太原工業学院

毕 业 设 计

基于 ARM Linux 环境的智能房屋安全控制系统设计

学生姓名：段旭　　学号：235033101

系　　部：自动化系

专　　业：电气工程及其自动化

指导教师：朱珊（讲师）

二〇二五年五月

诚信声明

本人郑重声明：

所呈交的毕业论文，是本人在导师的指导下，独立进行研究工作所取得的成果。除文中已经注明引用的内容外，本论文不包含任何其他个人或集体已经发表或撰写过的作品成果。对本文的研究做出重要贡献的个人和集体，均已在文中以明确方式标明。本人完全意识到本声明的法律结果由本人承担。

本人签名：　　年　　月　　日

基于 ARM Linux 环境的智能房屋安全控制系统设计

摘 要

本系统基于全志 H616 处理器平台，以香橙派 Zero2 为主控单元，构建了一套模块化的智能家居安全管控体系。系统集成了可燃气体泄漏检测、视频监控、人脸识别开锁、家电设备状态显示与远程控制等功能，采用 MQ-2 气体传感器、OLED 显示屏、高清摄像头以及 SU-03T 语音模块，满足多场景应用需求。软件部分采用多线程并发机制，分别实现语音监听、网络通讯、MQTT 协议数据交换、环境监测及报警等功能，并通过 MJPG-Streamer 实现摄像头实时视频流推送，保证系统可实现高效与实时的运行。同时根据配套的安卓应用，用户可远程对设备进行状态检测与控制，进一步提升了系统的人机交互体验及智能化水平。整体设计兼顾实用性、安全性与可扩展性，拥有良好的应用潜力与推广价值。

关键词：可燃气体检测，ARM Linux，嵌入式，人脸识别

Design of Smart Home Security Control System Based on ARM Linux Environment

Abstract

This system is built on the Allwinner H616 processor platform, with the Orange Pi Zero2 serving as the main control unit, and forms a modular intelligent home security management solution. It integrates multiple functions including combustible gas leak detection, video surveillance, face recognition unlocking, appliance status display, and remote control. By incorporating the MQ-2 gas sensor, OLED display, high-definition camera, and SU-03T voice module, the system supports a variety of smart home application scenarios. The software adopts a multithreaded concurrent processing mechanism to achieve voice monitoring, network communication, MQTT protocol data exchange, environmental monitoring, and alert functionality. Real-time video streaming is enabled through MJPG-Streamer, ensuring efficient and responsive system performance. In addition, with the support of a dedicated Android application, users can remotely monitor and control devices, greatly enhancing the system's interactivity and level of intelligence. Overall, the system balances practicality, security, and scalability, demonstrating strong application prospects and potential for wide adoption.

Keywords: Combustible Gas Detection, ARM Linux, embedded system, face recognition

目 录

第1章 绪 论

1.1 研究目的及意义

伴随科技的急速发展，通信技术、嵌入式技术与物联网技术持续融合，大幅改变了人们的生产和生活样式。智能家居作为智慧生活的关键组成，正从设想走进现实，慢慢踏入普通家庭的生活范畴，而且在高端住宅里获取了初步应用。借助对家庭环境、家电设备、安全防护等方面实施集成控制与智能管理，智能家居系统有效地提升了居住环境的安全性，舒适度和便利度，提高了用户生活的效率与品质。

现阶段市面上大量智能家居解决方案依旧存在不少难题，例如成本高、布线复杂、维护不易、系统扩展性差以及用户操作门槛较高等情况，这些问题在中低端家庭里体现得尤为明显，制约了智能家居技术的普及和推广[1]。如何开发出一套成本合适、功能实用、操作简单、系统开放性和扩展性良好的智能控制系统，成为智能家居研究领域亟待处理的关键难题。

本设计根据 ARM 开发板和 Linux 操作系统平台实施，拟定了一个智能房屋安全控制体系，系统采用模块化的设计方式，使用传感器对室内环境状态实时监控，同时完成对家用电器和安防设备的智能控制与状态反馈。系统的核心部分把嵌入式 Linux 平台当作控制中心，还借助本地通信接口把各类设备连接起来；用户可凭借基于 Android 平台开发的手机客户端完成远程控制、状态查询等操作，真正做到“人在外面逛，家可被掌控”的目标。

