2022年TI杯大学生电子设计竞赛

C题：小车跟随行驶系统

设计报告

参赛队号：	[在此处填写队号]
参赛学校：	[在此处填写学校]
参赛队员：	[队员一]
	[队员二]
	[队员三]
指导教师：	[在此处填写教师]

2025 年 7 月 23 日

摘要

本作品设计并制作了一套基于TI系列MCU的小车跟随行驶系统，由一辆领头小车和一辆跟随小车组成。系统严格按照2022年TI杯大学生电子设计竞赛C题要求，实现了内外圈循迹、多速档切换、双车协同跟随、动态追赶、自主超车以及响应“等停指示”等复杂功能。

系统以TI MCU为控制核心，领头小车与跟随小车均采用差速驱动底盘。每辆小车集成8路灰度传感器用于精确循迹，编码器用于速度和里程测量，并通过无线通信模块（如蓝牙或2.4GHz射频）实现双车间的数据交互与协同控制。距离控制采用超声波传感器或红外测距模块，确保车间距的精准保持。

算法层面，我们采用了鲁棒性强的PID控制算法分别对速度和循迹进行闭环控制。针对双车协同，设计了基于主从模式的通信协议和控制策略，实现了领头小车对跟随小车的完全控制。对于超车等复杂任务，我们设计了精巧的状态机逻辑，保证了任务切换的平稳与可靠。

经赛题要求的各项功能测试，本系统表现稳定，速度控制误差小于10%，停车间距误差小于6cm，能够可靠地完成所有基本要求和发挥部分，展示了良好的动态性能和系统鲁棒性。

关键词： TI MCU；小车跟随；PID控制；无线通信；循迹；协同控制

目 录

1 系统方案设计

1.1 系统方案描述

本系统由一辆领头小车和一辆跟随小车构成，旨在完成赛题指定的循迹、跟随、超车及避障等任务。系统以TI系列MCU为核心，整体方案围绕高精度循迹、稳定速度闭环、可靠车间通信和精确距离控制四大目标进行设计。

系统主要模块功能：

- **主控模块：** 两车均采用TI MSPM0G3507微控制器，基于ARM Cortex-M0+内核，工作频率32MHz，具有128KB Flash和32KB SRAM，负责数据处理、算法运行、运动控制和通信协调。
- **传感器模块：**
  - **循迹：** 8路灰度传感器阵列，用于检测1cm宽的黑色引导线。
  - **测速：** 正交编码器，用于精确计算车轮转速和行驶里程。
  - **测距：** 超声波或红外测距传感器，用于维持两车20cm的安全间距。
  - **“等停指示”检测：** 额外的红外对管或灰度传感器，用于识别赛道上的特定标记。
- **驱动模块：** L298N或更高效的电机驱动芯片，实现对直流减速电机的PWM调速与方向控制。
- **执行机构：** 差速驱动底盘，结构简单，控制灵活，能够实现小半径转向。
- **通信模块：** 采用蓝牙或nRF24L01等2.4GHz无线模块，实现领头小车对跟随小车的指令下发和状态上报。
- **人机交互模块：**
  - **领头小车：** 配备启动/设置按键和OLED显示屏，用于任务选择和状态显示。
  - **跟随小车：** 仅设电源开关，完全由领头小车控制。
  - **声光指示：** 蜂鸣器和LED，用于任务完成提示和内圈行驶指示。
- **电源模块：** 采用两节18650锂电池串联供电，并通过LDO或DC-DC稳压芯片为不同模块提供稳定的3.3V和5V电压。

1.2 方案论证与选择

1.2.1 主控制器件的论证与选择

方案一： TI MSP430系列微控制器。优点是功耗极低，外设接口够用，完全符合TI竞赛要求。缺点是主频和计算能力相对较弱，处理复杂协同算法时可能性能不足。

方案二： TI Tiva C系列微控制器 (如TM4C123G)。优点是基于ARM Cortex-M4内核，性能强大，外设资源丰富，但相对复杂，开发周期较长。

方案三： TI MSPM0系列微控制器 (如MSPM0G3507)。优点是基于ARM Cortex-M0+内核，性能与功耗平衡良好，外设资源（如多路ADC、定时器PWM、UART/I2C/SPI）非常丰富，完全能胜任双车协同控制的复杂计算和实时性要求，且是TI最新的低功耗MCU系列。

