2024 年全国大学生电子设计竞赛

自动行驶小车（H 题）

2024 年 8 月 1 日

摘　要

本设计报告介绍了一个基于 TI MSPMO 系列 MCU 控制的自动行驶小车的设计和实现过程。该自动行驶小车以 TI MSPMOG3507 MCU 为核心，能够在预设路径上自动行驶，并且在行驶过程中能够实现声光提示和路径循迹功能。设计过程采用步进电机作为驱动系统，并采用八路灰度传感器进行循迹，使用 4.3 寸串口屏提供人机交互界面。通过分析小车在自动行驶过程中可能产生误差的来源，提出了相应的解决方案和优化措施。在路径控制方面采用了分段控制的方法，通过调整电机左右轮的加速度和速度，实现了小车的直线行驶和转弯循迹。最后通过一系列的测试方法，对小车的直线行驶、曲线行驶及循迹效果等方面进行了测试和分析，完成了基于 TI MSPMO 系列 MCU 控制的自动行驶小车。系统能够按照预定的轨迹在规定时间内完成任务，并进行声光提示。

关键词：MSPMOG3507，循迹，自动行驶

1. 系统方案设计与比较

1.1 系统设计思路

自动行驶小车的设计思路主要包括硬件设计和软件设计两大部分。硬件设计方面，选择 TI MSPMOG3507 MCU 作为控制核心。这款芯片具有高性能、低功耗的特点以及较多的外设接口，非常适合嵌入式实时控制系统。我们采用步进电机作为驱动系统，八路灰度传感器用于路径循迹，通过 GPIIO 口读取的信号来与 MCU 通信。人机交互界面使用 4.3 寸串口屏，用户可以通过触摸屏选择不同的任务模式，并且使用有源蜂鸣器来提示任务完成状态。

软件设计方面，主要包括电机控制算法、路径控制算法和人机交互界面的开发。电机控制算法确保小车能够按照预设的路径精确行驶，根据灰度传感器数据来动态调整小车的行驶路线；人机交互界面提供了友好的用户操作平台，用户可以通过触摸屏轻松控制小车的运行逻辑。

1.2 控制核心的可行性论证

MSPMOG3507 MCU 采用了 ARM Cortex-MO+内核，主频高达 80MHz，能够处理复杂的实时控制任务，确保小车在行驶过程中快速响应传感器数据并做出调整。该芯片具有低功耗特性，支持多种低功耗模式，可以在不影响性能的情况下显著降低能耗。这对于依靠电池供电的小车尤为重要，延长了小车的续航时间。此外，MSPMOG3507 内置多个定时器、ADC、DAC、UART、I2C、SPI 等外设接口，能够满足小车控制系统中各模块的通信需求。如该任务中与灰度传感器的通信，及通过 UART 接口可以实现与串口屏的通信。

TI 芯片以其高可靠性和稳定性著称，能够在各种复杂环境下稳定运行。这对于自动行驶小车来说至关重要，因为在实际应用中，系统可能会面临多种不确定因素，可靠性高的控制芯片能够提高系统的整体稳定性和安全性。因此，选择 MSPMOG3507 作为控制核心是本次设计的最佳选择。

1.3 各功能块的划分与组成

系统主要划分为控制模块、驱动模块、传感器模块、显示模块这几个部分。

控制模块由 MSPMOG3507 MCU 及其外围电路组成，是整个系统的核心。上电后，MCU 首先进行系统初始化，包括 GPIO、定时器、UART 接口等的初始化，确保各个外设能够正常工作。循迹过程中，控制模块会不断读取灰度传感器的数据传输到 MCU，MCU 根据传感器数据计算小车的当前状态，并根据预设的路径规划来控制步进电机的转动。同时，控制模块还负责与串口屏进行通信，实现人机交互。

驱动模块由步进电机及其驱动电路组成，是小车运动的动力来源。步进电机接收来自 MCU 的控制信号，按照预设的步进方式进行转动，驱动小车前进。

传感器模块由八路灰度传感器组成，用于检测地面上的黑线。灰度传感器通过检测地面反射的光线强度来判断黑线的位置，并将数据传输给 MCU，MCU 根据传感器数据调整小车的行驶方向。八路灰度传感器能够实时获取小车行驶过程中的路径信息，帮助小车在黑线轨迹上准确行驶。

