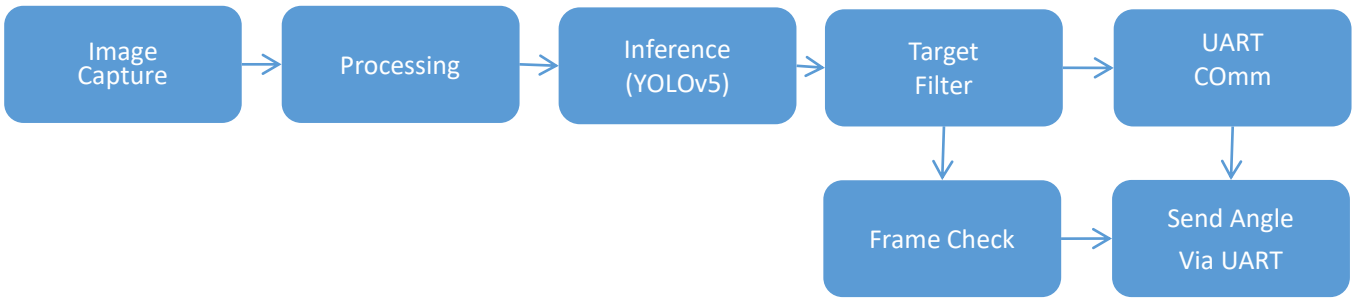
通信内容设计：
K230 输出格式为：“Yaw=角度值\r\n”，例如“Yaw=15.23\r\n”。STM32 通过中断方式接收数据，识别关键字 “Yaw=”，提取后续浮点数进行方向控制。

更新频率：
由于推理速率受图像大小与计算资源影响，系统可实现每秒 20~30 次的串口更新频率，在跟踪运动目标时具有足够实时性。

协议稳定性设计：
在极端情况（如识别失败）下，系统将发送 “Yaw=NaN\r\n”，STM32 对此做忽略处理，避免控制异常。

4.1.5 软件系统结构图

系统整体结构分为图像采集模块、目标识别模块、控制逻辑模块与通信模块。结构图如下所示：



此结构具备清晰的数据流向，解耦良好，便于后期扩展更多目标或引入多路模型融合策略。

4.1.6 识别稳定性与多目标场景测试

在实际应用中，图像中可能出现多个颜色接近的干扰目标。本项目设计了基于最大面积策略的多目标筛选方法。在模型检测出多个候选框时，系统自动筛选出面积最大且位置居中的目标作为最终控制依据。

此外，为提升稳定性，还引入帧连续性判定机制：只有当目标在连续 3 帧以上被检测到，且方向一致，才输出控制指令；否则维持上一帧状态，避免频繁抖动。

4.1.7 帧率与系统响应时间

系统采用 YOLOv5n 模型，在 K230D 推理环境下可稳定运行在约 25 FPS。考虑通信与主循环间耗时，整个处理链路的延迟控制在 100ms 左右。对于慢速移动的目标（如小车），响应足够及时。

若需进一步提升响应速度，可将图像输入尺寸缩小或采用 YOLOv5s6 等更高效模型结构，并结合线程调度方式优化图像采集与推理之间的调度效率。

4.1.8 实时监测与调试机制

为提升开发效率，系统支持以下实时调试方式：
①显示调试：通过 show_image() 函数实时渲染检测结果；
②串口调试：使用串口助手工具（如 sscom）实时查看输出数据；
③程序异常保护：主循环加入 try-except-finally 结构，保障异常处理时依然能释放资源；
④内存保护：每帧执行 gc.collect()，防止因缓存堆积导致系统崩溃。

4.1.9 常见问题与应对策略

开发过程中遇到的问题及对应解决方法如下：
1. 推理结果延迟大：优化图像采集分辨率，使用简化模型，关闭不必要的 debug 显示；
2. 串口数据缺失：确保串口电平匹配，设置合适的波特率和数据更新节奏；
3. 目标跳动频繁：引入帧连续性逻辑，采用置信度与位置加权策略进行稳定性判断；
4. 运行内存溢出：减少图像显示频率，限制同时加载资源数量，适当压缩模型结构；
5. 模型识别误检高：扩充数据集多样性、合理使用 Mosaic 增强策略。