1.2 国内外研究现状

1.2.1 国外研究进展

国外智能家居起步较早，技术相对成熟，20 世纪 70 年代，X10 通信协议的问世拉开了智能家居早期探索的序幕。自 21 世纪开启后，物联网推动着智能家居的进步，Google、Amazon、Apple 等公司按顺序推出智能产品，例如 2011 年出现的 Nest 恒温器、2014 年出现的 Amazon Echo 智能音箱，采用云计算和语音交互进行结合，增进家庭自动化水平，人工智能、大数据、边缘计算等技术推动了智能家居生态的进一步优化，拉动智能安防、智能照明和智能家电进行深度融合[2]。

目前国外研究重点集中在智能交互、家居安防、能源管理和隐私保护上，Google

Assistant、Amazon Alexa 这类 AI 语音助手应用十分广泛，支持采用自然语言加以控制，Google Nest、Ring 等公司推出相关智能摄像头与智能门锁，采用人脸识别和移动检测技术，实现远程的监控以及自动报警。智能家居在能源管理上获得突破，例如 Nest 恒温器能根据用户日常习惯智能调节，增强能源的利用率[3]。与此同时，隐私保护成为聚焦研究方向，一些企业采用端到端加密、区块链等相关技术，以提升用户数据的安全性。当技术不断积累且商业模式逐步走向成熟，智能家居在国外逐渐从实验室走进生活，其研究逐步达到了系统化、规模化[4]。

国外智能家居研究重点多集中在系统平台统一、生态搭建以及用户体验优化等方面。以像美国、德国、日本这样的国家为代表，研究机构与企业针对智能家居的“平台化集成”“场景智能协同”“用户行为建模”等内容展开深度探究，突出系统的稳定性、兼容性以及智能决策能力，不少高校以及科研机构也对智能家居的人机交互界面、行为识别机制、安全性建模等课题进行了长期摸索，为智能家居系统的智能程度和实用效果提供了理论支撑[5]。

国外研究同样十分关注智能家居的开放性与标准化打造，期望打破不同厂商设备之间的阻隔，增进系统的可扩展性及互操作性。国外针对智能家居系统架构设计、统一协议标准以及智能化服务等方面的研究起步较早、基础扎实，已构建起较为完善的理论体系与实践框架，取得了显著效果。

1.2.2 国内研究进展

我国在智能家居领域起步相对较晚，但是近些年来发展势头迅猛，已逐步缩小与发达国家的差距。尤其是在产业推动及应用落地方面有显著的进步，国内智能家居研究大多集中在控制系统与单点功能的实现。例如灯光调控、安防警报等，伴随“智慧城市”“物联网+”等国家战略的推进，不少高校、科研机构和企业开始把目光投向智能家居的整体系统设计与智能协同控制。

国内智能家居研究大多围绕系统集成、平台互联、人机交互优化、智能控制逻辑等方向开展。若干高校、科研院所陆续搭建起智能家居实验平台，就系统模型搭建、控制策略规划直至平台搭建开展了系统性考察，国内科研越发关注用户体验与场景适配，凭借嵌入式开发、数据建模等方式提升系统在动态环境下的智能化响应能力与适应性性能[5]。

就应用推广而言，国内企业也积极介入到智能家居的研究与开发进程中，依靠与

科研单位的协同合作，加快智能家居系统在居家生活、社区管理等场景的大规模布置，当下国内研究依旧有标准不统一、系统比较分散、智能化水平参差不齐的问题，理论体系的完善程度欠佳，应在技术整合、平台架构、安全策略等方面进一步加强力度。

1.3 本文主要研究的内容

本论文设计了一套基于香橙派 Zero2 的智能房屋安全控制系统，采用简单工厂设计模式搭建软件架构，支持通过 Android 客户端实现远程控制与环境实时监测。系统将人脸识别模块与可燃气体泄漏检测报警模块整合起来，切实提升了住宅的安全防护能力。香橙派 Zero2 具备较强的处理能力，结合工厂模式设计，增加了系统的扩展性及灵活性，用户可依靠移动端随时对家庭环境开展监测及管理工作，守护人身跟财物安全。整体系统的结构合理，在增强智能家居智能化水平与便捷性的同时，为用户打造更安全、舒适的居住体验。

第2章 系统总体设计方案

本设计的智能房屋安全控制系统以全志的 H616 处理器作为主控核心，依据其强大的处理能力和多样的外设接口，形成了一个具有高度集成性以及良好拓展效果的智能控制平台。系统在整体上采用模块化设计以及简单工厂模式的软件架构，大幅提升了程序的可维护性及可扩展性，方便后期功能扩展与系统升级。