显示模块由 4.3 寸串口屏组成，用于提供人机交互界面。串口屏通过 UART 接口与 MCU 通信，显示小车的状态信息和操作界面。用户可以通过触摸屏选择不同的任务模式，启动或停止小车。串口屏具有高分辨率和广视角，能够清晰显示操作界面，提高用户体验。

提示模块由有源蜂鸣器组成，用于在小车完成特定任务或到达指定位置时进行声光提示。当小车到达预定位置或完成某项任务时，MCU 通过 GPIO 控制有源蜂鸣器发出声音提示。蜂鸣器的频率和音量可以根据需要进行调整，确保提示音清晰可辨。

2. 理论分析

2.1 误差分析

误差主要来源于以下几个方面：电机性能、小车重心偏移、编码器精度、传感器检测误差和环境因素。

电机性能直接影响小车的运动轨迹。步进电机具有较高的精度，但在实际运行中，电机的驱动电流、电压和负载变化都会对其性能产生影响。例如，步进电机在负载变化时可能会出现失步现象，导致小车运动轨迹偏离预设路径。为了解决这一问题，我们选择了高精度、高扭矩的步进电机，提高电机的平稳性和定位精度，尽可能减少电机运行中的误差。

小车重心偏移也是影响运动精度的重要因素。由于小车各部件的重量分布不均匀，重心位置可能会发生偏移，导致小车在转弯或加速时发生倾斜和偏移。为了解决这个问题，我们在设计过程中对小车的重心进行了精确测量和校正。通过合理布局电池、传感器和电机的位置，确保小车重心尽可能接近几何中心，减少重心偏移对运动轨迹的影响。

传感器检测误差也是需要考虑的一个重要方面。八路灰度传感器用于检测地面上的黑线，实现路径循迹功能。然而，由于光照变化、地面反射率差异等因素，传感器的检测结果可能会出现误差。为了解决这一问题，我们在设计中对传感器进行了反复多次的校准，确定各传感器的基准值和阈值，从而确保在不同光照条件下都能准确检测到黑线。

2.2 轨迹控制

在路径控制策略方面，我们采用了分段控制的方法，将小车的运动过程分为直线行驶和曲线转弯两个部分。直线行驶时，通过控制两侧步进电机的速度一致，实现小车的直线行驶。在加速和减速过程中，精确控制电机的步进频率，确保小车在规定时间内到达目标位置。曲线转弯时，通过控制左右轮的速度差，实现小车的曲线运动。在转弯过程中，实时检测运动轨迹误差，动态调整电机控制参数，确保小车沿着预定的轨迹行驶。

2.2.1 直线行驶

在直线行驶过程中，小车需要在最短时间内从一个点移动到另一个点。例如从 A 点到 B 点。为了实现这一目标，我们通过控制两侧步进电机的速度一致，使小车沿直线行驶。具体步骤如下：

加速阶段：小车从静止状态开始，以一定的加速度启动。加速度的大小通过实验确定，以确保小车能够平稳加速而不失步。步进电机的步进频率逐渐增加，直到达到预设的行驶速度。

匀速行驶阶段：在达到预设速度后，小车以恒定速度行驶一段时间。这个时间通过实验确定，使得小车刚好能在经过这一段时间后从 A 点到达 B 点。

减速阶段：接近终点时，小车以一定的减速度减速。减速度的大小同样通过实验确定，以确保小车能够平稳减速并在 B 点停下。步进电机的步进频率逐渐降低，直到小车完全停止。

在整个直线行驶过程中，控制系统实时监测步进电机的运行状态，确保两侧电机的步进频率一致，避免因速度差异导致的小车偏移，系统应能根据实时检测到的运动误差，来调整电机的控制参数，确保小车沿直线准确行驶。