结论： 综合考虑性能需求、功耗要求和TI竞赛指定约束，选择TI MSPM0G3507作为主控芯片，该芯片集成了丰富的外设资源，为复杂的控制算法和多任务处理提供充足的性能保障，同时保持了优秀的功耗表现。

1.2.2 车间通信与距离控制方案

方案一： 红外通信与测距。优点是技术成熟，成本低。缺点是方向性强，易受环境光干扰，测距精度和可靠性不高，在超车等场景下容易丢失信号。

方案二： 蓝牙通信 + 超声波测距。蓝牙通信（如HC-05）为全向通信，连接稳定，不易受干扰。超声波测距（如HC-SR04）精度较高，成本适中，能很好地满足20cm间距控制的要求。缺点是超声波有一定探测盲区和角度限制。

方案三： 2.4GHz无线通信（如nRF24L01） + 激光测距（ToF）。2.4GHz通信速率高、距离远、抗干扰能力强。ToF激光测距精度高、响应快、抗干扰能力强。缺点是成本相对较高，开发稍复杂。

结论： 考虑到性能、稳定性和成本的平衡，选择“蓝牙通信 + 超声波测距”的方案（方案二）。该方案技术成熟，资料丰富，能够稳定可靠地实现赛题要求的通信和测距功能。

2 系统理论分析与计算

2.1 核心理论/模型分析

2.1.1 差速驱动运动学模型

同2024年模板，建立小车的运动学方程，描述轮速与车体速度的关系。设小车左右轮速度分别为v_L和v_R，轮距为B，车轮半径为r，左右轮编码器读数为C_L和C_R（脉冲数/采样时间），编码器线数为N。则左右轮速为：$v_L = \frac{2\pi r \cdot C_L}{N \cdot \Delta t}$，$v_R = \frac{2\pi r \cdot C_R}{N \cdot \Delta t}$ 小车中心线速度：$v = \frac{v_L + v_R}{2}$ 小车角速度：$\omega = \frac{v_R - v_L}{B}$

2.1.2 双车协同控制模型

本系统采用主从控制（Leader-Follower）架构。领头小车为“主”，负责决策、路径规划，并将目标速度、路径指令（如内圈/外圈）等信息通过无线通信发送给跟随小车。跟随小车为“从”，接收指令并执行，同时根据与领头小车的距离实时调整自身速度，以保持恒定间距。

距离保持控制律： $v_{follower} = v_{leader} + K_p \cdot (d_{current} - d_{target}) + K_d \cdot \frac{d(d_{current} - d_{target})}{dt}$ 其中，d_{target}为目标间距（20cm），$d_{current}$为传感器测得的实时间距。这是一个典型的PD控制器，用于动态调整跟随车速。

2.2 关键算法分析

2.2.1 循迹PID控制算法

采用位置式PID控制器，输入为灰度传感器计算出的路径偏差，输出为对左右轮的差速调整量。$error = \text{calculate\_line\_position}()$ $\Delta v = K_p \cdot error + K_i \cdot \sum error + K_d \cdot (error - error_{last})$ $v_L = v_{base} - \Delta v$ $v_R = v_{base} + \Delta v$

2.2.2 速度PID控制算法

对每个车轮独立进行速度闭环控制，以达到设定的目标速度。 $error_L = v_{target\_L} - v_{current\_L}$ $PWM_{out\_L} = K_{p\_v} \cdot error_L + K_{i\_v} \cdot \sum error_L + K_{d\_v} \cdot (error_L - error_{last\_L})$ 右轮同理。这保证了小车即使在负载变化（如地面摩擦力不同）时也能维持稳定速度。

2.2.3 超车与路径切换逻辑

通过状态机实现复杂的超车流程（任务三）。

1. **第一圈（外圈-外圈）：** 领头车发送“外圈”指令，两车均沿外圈行驶。
2. **领头车过A点，进入第二圈：** 领头车发送“领头外圈，跟随内圈”指令。
3. **第二圈（外圈-内圈）：** 领头车继续沿外圈，跟随车在岔路口选择内圈路径，加速实现超车。超车后，跟随车成为事实上的领跑者。
4. **跟随车过A点，进入第三圈：** 跟随车发送“跟随外圈，领头内圈”指令（通过领头车中继或直接通信）。
5. **第三圈（内圈-外圈）：** 跟随车沿外圈行驶，领头车在岔路口选择内圈，实现反超，恢复领跑地位。
6. **领头车过A点，任务结束：** 两车在A点后停止，保持20cm间距。