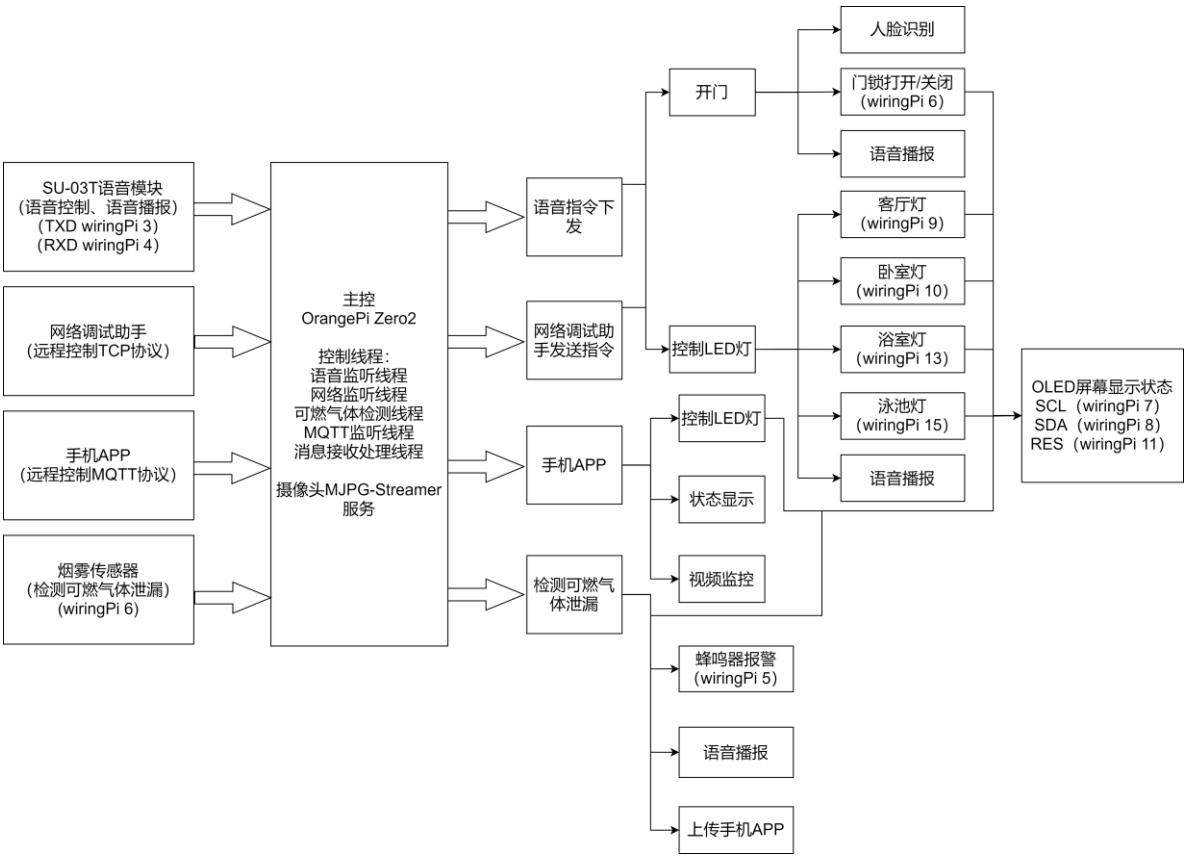

图 2.1　总体方案设计框图

本设计对应智能房屋安全控制系统的设计框图如图 2.1 所示，系统实现的核心功能包括：可燃气体泄漏检测、视频监控查看状态、人脸识别解锁途径、远程控制、家电设备状态呈现等。作为用户交互主要界面，安卓 APP 客户端支持远程状态查看与对设备进行控制操作，用户随时可借助手机对家中设备远程操作，优化了系统的实用性以及安全特性。功能上，系统规划了多个功能线程以达成多任务并行处理。包括：语音监听线程、网络监听线程、可燃气体检测线程、MQTT 监听线程以及消息队列监听线程。各线程协同配合运行，保证系统在高并发和复杂环境里的稳定性和响应高效性。