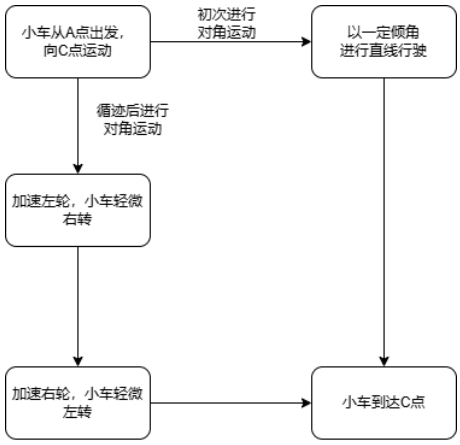
2.2.2 对角线行驶

在对角线行驶过程中，小车需要根据预设的转弯半径进行轨迹控制。例如，从 A 点到 C 点。如果初始位置是 A 并且直接到 C 点，可以在初始摆放时让小车对准 C 点；如果小车在循迹完后到了 A 点，然后再向 C 点运动，可以让小车走两段弧线。具体步骤如下：

第一段弧线：小车从 A 点出发，加速左轮，使小车向右微微转动。通过控制左轮的步进频率高于右轮，实现小车向右转弯。转弯的半径和角度通过实验确定，确保小车能够准确进入第二段弧线的起点。

第二段弧线：小车在第一段弧线结束后，加速右轮，使小车向左转动。通过控制右轮的步进频率高于左轮，实现小车向左转弯。转弯的半径和角度同样通过实验确定，确保小车能够准确到达 C 点。

在整个对角线行驶的过程中，应根据实时检测到的运动误差，来调整电机的运行参数（如两段轨迹的运行时长等），以确保小车沿预设的弧线轨迹准确行驶。



图：对角行驶算法原理（以A到C为例）

2.2.3 路径循迹

路径循迹是小车在弯道处自动行驶的重要功能。系统采用八路灰度传感器实现路径循迹，通过检测地面上的黑线位置，动态调整小车的行驶方向。具体步骤如下：

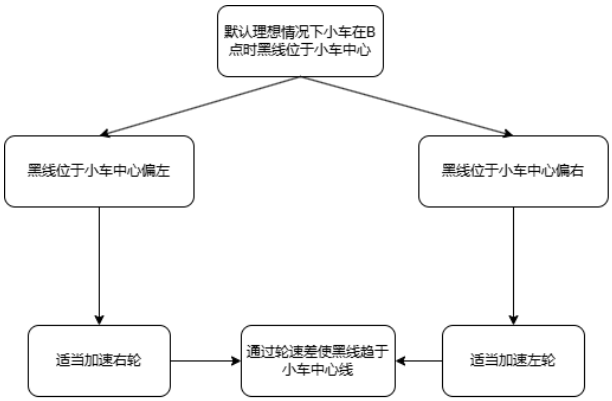
传感器数据读取：控制系统不断读取八路灰度传感器的数据，判断黑线的位置。传感器的数据传输给 MSPM0G3507，实时更新黑线位置。

位置判断：根据传感器数据，系统判断黑线相对于小车的位置。如果黑线位于小车的中间，说明小车沿预设轨迹行驶；如果黑线偏离中间位置，说明小车发生了偏移。当检测到黑线偏离中间位置时，系统根据偏移量调整左右轮的速度。具体调整策略如下：

黑线偏左：加速右轮、减速左轮，使小车向左转动，重新回到黑线中间位置。

黑线偏右：加速左轮、减速右轮，使小车向右转动，重新回到黑线中间位置。

控制系统应根据传感器数据的实时变化，不断调整左右轮的速度和方向，确保小车始终沿黑线行驶。在路径循迹过程中，同样需要根据实时检测到的误差，调整小车运动的参数，以提高循迹的准确性和稳定性。



图：循迹算法原理（以B到C为例）

3. 电路与程序设计

3.1 步进电机驱动电路

步进电机驱动电路是小车运动控制的核心部分。步进电机的选择主要考虑其扭矩、步距角和额定电压电流等参数。为了保证小车在不同负载条件下都能稳定运行，我们选择了一款高扭矩、低噪音的步进电机。步进电机驱动电路采用专用步进电机驱动芯片，该芯片可以实现更加平滑的步进控制。我们选择的步进电机驱动芯片具有如下特点：