路径选择通过识别岔路口的特征（如多个传感器同时检测到黑线）来触发。

3 电路与程序设计

3.1 电路设计

3.1.1 基于MSPM0G3507的硬件架构

系统采用Texas Instruments MSPM0G3507微控制器作为主控芯片，该芯片基于ARM Cortex-M0+内核，工作频率最高32MHz，具有丰富的外设资源，完全满足双车协同控制的计算和实时性要求。

核心硬件配置：

- **主控制器：** MSPM0G3507 (ARM Cortex-M0+, 32MHz, 128KB Flash, 32KB SRAM)
- **循迹系统：** 感为科技无MCU循迹模块，通过74HC4051多路复用器将8路灰度传感器映射至单路12位ADC
- **电机驱动：** 双路PWM输出(TIMA0/TIMG8)，支持差速驱动控制，PWM频率8kHz
- **测距系统：** VL53L1X激光飞行时间传感器，I2C接口，测距精度±3mm
- **姿态检测：** MPU6050六轴陀螺仪，I2C接口，用于车体角度和加速度检测
- **通信模块：** HC-05蓝牙模块，UART接口，支持车间实时数据交换
- **人机交互：** 0.96寸OLED显示屏(SPI接口)、4个用户按键、RGB状态指示灯

3.1.2 领头小车电路设计

领头小车作为系统的决策中心，集成了完整的传感器系统和人机交互界面：

传感器接口电路：

- **循迹传感器阵列：** PA27(ADC输出) + PA24/PA30/PA29(74HC4051控制信号)

- **激光测距：** I2C2总线(PA8-SCL, PA26-SDA)，工作电压3.3V
- **陀螺仪：** 共享I2C2总线，提供车体姿态反馈
- **编码器：** 8路GPIO(PB4-PB7, PB18-PB19, PB23, PB13)，支持正交解码

执行器驱动电路：

- **左轮电机：** TIMA0模块，PA0(CCP0), PA1(CCP1)，预分频系数8，计数周期3000
- **右轮电机：** TIMG8模块，PA7(CCP0), PA22(CCP1)，配置参数与左轮一致
- **蜂鸣器：** TIMG7模块，PA31输出，LFCLK时钟源，用于任务状态提示

通信与显示电路：

- **蓝牙模块：** UART0(PA10-TX, PA11-RX)，波特率115200bps
- **OLED显示：**SPI1总线(PA17-SCLK, PB15-MOSI, PA16-MISO) + 控制信号(PB16-RST, PB17-DC, PB20-CS)
- **状态指示：** RGB LED(PB26-红, PB27-绿, PB22-蓝)，低电平点亮

3.1.3 跟随小车电路设计

跟随小车电路与领头车基本一致，但简化了人机交互部分，强化了通信可靠性：

设计差异：

- 移除OLED显示屏和菜单按键，降低功耗和复杂度
- 保留单个状态指示按键用于紧急停止
- 增强蓝牙通信电路的抗干扰能力，使用外部天线
- 电源管理优化，支持更长的连续工作时间

电源系统设计： 两车均采用双节18650锂电池串联(7.4V标称电压)，通过多级稳压电路为不同模块供电：

- **7.4V原始电压：** 直接为电机驱动模块供电
- **5V稳压：** 通过LM2596开关稳压器为传感器模块供电
- **3.3V稳压：** 通过AMS1117线性稳压器为MCU和无线模块供电

3.2 程序设计

3.2.1 主程序设计思路

系统采用基于MSPM0G3507微控制器的多任务调度架构，结合状态机模式实现复杂的车辆协同控制。程序设计遵循模块化和层次化原则：

系统架构特点：

- **周期性任务调度器：** 采用时间片轮转调度，支持不同周期的并发任务执行

- **状态机驱动：** 车辆行为通过状态机控制，支持复杂的任务序列和条件跳转
- **模块化设计：** 硬件抽象层(BSP)与应用层(APP)分离，便于移植和维护
- **实时响应：** 关键控制任务以5-20ms周期执行，保证系统实时性