系统采用集成 SU-03T 语音模块的方式，实现语音指令识别与语音播报功能，用户可依靠语音模块对家庭照明、门锁等设备实施控制，进而获取实时语音反馈提醒。

为强化用户居住的安全水平，系统采用了人脸识别模块实现智能门锁的操控，识别无误后自动完成门锁的开闭动作，防止因传统钥匙丢失产生的安全隐患。系统添加了可燃气体传感器，实时去检测环境中的可燃气体浓度，若检测到有气体泄漏，会马上启动报警机制，采用 OLED 屏幕状态展现、语音播报和 APP 消息通告三重方式联合提示，最大程度守护用户的生命财产安全。

系统基于 MJPG-Streamer 搭建了视频监控服务，用户可借助 APP 实时查看家庭摄像头的画面，增强系统的监控效果。系统另外设计了 OLED 显示模块，用于本地实时呈现门锁状态、照明状态和报警提示等关键资讯，强化了系统的人机交互体验。

系统支持三种主要控制方式：一是采用语音控制模式，可凭借语音实现设备的操作与状态播报；二是采用 TCP 的远程控制模式，经由网络调试助手传送控制指令；三是采用手机 APP 控制模式，APP 通过 MQTT 协议达成对设备的远程控制及状态同步，保证数据及时性与操作的便捷性。

本智能房屋安全控制系统对硬件设计、软件架构与功能集成进行了系统化、模块化设计，实现了多种智能化管控和可靠的安全守卫。系统拥有良好的扩展性、操作灵活度及高安全性，可为用户提供更智能、便捷的相关服务安全的现代化居住环境。

第3章 系统硬件设计

3.1 硬件电路设计

本系统选用香橙派 Zero2 作为主控平台，设计了一套面向家庭安全场景的智能控制系统。系统围绕高性能、低功耗、易扩展的设计目标，集成了语音识别、气体泄漏报警、继电器控制、OLED 状态显示、蜂鸣器报警等多种硬件模块，形成了功能完整、结构清晰的智能家居硬件控制框架。如图 3.1 所示为本系统的总体硬件连接图。

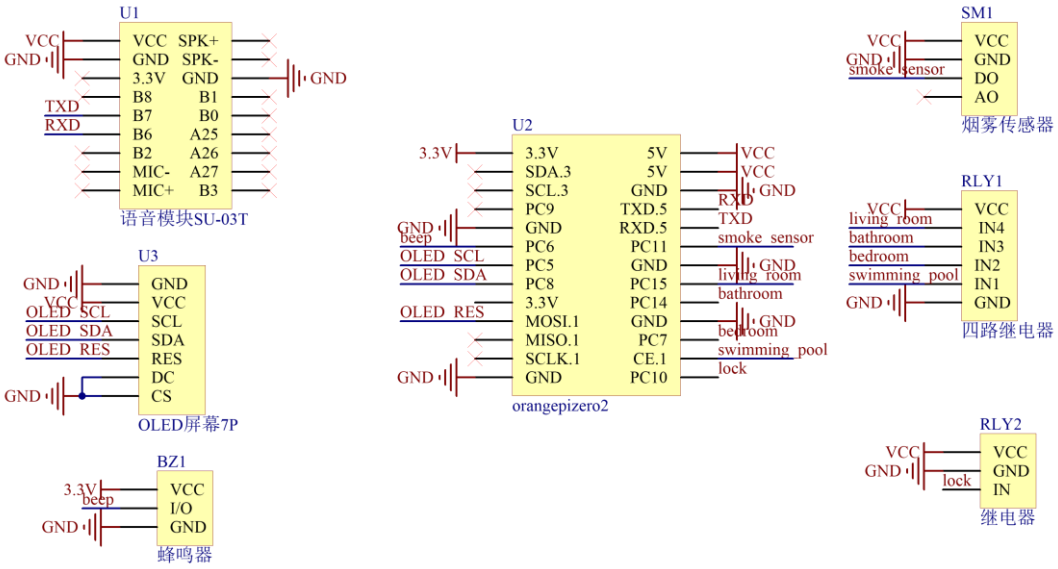

图 3.1　硬件接线原理图

3.2 主控制器

如图 3.2 所示本系统选择以全志 H616 芯片为基础的香橙派 Zero2 作为主控平台。H616 是一款有着高性价比且功耗较低的四核 64 位 Cortex - A53[7]处理器，最高可实现 1.5GHz 的主频。可运行 Linux、Debian 等操作系统，具备较好的多任务处理实力，适合在智能家居里对计算性能与稳定性要求高的场景。比起传统的 8 位或 32 位单片机，H616 在性能及能效方面优势突出，能同时实现语音识别、人脸识别、设备控制、远程通信等多项功能，维持系统高效运转。