内置电流控制：通过调节电流感应电阻，可以设置电机的驱动电流，确保电机在不同负载条件下都能稳定运行。

过流保护和过热保护：提高了驱动电路的可靠性，防止电机和驱动芯片因过载而损坏。

3.2 灰度传感器电路

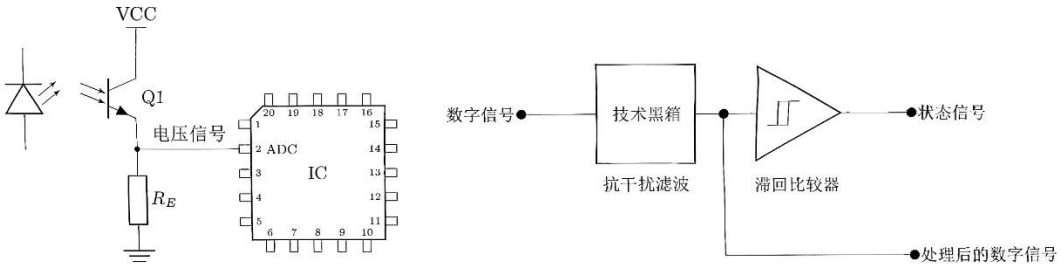
灰度传感器用于检测地面上的黑线，实现路径循迹功能。该系统采用八路灰度传感器，通过 GPIO 口与控制核心通信。灰度传感器的选择主要考虑其灵敏度、响应时间和抗干扰能力。

灵敏度：选择灵敏度高的灰度传感器，以确保在不同光照条件下都能准确检测到黑线。

响应时间：选择响应时间小于 1ms 的传感器，以提高系统的实时性。

抗干扰能力：选择具有良好抗干扰能力的传感器，减少环境光线变化对检测结果的影响。

灰度传感器模块由八个独立的灰度传感器组成，每个传感器通过 GPIO 口与 MSPM0G3507 通信。传感器模块的原理图如下：



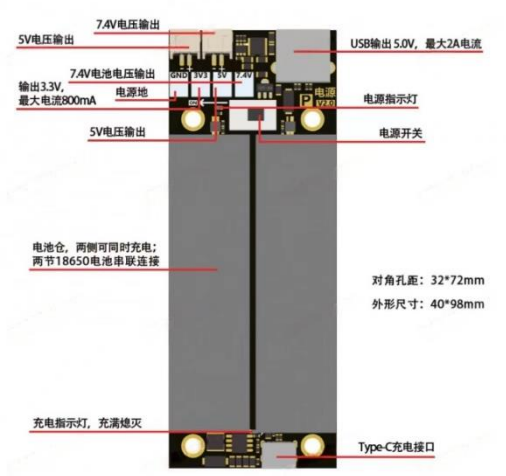
3.3 控制核心电路

控制核心电路是整个系统的控制中心，由 MSPM0G3507 MCU 及其外围电路组成。控制核心电路的设计主要包括电源管理、时钟配置、通信接口和外设控制等部分。

电源管理

MSPM0G3507 的工作电压为 1.8V~3.6V，电源管理电路采用降压电源模块，供控制核心和传感器模块使用。降压模块选择了一款高效率、低噪声的型号，以提高系统的电源利用效率和稳定性。

电池装配电路设计如下：



通信接口

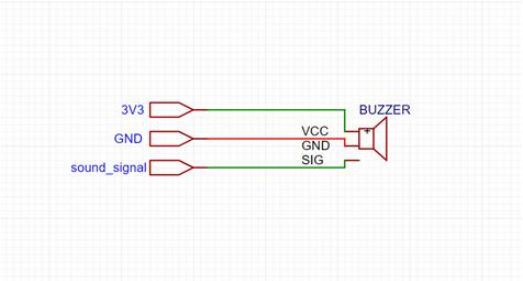
控制核心电路需要与多个外设进行通信，包括步进电机驱动、灰度传感器、串口屏和蜂鸣器等。通信接口的配置如下：