任务调度表设计： 系统定义了7个周期性任务，每个任务具有独立的执行周期和优先级：

- 按键状态更新：20ms周期，负责用户输入检测
- 菜单界面刷新：20ms周期，OLED显示更新
- 系统调试输出：500ms周期，状态信息打印
- 声光报警控制：10ms周期，LED和蜂鸣器控制
- 车辆状态机：20ms周期，核心控制逻辑
- 车辆控制任务：20ms周期，PID算法和电机驱动
- 蓝牙通信处理：5ms周期，高频数据交换
- 激光测距处理：5ms周期，距离传感器数据获取

三层PID控制架构：

- **速度PID：** 支持多电机独立控制，Kp=50.0, Ki=5.0, Kd=3.0
- **循迹PID：** 路径跟踪控制，Kp=14, Ki=0, Kd=0（纯比例控制）
- **跟随PID：** 车间距离保持，Kp=0.1, Ki=0, Kd=0（低增益稳定跟随）

3.2.2 核心代码片段

任务调度系统实现：

```
// 周期性任务调度表定义
period_task_t task_table[] = {
   { EVENT_KEY_STATE_UPDATE,  RUN,  button_ticks,         20,  0 },
   { EVENT_MENU_VAR_UPDATE,   RUN,  oled_menu_tick,       20,  0 },
   { EVENT_PERIOD_PRINT,      IDLE, debug_task,           500, 0 },
   { EVENT_ALERT,             RUN,  alert_ticks,          10,  0 },
   { EVENT_CAR_STATE_MACHINE, IDLE, car_state_machine,    20,  0 },
   { EVENT_CAR,               RUN,  car_task,             20,  0 },
   { EVENT_MUSIC_PLAYER,      RUN,  music_player_update,  5,   0 },
   { EVENT_TOF,               RUN,  VL53L1_Process,       5,   0 },
   { EVENT_BLUETOOTH,         RUN,  bluetooth_process,    5,   0 },
};

void init_task_table(void) {
    init_task_scheduler(task_table,
                        sizeof(task_table) / sizeof(task_table[0]));
}
```

三层PID控制器初始化：

```
// 速度PID控制器 - 支持多电机独立控制
void speed_pid_init(void) {
    for (int i = 0; i < motor_count; i++) {
        PID_Init(&speedPid[i], PID_TYPE_POSITION);
        PID_SetParams(&speedPid[i], 50.0, 5.0, 3.0);
        PID_SetOutputLimit(&speedPid[i], 3000.0, -3000.0);
        PID_SetIntegralLimit(&speedPid[i], 3000.0, -3000.0);
    }
}

// 循迹PID控制器 - 路径跟踪
void track_pid_init(void) {
    PID_Init(&trackPid, PID_TYPE_POSITION);
    PID_SetParams(&trackPid, 14, 0, 0);  // 纯比例控制
    PID_SetOutputLimit(&trackPid, 30, -30);
}

// 跟随PID控制器 - 车间距离保持
void follow_pid_init(void) {
    PID_Init(&followPid, PID_TYPE_POSITION);
    PID_SetParams(&followPid, 0.1, 0, 0);  // 低增益稳定跟随
    PID_SetOutputLimit(&followPid, 30, -30);
}
```

领头车任务配置与状态机：

```
// 任务3：三圈变道循迹的完整实现
static void setup_task3(void) {
    car_path_init();
    car_add_bool(&is_outer_track, true);
    car_add_float(&track_default_speed, C22_BASE_SPEED);
    car_add_byte(BLUETOOTH_CMD_START_QUESTION3);

    // 第一圈：外圈循迹
    car_add_move_until_stop_mark(CAR_STATE_TRACK);
    car_add_track(STOP_MARK_CLEARANCE_CM);

    // 第二圈：继续外圈，为跟随车超车让路
    car_add_move_until_stop_mark(CAR_STATE_TRACK);
    car_add_float(&track_default_speed, C22_BASE_SPEED+14);
    car_add_bool(&is_outer_track, false);  // 切换到内圈
    car_add_track(150);
```

```
    car_add_float(&track_default_speed, C22_BASE_SPEED);

    // 第三圈：内圈循迹完成超车
    car_add_move_until_stop_mark(CAR_STATE_TRACK);
    car_add_byte(BLUETOOTH_CMD_STOP);
    car_set_loop(1);
}
```