香橙派 Zero2 搭载着 1GB DDR3 内存与 Mali G31 MP2 GPU，拥有图像处理以及视频流播放的能力，能符合系统对图像识别和本地显示的需求。凭借低功耗特性，系统适合 24 小时长时间工作，减少能耗及相关维护费用，增添系统的可靠性及实用价

值。

香橙派 Zero2 有着丰富的接口资源，有 26 个 GPIO 引脚、3 个 USB 接口、HDMI 输出、千兆以太网，再加上可选的 Wi-Fi 和蓝牙模块，便于和各类传感器及外设相连接。用户可结合实际需求增添如温湿度检测、照明控制、门禁管理等功能，以此让系统呈现智能化与多样化。该平台为智能房屋安全控制系统搭建了性能强、能耗低、扩展性不错的理想硬件基础。

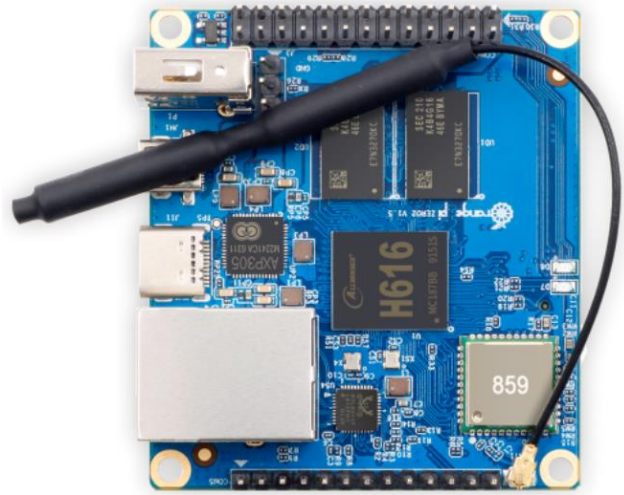

图 3.2　香橙派 Zero2 正面视图

3.3 可燃气体检测

智能房屋安全控制系统当中采用 MQ-2 可燃气体传感器模块，用于实时探测环境里是否有液化气、天然气、酒精、氢气、烟雾等可燃气体，MQ-2 模块拥有灵敏度高、响应迅速且高效、稳定性良好、抗干扰能力强、驱动电路简洁等优点，普遍应用在家庭、工厂等场景的气体泄漏预警系统。

如图 3.3 所示该传感器所采用的核心材料是二氧化锡半导体，属于表面离子型 N 型导体，当处于工作所需的温度时，二氧化锡吸附氧分子进而形成负离子，引起表面电子的数量下降，一旦与可燃气体碰到一起，氧离子跟气体起化学反应放出电子，使得导电能力增强、电阻降低，于是产生跟气体浓度相关的电压变化。

MQ-2 模块输出有模拟信号，本系统主要根据其 DO 引脚输出逻辑电平与主控芯片连接起来，若气体浓度超过所设定的阈值，比较器向外输出低电平信号，主控接收信号后引发蜂鸣器报警动作，同时借助网络把警报信息推到用户的手机 APP 上，达成

实时状态下的远程预警，模块所设电位器可调节比较器的门槛电压，进而调节传感器灵敏度以匹配不同应用环境。

MQ-2 具有良好的重复性与长期稳定性，正常开展检测的浓度范围为 100～10000ppm，在电路设计这个阶段，依靠限流电阻保护传感器加热丝，保障其长时间维持稳定工作，整体设计使得系统在复杂环境之中能精确识别气体泄漏并迅速回应，提升家庭安全层级。

图 3.3　MQ-2 模块实物图

3.4 OLED 显示模块

在智能家居系统当中，OLED 模块凭借其出色显示性能，成为不可缺失的组成部分。通过自发光特性，OLED 屏幕可展现对比度高、颜色鲜艳的清晰画面，即使在光线欠佳的环境中也可稳定显示，如图 3.4 所示。其具备的低功耗优势，使它成为理想的信息显示方案。

图 3.4　OLED 模块实物图

在系统运行的进程内，OLED 模块作为核心交互界面，主要负责设备状态提示、信息呈现等职能。在对设备实施控制期间，OLED 模块会实时展示各类信息，例如门锁的开闭状态、灯光开关状态、烟雾传感器的状态等等，联合语音播报，达成视觉及听觉的双重反馈，增进交互效果。