UART 接口：用于与串口屏通信，实现人机交互界面。

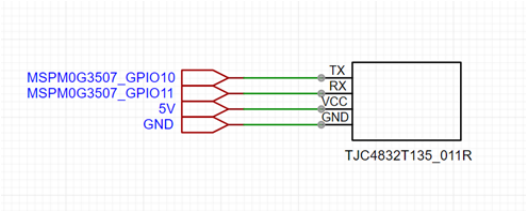
GPIO 接口：用于与灰度传感器模块通信，实时获取传感器数据，控制步进电

机驱动信号和蜂鸣器的启停。

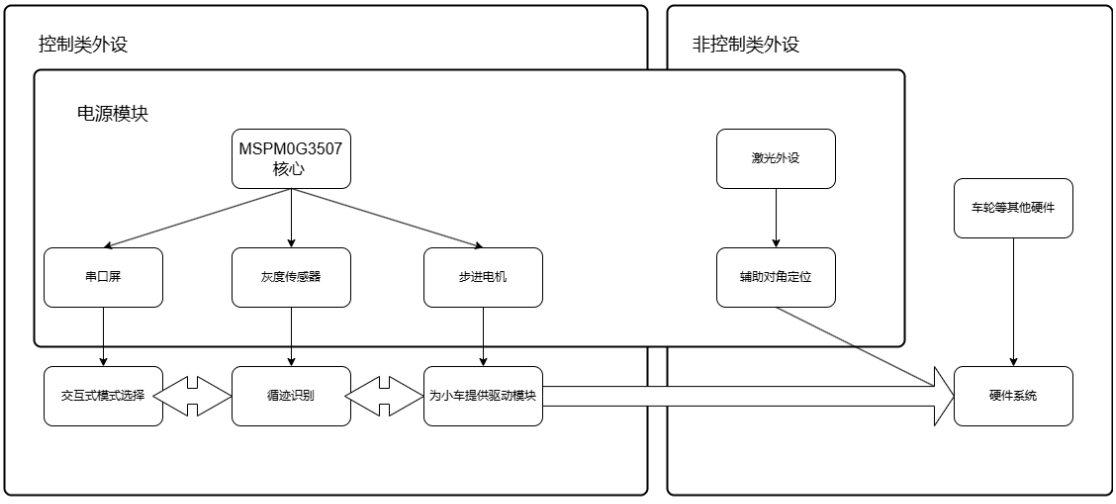
声音提示模块电路



显示屏模块电路



系统架构



图：小车车身架构

3.4 程序流程

为了实现小车的自动行驶、路径循迹、声光提示等功能，我们在程序设计中采用了模块化编程的思想，将整个系统分为多个功能模块，包括初始化模块、传感器读取模块、路径规划与控制模块、人机交互模块和声光提示模块等。本部分将详细介绍各个模块的设计思路和实现方法，并通过流程图来说明程序的整体结构。

3.4.1 初始化模块

初始化模块负责系统上电后的初始化工作，包括 MSPM0G3507 MCU 的初始化、外设接口的配置、传感器的校准和驱动电路的初始化等。初始化模块的设计目标是确保系统在启动时能够正确配置各个外设，并进入正常工作状态。系统初始化包括时钟配置、GPIO 配置、和外设接口初始化等。以下是初始化模块的主要步骤：

时钟配置：设置系统时钟频率，确保 MCU 和外设工作在合适的频率下。

GPIO 配置：配置各个外设的 GPIO 引脚，包括步进电机控制引脚、传感器接口引脚和蜂鸣器控制引脚等。

外设接口初始化：初始化 UART 接口等，确保用户输入等数据能够正确传输。

3.4.2 传感器读取模块

传感器读取模块负责从八路灰度传感器获取实时数据，并将数据传输给路径规划与控制模块。为了提高数据读取的效率和准确性，我们采用 GPIO 口进行传输信号，读取传感器数据。传感器数据读取的主要步骤包括启动 GPIO 口通信，通过 GPIO 口读取八路灰度传感器的数据。