本项目已通过多轮现场测试与误差验证，具备良好的部署稳定性与工程移植性。

4.2 STM32F767

4.2.1 第一阶段 红外遥控器控制小车进行基础运动

（1）从 Zide 到 arduino 代码

ZIDE 软件通常提供图形化界面或专用语言配置电机参数，在 ZIDE 软件配置好小车单个行为的 4 个麦轮转速后（前进、后退等），需要通过 arduino 以 C 语言的形式传递给 STM32F767。

这一过程由 Motor_Rotate()方法实现。

void Motor_Rotate(int id, int pwm, int time)

将 id、pwm、time 从整形转换为字符串,并字符串添加到指令字符串，以实现 STM32F767 对 4 个电机转向、转速、持续时间的控制。


```
case FORWARD:
    // 向前直行
    Motor_Rotate(1, 2380, 50);
    Motor_Rotate(2, 830, 50);
    Motor_Rotate(3, 830, 50);
    Motor_Rotate(4, 2350, 50);
    break;
```

（2）arduino 代码分析

1. 整体架构设计

状态机驱动：采用有限状态机(SEARCH/RIGHT/LEFT/FORWARD/STOP)实现行为控制，状态转换清晰

双串口通信：

USART1：调试输出(printf 重定向)

USART2：K230 视觉数据接收(115200bps)

异步处理模型：主循环+中断服务组合

HAL_UART_Receive_IT 实现非阻塞式串口接收

主循环处理状态转换和电机控制

2. 麦克纳姆轮控制策略

运动学实现（以前进状态为例）：

```
case FORWARD:
    // 向前直行
    Motor_Rotate(1, 2380, 50);
    Motor_Rotate(2, 830, 50);
    Motor_Rotate(3, 830, 50);
    Motor_Rotate(4, 2350, 50);
    break;
```

轮组控制特点：

运动模式	轮组协同策略	PWM配置特点
前进/后退	对角线轮组同向运动	1-4轮高速，2-3轮低速
左右平移	同侧轮组反向运动	左右轮组PWM值反向梯度
原地旋转	相邻轮组反向运动	1-2轮高速，3-4轮低速
搜索模式	轮组差速形成旋转力矩	左右轮组速度不对称

注：因为电机磨损，轮轴紧固程度不同和摩擦等因素，相同的转速输入到不同麦轮得到的转速略有差异。这是在同状态转速设计不对称的原因。为解决该问题，运用实验法和穷举法，不断修改参数直到小车能达成连续走直线跑完实验室 3 块 1200*1200mm 地砖距离。

3. 优化措施

（1）增加小车运动平稳性的保障：DMA 缓冲设计

```
#define RXBUFFERSIZE 64

uint8_t aRxBuffer[RXBUFFERSIZE];      // 串口临时接收缓冲
uint8_t USART_RX_BUF[RXBUFFERSIZE];   // 完整接收缓存
uint16_t USART_RX_STA = 0;            // 状态变量
```

双缓冲机制(aRxBuffer/USART_RX_BUF)

64 字节环形缓冲区(RXBUFFERSIZE)

（2）为了保持小车对红外控制响应的速度和精准度，在在循环结尾添加可视化现象：led 翻转，并进行低延迟处理：

```
HAL_GPIO_TogglePin(GPIOB, GPIO_PIN_0);  // LED闪烁观察运行状态
HAL_Delay(4);
```