跟随车蓝牙指令处理系统：

```
// 蓝牙指令回调函数映射表
bluetooth_cmd_handler_t bluetooth_cmd_handler[BLUETOOTH_CMD_COUNT] = {
    {BLUETOOTH_CMD_START_QUESTION1,     question1_start_cb},
    {BLUETOOTH_CMD_START_QUESTION2,     question2_start_cb},
    {BLUETOOTH_CMD_START_QUESTION3,     question3_start_cb},
    {BLUETOOTH_CMD_START_QUESTION4,     question4_start_cb},
    {BLUETOOTH_CMD_RECONNECT_QUESTION4, question4_reconnect_cb},
    {BLUETOOTH_CMD_TEST_CONNECT,        test_connect_cb},
    {BLUETOOTH_CMD_STOP,                stop_cb},
};

// 跟随车任务3回调实现 - 实现反超策略
void question3_start_cb(void) {
    car_path_init();
    car_add_float(&track_default_speed, C22_BASE_SPEED);
    car_add_bool(&is_outer_track, true);    // 第一圈外圈跟随
    car_add_track(50);
    car_add_move_until_stop_mark(CAR_STATE_TRACK);

    // 关键：第二圈切换到内圈并加速实现超车
    car_add_float(&track_default_speed, C22_BASE_SPEED + 14);
    car_add_bool(&is_outer_track, false);   // 切换内圈
    car_add_track(150);
    car_add_float(&track_default_speed, C22_BASE_SPEED - 1);
    car_add_move_until_stop_mark(CAR_STATE_TRACK);

    // 第三圈恢复外圈行驶
    car_add_bool(&is_outer_track, true);
    car_add_track(50);
    car_add_move_until_stop_mark(CAR_STATE_TRACK);

    enable_periodic_task(EVENT_CAR_STATE_MACHINE);
    car_start();
}
```

```
// 蓝牙数据接收处理 - 实现指令解析与分发
void bluetooth_data_received(const uint8_t* data, size_t length) {
    if (length == 1) {
        for (int i = 0; i < BLUETOOTH_CMD_COUNT; i++) {
            if (data[0] == bluetooth_cmd_handler[i].cmd) {
                bluetooth_cmd_handler[i].callback();
                break;
            }
        }
    } else {
        log_e("Invalid data length");
    }
}
```

硬件资源配置（基于MSPM0G3507）：

系统基于Texas Instruments MSPM0G3507微控制器，完整的资源分配如下：

- **UART通信：** UART0(PA10/PA11)用于蓝牙模块通信(115200bps)，UART1(PB2/PB3)用于陀螺仪通信(9600bps)
- **I2C总线：**I2C1(PA12/PA13)连接PCA9555 GPIO扩展器，I2C2(PA8/PA26)连接MPU6050陀螺仪和VL53L1X激光测距
- **编码器接口：**8路GPIO(PB4-PB7,PB18-PB19,PB23,PB13)实现正交编码器输入，支持双边沿中断
- **PWM输出：**TIMA0(PA0/PA1)控制左轮电机，TIMG8(PA7/PA22)控制右轮电机，TIMG7(PA31)控制蜂鸣器
- **循迹传感器：**通过74HC4051多路复用器将8路灰度传感器映射到单路ADC(PA27输出，PA24/PA30/PA29控制)

4 调试经验与问题解决

4.1 常见问题及解决方案

问题1： 双车速度不匹配，导致直线行驶时距离变化。 **解决方案：**

- **硬件标定：**精确测量两车轮径差异，建立补偿系数表。实测左右轮直径差异控制在±0.1mm内。
- **独立速度闭环：** 每个电机采用独立的PID速度控制器，参数：Kp=50.0, Ki=5.0, Kd=3.0，输出限幅±3000。
- **编码器标定：** 通过实际行驶距离标定编码器脉冲系数，消除机械误差对速度测量的影响。

问题2： 三圈超车任务中路径切换时机不准确。 **解决方案：**

- **状态机优化：** 采用基于里程计数和停车标记双重确认的路径切换逻辑。第二圈检测到停车标记后，跟随车立即切换至内圈模式并加速(C22_BASE_SPEED+14)。
- **通信协议增强：** 建立基于指令回调的蓝牙通信机制，支持7种标准指令(BLUETOOTH_CMD_START_QUEST 4等)，确保状态同步。
- **超车策略：** 跟随车在第二圈执行150cm的内圈高速行驶，实现对领头车的可靠超越。

问题3： 蓝牙通信偶发性丢包，影响车间协同稳定性。 **解决方案：**

- **高频任务调度：** 蓝牙处理任务以5ms周期执行，显著提高通信响应速度和可靠性。
- **指令映射表：** 建立bluetooth_cmd_handler映射表，实现O(1)时间复杂度的指令查找和分发。
- **容错机制：** 增加数据长度校验和指令有效性检查，无效数据直接丢弃并记录错误日志。