3.4.3 路径规划与控制模块

路径规划与控制模块是自动行驶小车的核心部分，负责根据传感器数据计算小车的运动轨迹，并通过控制步进电机实现小车的行驶和转弯。该模块采用 PID 控制算法，通过实时调整电机的驱动参数，确保小车沿着预定路径行驶。路径规划的主要步骤包括从传感器读取模块获取实时数据，根据传感器数据计算小车与预定路径的偏差，从而调整电机的驱动参数（左右轮转速等）。

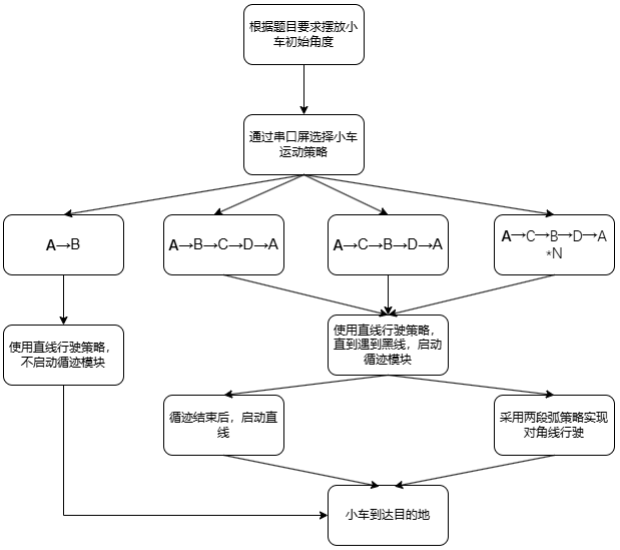
3.4.4 人机交互模块

人机交互模块负责用户与系统之间的交互，通过串口屏提供操作界面，用户可以选择不同的任务模式，启动或停止小车。该模块通过 UART 接口与 MSPM0G3507 通信，接收用户的输入指令，并显示小车的状态信息。用户输入处理的主要步骤为通过 UART 接口接收用户在串口屏上的操作指令，随后解析用户指令，根据指令内容执行相应的操作。

3.4.5 声光提示模块

声光提示模块用于在小车完成特定任务或到达指定位置时进行声光提示。该模块通过控制有源蜂鸣器发出声音提示，提醒用户当前任务状态。提示系统应能，检测小车的任务状态，判断是否需要进行提示，根据任务状态控制蜂鸣器的启停和声音频率，在蜂鸣器中播放提示音，提醒用户当前状态。

各个模块组合后的小车整体运行流程图如下：



图：小车控制系统架构

4. 系统测试

4.1 测试方案

为了全面验证自动行驶小车的功能和性能，我们制定了详细的测试方案，包括正常的功能性测试及小车不同运动情况的测试。测试环境设置在平整的白色地面上，无其他干扰标记，包括一个 220cm×120cm 的白色喷绘布场地，两个对称的半圆弧线（半径为 40cm，线宽 1.8cm），以及四个标记点（A、B、C、D）。场地布置示意图如下：

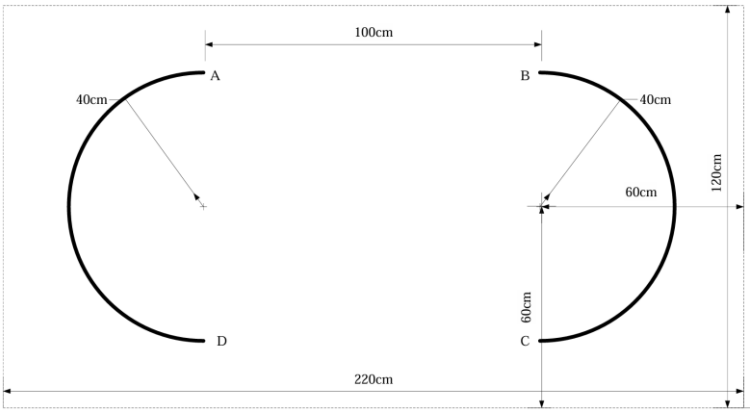


图 小车行驶场地示意图

测试设备包括万用表、秒表等，用于检测电机驱动信号、传感器输出和小车的运动状态，运动时间。测试项目包括直线行驶测试、曲线行驶测试、路径循迹测试、声光提示测试等这几个部分。每个测试项目分为多个测试步骤，以