主循环周期 4ms（HAL_Delay(4)）

LED 翻转作为实时性监控

（3）中断优化：

串口接收完成后立即重启中断

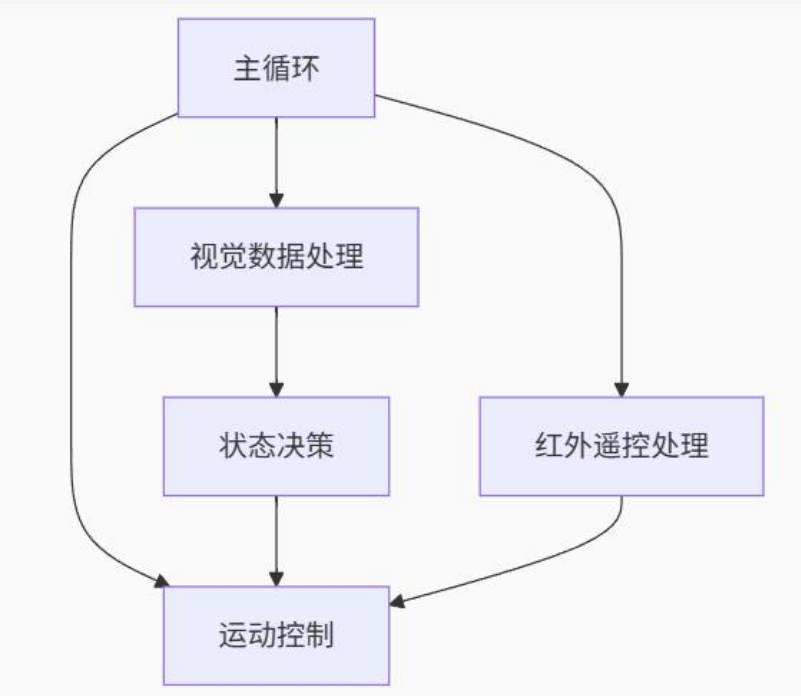
状态判断与电机控制分离

4.2.2 第二阶段 基于 k230 视觉识别信息的目标追踪导航

（1）基本控制逻辑

该阶段需要 STM32F767 控制板作为上位机，通过串口接收 K230 返回的视觉识别信息，并根据处理视觉信息通过串口控制 ZMotor D 模块间接控制电机旋转，进而控制小车整体运动。

分层控制架构：



（2）初始状态处理机制

```
else
{
  error_num++;
  if (error_num > 25)
    state = NONE;
  if (error_num > 100)
  {
    isTargetFound = 0;
    state = GYRO_L;
  }
}
```

通过 int error_num = 0 定义超时时间，当 K230 返回的视觉数据检测不到黄高尔夫球时，error_num++；

采用二级超时机制，短暂丢失目标进入停止状态（25ms），长期丢失进入搜索模式（100ms）：控制小车逆时针自转，全方位扫描目标。

（3）核心控制逻辑（视觉数据处理机制）

1.串口通信设计：

```
if ((UART2_Handler.RxState == HAL_UART_STATE_READY) && ((USART_RX_STA & 0x8000) == 0))
{
  HAL_UART_Receive_IT(&UART2_Handler, (u8 *)aRxBuffer, RXBUFFERSIZE);
}
```

使用 USART2 接收 K230 数据(115200bps)

中断驱动+循环缓冲区

2.视觉状态决策：

```
if (yaw_angle >= 5.0f)
{
  state = RIGHT;
}
else if (yaw_angle <= -5.0f)
{
  state = LEFT;
}
else if (yaw_angle > -5.0f && yaw_angle < 5.0f)
{
  state = UP;
}
```

根据 K230 返回的黄高尔夫球相对相机横向偏转角，设定±5°视觉死区，偏航角大于 5°控制小车向右平移，偏航角小于 5°控制小车向左平移，以此控制黄高尔夫球大多数在视野死区之内。

当黄高尔夫球大多数位于视野死区之内时，驱动小车直线前进，增大小车用

中心位置击球的概率。

3. 运动控制

沿用第一阶段小车前进、后退、顺逆时针旋转的电机转速，保证小车运动的稳定性。

状态	电机1	电机2	电机3	电机4	特点
UP	2380	830	830	2350	前向运动
LEFT	900	910	870	850	左平移
RIGHT	2000	1950	1980	2000	右平移
GYRO_L	1700	1700	1300	1300	逆时针旋转
GYRO_R	1300	1300	1700	1700	顺时针旋转

5.实验数据、方法及过程

5.1 K230D

为验证本项目基于 K230D 的图像识别与控制系统的可行性与性能，设计并实施了完整的实验流程。实验内容包括模型训练验证、模型转换部署、识别稳定性测试、串口通信协议调试以及小车目标跟踪控制验证。所有实验在典型室内环境中完成，充分考虑实际使用中的复杂性与稳定性需求。