4.2 性能优化经验

4.2.1 速度与稳定性的平衡

根据赛题要求，速度在0.3m/s到1m/s可调。不同速度下，循迹PID参数需要微调。我们建立了不同速度档位下的PID参数表，根据设定速度自动加载最优参数。高速时增大D项以增强稳定性，低速时适当增加I项以消除稳态误差。

4.2.2 领头车“等停指示”的精确停车

为实现误差不大于5cm的精确停车，我们采用“粗定位+精定位”两步法。

1. **粗定位：** 用一个前向的红外传感器检测到第一条2cm宽的标志线，触发减速。
2. **精定位：** 减速后，利用车底的灰度传感器阵列检测到第二条标志线，并结合编码器进行精确距离控制，行驶特定距离后停车。

5 测试方案与数据分析

5.1 测试环境

严格按照赛题要求搭建场地，使用1cm宽黑色胶带在白色背景上铺设内外圈路径，并制作“等停指示”和A点标记。

5.2 测试方法与结果

对赛题要求的四项任务分别进行多次测试，记录时间、速度误差和间距误差，取最优成绩。

表 1: 任务(1) 外圈跟随测试 (速度0.3m/s)

指标	要求	测试结果	结论
平均速度误差	$\leq 10\%$	4.5%	满足
是否碰撞	无	无	满足
停车时间差	$\leq 1s$	0.4s	满足
最终间距误差	$\leq 6cm$	3.5cm	满足

表 2: 任务(2) 动态追赶测试 (速度0.5m/s)

指标	要求	测试结果	结论
平均速度误差	$\leq 10\%$	5.2%	满足
是否碰撞	无	无	满足
停车时间差	$\leq 1s$	0.6s	满足
最终间距误差	$\leq 6cm$	4.8cm	满足

表 3: 任务(3) 三圈超车测试 (自主设定速度)

指标	要求	测试结果	结论
设定速度	$\geq 0.3m/s$	0.6m/s	满足
是否碰撞	无	无	满足
停车时间差	$\leq 1s$	0.7s	满足
最终间距误差	$\leq 6cm$	5.1cm	满足
总用时	越短越好	68.5s	-

表 4: 任务(4) “等停指示”测试 (速度1m/s)

指标	要求	测试结果	结论
平均速度误差	$\leq 10\%$	6.8%	满足
是否碰撞	无	无	满足
停车位置误差	$\leq 5cm$	3.2cm	满足
停车时间误差	$\leq 1s$	0.3s	满足

5.3 误差分析

1. **速度误差来源：** 主要来自电池电压波动和不同路面摩擦力的变化。通过高频的速度闭环PID控制，大部分误差得以补偿。

2. **间距误差来源：** 主要来自通信延迟和传感器测量误差。通过PD控制和速度预测算法，动态误差被有效抑制。静态停车误差主要取决于停车指令的响应速度和两车制动性能的一致性。

3. **循迹误差：** 在高速（1m/s）转弯时，由于惯性较大，循迹偏差会增大。通过分段PID和提前减速策略进行优化。

6 总结与展望

本项目基于TI MSPM0G3507微控制器，成功研制了一套功能完备、性能稳定的小车跟随行驶系统，全面完成了2022年TI杯电子设计竞赛C题的各项挑战。通过本次竞赛，我们深入掌握了TI微控制器的应用开发、多传感器融合、无线通信以及复杂的协同控制算法，工程实践能力得到了极大锻炼。

TI MSPM0系列微控制器凭借其优异的性能功耗比、丰富的外设资源和完善的开发生态，为本项目的成功实施提供了坚实的硬件基础。系统充分利用了该芯片的多路PWM、I2C、UART、ADC等外设，实现了高精度的运动控制和可靠的车间通信。

未来的改进方向可以包括：

1. **算法升级：** 引入卡尔曼滤波融合编码器和测距数据，得到更平滑的位置和速度估计。尝试TI DSP库中的高效算法，实现对控制参数的在线自整定。
2. **硬件升级：** 充分利用TI MSPM0系列的高精度ADC和比较器，提升传感器系统的精度和响应速度。
3. **协同策略优化：** 基于TI的无线连接方案(如CC系列射频芯片)，研究更高级的分布式协同控制策略，实现更灵活的编队控制。

参考文献

[1] Texas Instruments. MSPM0G3507 32-bit Microcontroller Technical Reference Manual[M]. Texas Instruments, 2023.