OLED 屏幕不光支持静态文字显示，还可借助简单图标或动画强化信息传达，其具备的广视角、快速响应和高分辨率进一步充实了显示内容，让系统更洋溢科技感，作为连接用户跟系统的关键桥梁，OLED 模块对智能家居系统智能化和实用性的提升起到了有力支持。

3.5 摄像头模块

摄像头模块在智能家居系统当中同样起到关键作用。作为视觉采集的核心，它凭借其高效的图像采集能力，为系统提供了重要的数据后盾。项目采用了 200 万像素高清摄像头，能清晰采集环境中的图像信息，为人脸识别、安防监测的功能提供图像根基，该摄像头采用 USB 接口和系统连接在一起，如图 3.5 所示，实现了图像数据的高速传输及稳定处理，进一步提升了系统整体的性能与反应速度。

图 3.5　摄像头模块实物图

为达成实时监控的功效，系统添加了开源的 MJPG-Streamer 库，把摄像头采集的视频流编码成 MJPEG 格式之后网络传送，采用此技术途径，用户可利用手机 App 或浏览器实时观看家里的监控画面，不需要额外的设备，把开发流程进行了简化，还改善了系统的易用性与部署的效率。

使用高像素摄像头与 MJPG-Streamer 结合起来，系统实现了高质量图像的采集，并满足了视频流高效传输和稳定播放需求。系统采用 USB 接口也有效降低了功耗，保证系统在长时间内稳定运行，利用这种途径，用户可以随时知晓家里情形，系统的智能化程度与安全性大幅跃升，提供了更为流畅且可靠的用户体验。

3.6 语音模块

语音模块作为智能家居系统里人机交互的主要中枢，为用户提供了一种既自然又便捷的控制方式，采用语音指令，用户无需碰到设备就可实现操作，改善了使用的便捷水平，也更贴合居家、老人或特殊人群的应用需求。

系统采用 SU-03T 型号的语音识别模块，具备良好的可配置性与兼容性。此模块支持经由网页完成配置，支持用户自主定义关键词与唤醒词，供电方式有多种可选，可使用 USB 接口来供电，也能使用 VCC 与 GND 引脚连接到主板，硬件连接简单方便，如图 3.6 所示。

图 3.6　语音模块实物图

在实际的应用操作里，用户可采用语音控制灯光、门锁、空调等智能设备，例如说出“打开客厅灯”，会触发语音模块经串口朝主控设备发送指令，主控板接收到信号后执行对应的控制逻辑，系统同样支持语音反馈这一功能，如播报“已为您打开客厅灯”，增强交互的体验感。