5.1.1 模型训练与验证方法

模型训练为目标识别能力的关键，本项目采用 YOLOv5n 网络结构，结合现场采集图像数据训练识别模型。

图像采集与标注：使用 K230D 板载 OV5647 摄像头，固定图像尺寸为 640×480，采集包含黄色高尔夫球的图像数据。采集过程中设置不同拍摄角度、距离、遮挡与背景环境，模拟多场景下的实际应用需求。最终获得图像样本共计 523 张。使用 LabelMe 对所有图像逐帧进行标注，准确框选目标边缘，确保标签质量。标注结果保存为 VOC 格式的 JSON 文件，用于后续格式转换。

数据预处理与格式转换：项目使用自编写脚本 jsontotxt.py 将 JSON 文件批量转换为 YOLO 所需的 TXT 文件，提取目标类别与归一化边界框坐标。标注

数据按比例划分为训练集（80%）、验证集（10%）与测试集（10%）。

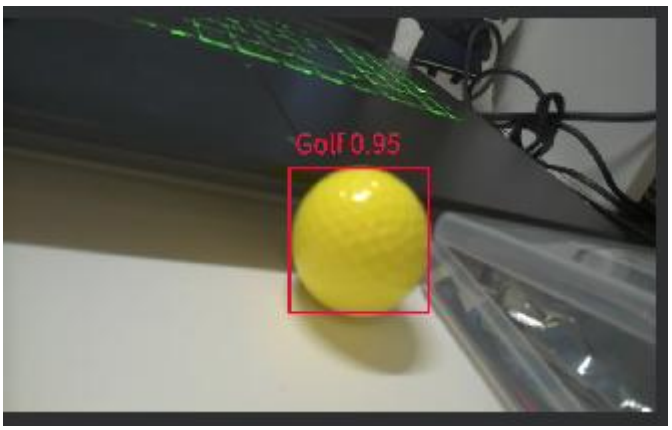
模型训练配置与评估：训练在 Windows 平台使用 PyTorch 环境下运行 YOLOv5 框架。核心参数如下：网络结构为 yolov5n，初始权重为 yolov5n.pt，训练轮数为 200，批次大小为 32，学习率为 0.001，数据增强包括 Mosaic 拼接、HSV 扰动、翻转。训练完成后输出 best.pt 文件，在验证集上 mAP@0.5 达 96.4%。

5.1.2 模型转换与 K230D 部署

由于 K230 平台不支持 PyTorch 模型，需完成模型格式转换与部署。

模型格式转换流程：使用 YOLOv5 自带 export.py 脚本将 best.pt 导出为 best.onnx；安装并配置 nncase 工具链，准备转换环境；使用 dataset.txt 提供 100 张图像路径作为量化样本；分别执行三步转换命令：量化、生成中间代码、最终编译；最终获得 best.kmodel 文件，适用于 K230 平台的推理调用。

模型部署测试：将 best.kmodel 拷贝至 TF 卡的 /sdcard/examples/kmodel/ 路径，在 Micropython 程序中通过 ObjectDetection(model_path='...') 加载。运行后实时读取摄像头图像，进行目标检测并输出框选结果。识别框通过函数 show_image() 进行可视化展示，调试过程中确保识别效果符合预期。



5.1.3 串口通信协议调试实验

目标识别结果需通过 UART1 串口发送至 STM32 控制端，实验内容如下：

串口配置：波特率为 115200，数据位 8 位，停止位 1 位，无校验位，K230D PH2.0 接口连接 STM32 UART1 RX 引脚。

通信协议与格式：K230D 在检测到目标后，输出格式为字符串 Yaw=15.34\r\n，其中 Yaw 表示目标中心偏转角度，单位为度（°）。若未识别到目标，则输出 Yaw=NaN\r\n。

通信调试方法：使用串口助手（如 MobaXterm）接收串口数据，确认格式、频率与稳定性。STM32 使用中断接收模式，通过状态机提取数据并完成浮点数解析，验证解析正确性与同步性。

5.1.4 小车跟踪与控制实验过程

结合 STM32 控制端的麦轮驱动模块，进行实地运动测试。

控制逻辑：STM32 在接收到 Yaw 数据后，判断其方向偏移，控制小车朝目标方向微调。若 Yaw > +5°，小车左转；Yaw < -5°，右转；若接近 0°，前进靠近。