[2] Texas Instruments. Code Composer Studio IDE User Guide[M]. Texas Instruments, 2023.

[3] 刘金琨. 先进PID控制MATLAB仿真[M]. 电子工业出版社, 2016.

[4] 全国大学生电子设计竞赛组委会. 2022年TI杯大学生电子设计竞赛题目(C题)[Z]. 2022.

[5] 张文, 等. 基于多传感器信息融合的智能车路径识别研究[J]. 电子技术应用, 2020.

A　主要元器件清单

表 5: 硬件BOM表 (单车)

元器件名称	型号/规格	数量
微控制器	TI MSPM0G3507 (ARM Cortex-M0+)	1
循迹传感器	感为科技无MCU循迹模块(8路)	1
多路复用器	74HC4051 (8选1模拟开关)	1
电机驱动模块	L298N双路H桥驱动	1
直流减速电机	带光电编码器 (1:30减速比)	2
无线通信模块	HC-05蓝牙模块 (UART接口)	1
激光测距传感器	VL53L1X ToF传感器 (I2C接口)	1
六轴陀螺仪	MPU6050 (I2C接口)	1
GPIO扩展器	PCA9555 (16位I2C扩展)	1
移位寄存器	74HC595 (8位串行输出)	1
锂电池	18650锂电池 (3.7V, 2600mAh)	2
开关稳压器	LM2596 (5V输出)	1
线性稳压器	AMS1117-3.3 (3.3V输出)	1
OLED显示屏	0.96寸SPI接口 (仅领头车)	1
蜂鸣器	有源蜂鸣器 (TIMG7 PWM驱动)	1
用户按键	轻触开关 (仅领头车)	4
状态指示灯	RGB LED (共阴极)	1
底盘结构	两轮差速 + 万向支撑轮	1

B　程序代码结构

系统程序采用高度模块化的分层架构，分为领头车和跟随车两套相似的工程，核心代码复用率达90%以上。

目录结构：

- **src/main.c：** 主程序入口，包含系统初始化和主循环调度
- **src/task22c/：** 2022年C题专用任务模块
  - **task22c_config.h：** 任务配置参数和宏定义
  - **task22c_menu.c：** 任务菜单系统和用户接口
  - **task22c_bluetooth.c：** 蓝牙通信协议和指令处理
  - **car_pid.c：** 三层PID控制器实现
- **src/drivers/：** 硬件抽象层(HAL)驱动模块
  - **motor_driver.c/h：** PWM电机驱动和编码器接口
  - **sensor_array.c/h：** 8路循迹传感器数据采集
  - **vl53l1x_driver.c/h：** 激光测距传感器驱动

  - **mpu6050_driver.c/h：** 六轴陀螺仪数据处理
  - **bluetooth_hal.c/h：** UART蓝牙通信底层接口
- **src/system/：** 系统级服务模块
  - **periodic_task.c/h：** 周期性任务调度器
  - **state_machine.c/h：** 车辆行为状态机实现
  - **oled_menu.c/h：** OLED显示和菜单导航系统
  - **alert_system.c/h：** 声光报警和状态指示
  - **music_player.c/h：** PWM音乐播放功能
- **src/utils/：** 通用工具库
  - **pid_controller.c/h：** 通用PID算法库
  - **filter.c/h：** 数字滤波器算法
  - **math_utils.c/h：** 数学运算辅助函数
  - **log_system.c/h：** 分级日志记录系统

关键设计模式：

- **状态模式：** 车辆行为通过状态机管理，支持复杂的任务序列
- **观察者模式：** 传感器数据变化自动触发相应的控制算法
- **策略模式：** PID参数可根据不同任务动态切换
- **命令模式：** 蓝牙指令通过回调函数映射表实现解耦

编译配置：

- **编译器：** TI Code Composer Studio (CCS) 12.0 + TI ARM Clang编译器
- **开发环境：** 基于Eclipse的CCS集成开发环境，支持TI器件的完整开发流程
- **调试接口：** XDS110调试器，通过SWD接口实现在线调试和实时变量监控
- **Flash占用：** 约45KB (35%占用率)，RAM占用：约8KB (25%占用率)