确保测试覆盖的全面性。

4.2 测试步骤

4.2.1 直线行驶测试

测试目的：验证小车从 A 点到 B 点的直线行驶性能，检测小车能否在规定时间内到达目标位置，并进行声光提示。

测试步骤：

将小车放置在 A 点，调整初始位置，使小车对准 B 点。

启动小车，开始计时。

观察小车的行驶轨迹，记录小车到达 B 点的时间。

检测小车到达 B 点后是否进行声光提示。

测试数据记录：

小车到达 B 点的时间。

小车行驶过程中的轨迹偏差。

声光提示的响应时间。

4.2.2 循迹测试

测试目的：验证小车在八路灰度传感器的辅助下，沿地面黑线进行路径循迹的性能，检测小车能否稳定地沿黑线行驶，并进行声光提示。

测试步骤：

在场地上布置黑线，形成一条弯曲的路径。

将小车放置在路径起点，启动小车。

观察小车沿黑线行驶的轨迹，记录小车运行时间。

检测小车经过标记点时是否进行声光提示。

测试数据记录：

小车沿黑线行驶的时间。

小车行驶过程中的轨迹偏差。

每个标记点的声光提示响应时间。

4.2.3 全路径行驶测试

测试目的：验证小车沿弧线从 A 点到 B，C，D 点，再从 D 点回到 A 点的全

路径行驶性能，检测小车能否在规定时间内完成一圈，并进行声光提示。

测试步骤：

将小车放置在 A 点，调整初始位置，使小车对准 B 点或 C 点。

启动小车，开始计时。

观察小车沿弧线行驶的轨迹，记录小车经过 C 点、D 点和回到 A 点的时间。

检测小车经过每个标记点时是否进行声光提示。

测试数据记录：

小车经过 C 点、D 点和回到 A 点的时间。

小车行驶过程中的轨迹偏差。

每个标记点的声光提示响应时间。

4.2.4 声光提示测试

测试目的：验证小车在到达指定位置或完成特定任务时，声光提示的准确性和及时性。

测试步骤：

启动小车，设置目标任务（如到达 B 点或完成一圈行驶）。

观察小车完成任务后的声光提示。

记录声光提示的响应时间和准确性。

测试数据记录：

声光提示的响应时间。

声光提示的准确性（是否在正确的时间和位置提示）。

4.3 测试结果

通过系统测试，我们得到了以下主要测试结果：

4.3.1 直线行驶测试结果

行驶时间：小车从 A 点到 B 点的平均行驶时间为 2-3 秒，符合规定的 15 秒以内的要求。

轨迹偏差：小车在直线行驶过程中，最大轨迹偏差为±1.0cm，整体轨迹稳定。

声光提示：小车到达 B 点后，声光提示的响应时间为 0.5 秒，提示准确。

4.3.2 循迹测试结果

行驶时间：小车沿黑线路径行驶的平均时间为 2-3 秒，路径循迹效果良好。

轨迹偏差：小车在路径循迹过程中，最大轨迹偏差为±1.5cm，整体路径稳定。

声光提示：小车经过标记点时，声光提示的响应时间为 0.5 秒，提示准确。

4.3.3 全路径行驶测试结果

行驶时间：小车沿弧线行驶一圈（A→B→C→D→A）的平均时间为 8-10 秒，符合规定的 30 秒以内的要求。

轨迹偏差：小车在曲线行驶过程中，最大轨迹偏差为±2.0cm，整体轨迹平稳。

声光提示：小车经过 C 点和 D 点时，声光提示的响应时间均为 0.6 秒，提示准确。

4.3.4 声光提示测试结果

响应时间：小车在到达指定位置或完成任务后的声光提示响应时间为 0.5 秒，提示及时。

提示准确性：声光提示的准确性为 100%，在每次测试中均能准确提示。

通过对测试数据的分析，我们验证了设计方案的可行性和程序实现的有效性。小车在直线行驶、曲线行驶和路径循迹等方面表现稳定，声光提示及时准确。测试结果表明，基于 TI MSPM0 系列 MCU 的自动行驶小车设计实现了预期功能，具有较高的稳定性和可靠性。