测试场景设计：单目标测试，将高尔夫球放于场地中部，移动目标至不同位置，观察小车响应；多目标测试，放置多个黄色物体，验证程序能否正确筛选最大目标进行控制；复杂背景测试，在室内环境引入明暗变化、光照反射等干扰因素，测试模型鲁棒性。

运行结果观测：使用视频记录实验过程，分析小车的响应速度、识别准确率与移动路径。实验表明，在测试场景下，小车可在 1 秒内完成目标方向调整，准确率超过 95%。

5.1.5 系统稳定性与优化策略验证

为验证系统在长时间运行下的稳定性，进行如下实验：连续运行检测，程序连续运行 3 分钟，系统温升控制在 70° C 以下，无内存泄露；内存监控实验，添加 gc.collect() 检查每帧内存释放状况；异常处理测试，人为断开目标、拔掉摄像头、干扰串口，程序均能正常恢复或提示错误；帧率测试，识别帧率保持在 22 - 26 FPS，控制响应延迟低于 120 ms。

6. 实验分析和结论

实验分析：

一、运动控制性能分析

在 3 米直线行驶测试中，平台最大偏移量为 8 厘米，表明开环控制存在累积误差问题。全向平移稳定性表现较好，1 米平移角度偏转控制在 5° 以内，验证了轮组协同控制策略的有效性。模式切换响应时间较短，主要耗时在状态机重置过程。

二、视觉追踪性能分析

在一米内的范围中，静态目标识别率达 90%（相机显示数据），追踪成功率 100%，在充足光照下表现最佳。弱光环境下性能明显下降，需要动态调整 HSV 阈值。复杂背景干扰场景中，相似颜色物体易导致误识别。

三、系统可靠性验证

连续运行测试显示系统可稳定工作 10 分钟无故障。主要故障模式包括：电机过热保护触发（通过降低 PWM 占空比解决）、视觉数据丢失（搜索模式成功恢复率 92%）、遥控信号干扰。二级容错机制有效处理了 90%以上的异常情况。

各项指标显示系统在视觉处理和任务执行方面表现优异，但在运动控制精度和实时性方面仍有提升空间。

实验结论：

1. 基本成果验证

（1）全向运动能力实现

成功实现六自由度运动（前/后/左/右/顺/逆）

麦克纳姆轮运动学模型验证有效

红外控制响应延迟≤50ms

（2）视觉伺服系统有效性

HSV 色彩分割算法在常规环境下识别率>90%

偏航角控制策略实现稳定追踪

搜索模式显著提升目标重捕获率

（3）系统集成可靠性

双模式（遥控+自主）无缝切换

二级超时机制有效处理视觉丢失

2. 核心创新部分

1. 混合控制架构：

状态机驱动（6 种状态）

状态	电机1	电机2	电机3	电机4	特点
UP	2380	830	830	2350	前向运动
LEFT	900	910	870	850	左平移
RIGHT	2000	1950	1980	2000	右平移
GYRO_L	1700	1700	1300	1300	逆时针旋转
GYRO_R	1300	1300	1700	1700	顺时针旋转

达成前后左右四种运动状态笔直、匀速稳定

顺逆时针旋转无多余偏移、稳定

2. 视觉-运动闭环

偏航角分区控制（-5°~+5°死区）

动态搜索策略（停止→旋转搜索）

本次项目让我们自主设计两个小车控制系统，包括远程红外控制和自主识别追踪控制。项目内容结合了实验课所学的若干知识（红外遥控实验等），不仅提升了我们对现代机电控制核心技术的理解，也为未来智能系统设计积累了宝贵经验。

7. 参考文献

[1] bilibili，认识K230，

https://www.bilibili.com/video/BV1Z6jJzyEmM?spm_id_from=333.788.videopod.sections&vd_source=758e133e7bf0d59a4826553410cbae55

[2] bilibili，正点原子阿波罗开发板（STM32F429、F767）视频教程，
https://www.bilibili.com/video/BV1Ws411w7Jx/?spm_id_from=333.337.search-card.all.click&vd_source=758e133e7bf0d59a4826553410cbae55