语音模块的加入显著提升了智能家居系统的操作便捷性和智能化程度，使用户在日常生活中能够以更自然的方式完成各种控制任务，为家庭环境带来更高的科技感与人性化体验。

第4章 系统软件设计

4.1 主程序设计

本系统凭借面向对象的编程思想来构造整体架构，主程序是系统运行的初始入口，承担着系统初始化、各模块加载以及线程调度等关键技术问题，其执行的流程如图 4.1 所示。

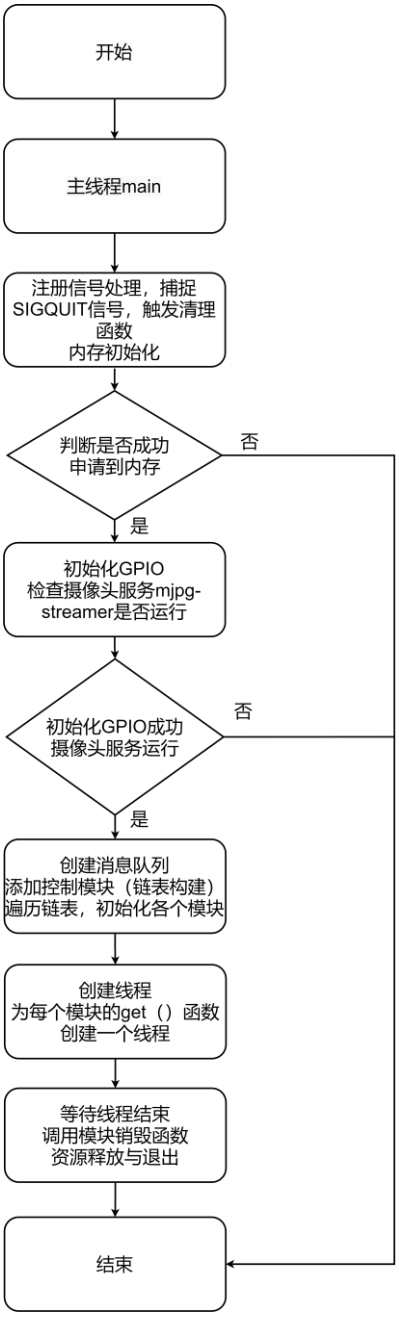

图 4.1　主函数执行流程图

程序进入主函数 main 开展操作，注册信号处理函数，用以捕获特定信号，当系统运行期间一旦接收到这个信号，将去执行相应的清理函数，保障资源安全释放，实现程序的退出，该机制增加了系统的健壮性与可维护性。

程序在执行过程中完成了必要的内存初始化操作，确保各模块数据空间正确配置并满足运行要求。若内存申请未获成功，系统会停止运行，在这一阶段直接退出。

内存初始化结束后，程序开始实施对 GPIO 的初始化，并查看摄像头服务是否正常运行。GPIO 控制是本系统开展外部设备通信的重要接口，而摄像头服务是实现人脸识别与视频监控等功能的先决条件。若 GPIO 初始化失败，或者摄像头服务未运行，系统将发出错误提示然后退出，以防在设备未准备就绪的情形下进行后续操作。

待硬件和服务模块准备就绪后，程序开启模块初始化阶段。一开始就创建消息队列结构，并利用链表形式添加各个控制模块。采用链表结构使模块的添加及维护变得更灵活，系统借助遍历链表，逐个去完成所有模块的初始化工作。

按照链表里面模块节点的数量，为每一个模块分别创建线程，线程中实施对应模块 get()函数的操作，以实现对模块状态的实时掌控与获取，该设计大大提升了系统运行的并发性与响应速度。

主线程完成创建线程的任务后，进入等待状态，直至全部子线程运行结束，主线程调用模块自带的资源释放函数，释放各类动态分配出去的内存资源，且对硬件接口实施关闭，完成系统的彻底退出。

依靠上述流程设计，系统在启动、运行、退出等各个阶段均具备明晰的控制机制，保障各模块实现协调运转，为后续功能的拓展及维护提供了良好的结构前提。

4.1.1 工厂模式架构设计

本系统软件结构采用简单工厂模式，在保证各模块职责不冲突的同时，兼顾到了良好的可维护及可扩展性。

系统整体划分成输入控制端工厂与输出执行端工厂这两大类：输入控制端工厂统一处理来自手机 APP、语音识别模块、网络调试助手和烟雾传感器等设备的指令输入和环境状态检测事宜；输出执行端工厂管理灯光、蜂鸣器、电磁门锁等执行设备的动作响应工作。

为了达成统一治理与灵活拓展，系统把全部控制模块和设备模块以结构体链表的形式组织实施管理，各个节点代表着一个功能实体，包含模块初始化步骤、运行处理

步骤、状态反馈步骤以及资源释放步骤等接口，主程序借助遍历链表，自动实现模块注册及线程创建，保障各模块独立并行开展，进而达成高效的资源利用和任务管理[8]。

系统采用链表方式来管理模块，各个模块保持相对独立，并通过统一的数据结构对接口进行标准化定义，从而简化了模块的管理与扩展，提升了系统的灵活性和可维护性。整体软件架构体现出模块化、并发性与可扩展性的设计理念，结合简单工厂模式、结构体链表管理方式与统一接口规范，不仅有效提升了开发效率与系统稳定性，也为后续功能拓展、设备的建通以及跨平台适配提供了有力支持。

4.1.2 智能家居同步机制设计

为实现各模块并行高效运行的效果，同时保障资源访问时的安全性，智能房屋安全控制系统采用以多线程为基础的架构设计。还结合互斥锁以及条件变量机制，达成了线程彼此间的互斥与同步。该设计不仅提升了系统总体的运行效率与稳定性，还稳妥保障了多线程环境下操作的精准无误与协调一致。

在并发执行的环境里，多个线程有可能同时去访问或者修改共享资源。例如设备状态、控制指令和消息队列的相关数据，若没有高效的同步机制，十分容易引发数据不一致、竞态条件等情况。系统在草共享资源有关的操作中引入互斥锁，保障同一时刻只有一个线程能去访问临界资源。一个线程成功获取锁，其余线程会进入阻塞状态，直到这一锁被释放，由此可有效防止数据冲突与系统出现异常。