5. 结论

通过本文所述的设计，我们成功完成了一款基于 TI MSPM0 系列 MCU 控制的自动行驶小车。总的来说，该系统采用高精度的步进电机，确保了小车的稳定行驶，使用八路灰度传感器来保证小车对路径的精准识别，用 4.3 寸串口屏提供了便捷的人机交互界面，有源蜂鸣器实现了声光提示。我们分析了影响小车自动行驶的误差来源，并提出解决方案和优化措施，通过分段控制调整左右电机的加速度和速度，实现了小车在直线和曲线循迹行驶过程中的稳定性。基于 TI 开发环境，编写运动控制算法和人机交互界面程序，确保了整个系统的高效运行。并通过对小车的直线、循迹行驶等方面进行了全面测试和分析，最终得到了满意的系统性能指标。本文验证了 TI MSPM0G3507 MCU 在自动行驶小车中的应用效果，证明了其在嵌入式实时控制系统中的优越性。

6. 参考文献

[1]沈建华，杨艳琴，MSP430 超低功耗单片机原理与应用[M].北京：清华大学出版社，2013

[2]胡寿松.自动控制原理[M].6 版.北京：科学出版社，2013

[3]童诗白，华成英.模拟电子技术基础[M].4 版.北京：高等教育出版社，2009

[4]张友德，赵志英，涂时亮.单片机微型机原理，应用与实践[M].5 版.上海:复旦大学 出版社，2009

7.附录

小车左右轮控制代码及循迹代码：

```
void Motor_Control(int left_speed, int right_speed)
{
	if(left_speed < 0)
	{
		Motor_Vel_Control(1, 1, -left_speed, 0, 1);
		delay_us(200);
	}
	else
	{
		Motor_Vel_Control(1, 0, left_speed, 0, 1);
		delay_us(200);
	}
	if(right_speed < 0)
	{
		Motor_Vel_Control(2, 0, -right_speed, 0, 1);
		delay_us(200);
	}
	else
	{
		Motor_Vel_Control(2, 1, right_speed, 0, 1);
```

```
            delay_us(200);
        }
        Motor_Synchronous_motion(0);
        delay_us(200);
}

volatile bool diverse;
volatile uint8_t flip;

void track(uint8_t vu, uint8_t vb)
{
    if (L4==1&&L3==1&&L2==1&&L1==1&&R1==1&&R2==1&&R3==1&&R4==1)
    {
        Motor_Control(vb, vb);
    }
    else if (diverse)
    {
        if (L4==0&&L3==0&&L2==0&&L1==0&&R1==0&&R2==0&&R3==0&&R4==0)
        {
            Motor_Control(vb+10*vu, vb);
        }
        else if (L1==0||R1==0)
        {
            Motor_Control(vb+3*vu, vb+vu);
        }
        else if (L4==0)
        {
            Motor_Control(vb, vb+vu);
```

```
    }
    else if (R4==0)
    {
        Motor_Control(vb+5*vu, vb+vu);
    }
    else if (L2==0||L3==0)
    {
        Motor_Control(vb+2*vu, vb+vu);
    }
    else if (R2==0||R3==0)
    {
        Motor_Control(vb+4*vu, vb+vu);
    }
}
else
{
    if (L4==0&&L3==0&&L2==0&&L1==0&&R1==0&&R2==0&&R3==0&&R4==0)
    {
        Motor_Control(vb, vb+(10+flip)*vu);
    }
    else if (L1==0||R1==0)
    {
        Motor_Control(vb+vu, vb+(3+flip)*vu);
    }
    else if (L4==0)
    {
        Motor_Control(vb+vu, vb+(5+flip)*vu);
    }
```

```
        else if (R4==0)
        {
            Motor_Control(vb+vu, vb);
        }
        else if (L2==0||L3==0)
        {
            Motor_Control(vb+vu, vb+(4+flip)*vu);
        }
        else if (R2==0||R3==0)
        {
            Motor_Control(vb+vu, vb+(2+flip)*vu);
        }
    }
    delay_ms(10);
}
```