系统采用条件变量来达成线程间的状态通知及同步控制。条件变量允许线程在特定条件不满足时自动进入阻塞，待条件达成，由另外的线程发送信号唤醒再继续执行。进而实现线程间高效的协同配合。例如消息队列监听线程正在等待消息数据，假如队列为空的话，它就会一直处于挂起状态，防止 CPU 资源出现空耗。一旦别的线程把新消息写入队列，就使用条件变量通知消息队列监听线程进行唤醒处理，极大增加了系统资源的利用效率和响应速度。

整个系统规划了 5 个主要控制线程以及若干动态生成的设备处理线程，5 个主要控制线程分别为：语音监听线程、网络监听线程、MQTT 监听线程、消息队列监听线程、烟雾报警监听线程。每个控制线程负责特定的功能模块，通过统一的消息队列与同步机制实现信息交互与控制指令的流转。如图 4.2 所示下边具体描述消息队列监听线程设计和语音监听线程设计。

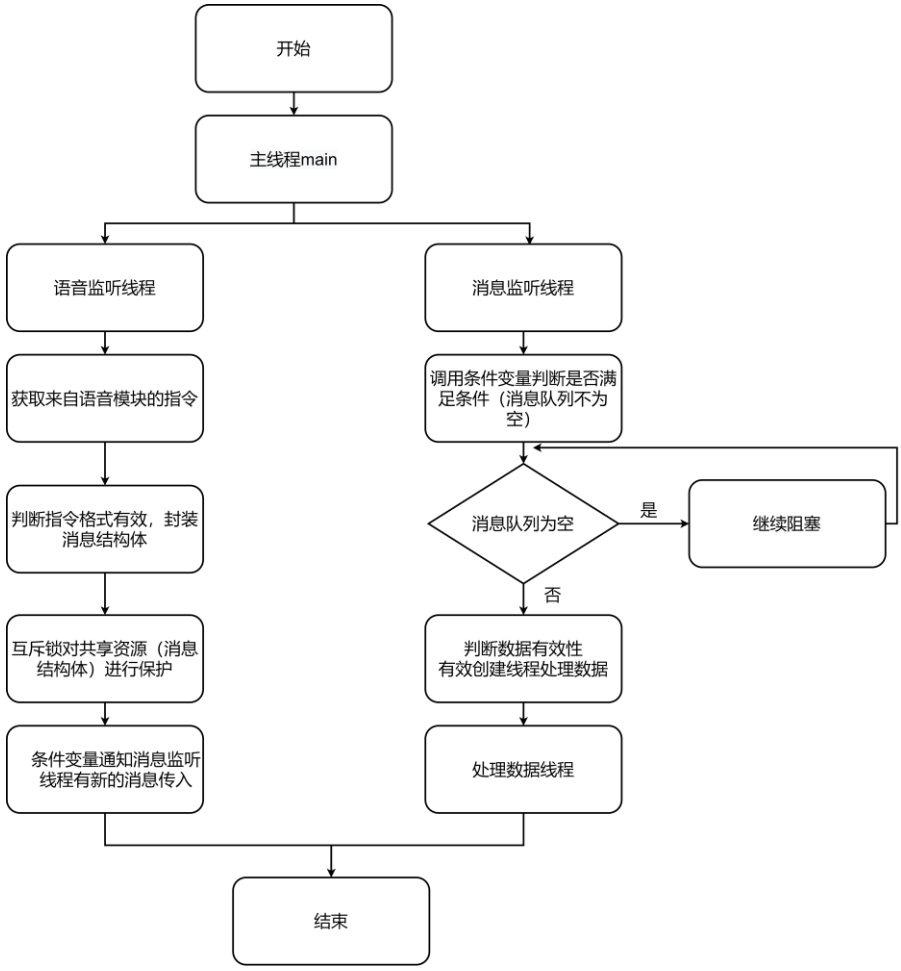

图 4.2　多线程模块设计方案图

每个控制线程负责特定的功能模块，通过统一的消息队列与同步机制实现信息交互与控制指令的流转。下边具体描述消息队列监听线程设计和语音监听线程设计。

语音监听线程主要负责采集并处理用户通过语音输入的控制指令。其工作流程为：一开始在线程内部调用专门的接口函数，得到语音识别模块回传的数据。接着对所接收到的数据做基本的格式检查，若指令格式合法且内容切实有效，则把数据封装为标准的消息结构体。其次凭借互斥锁保护访问途径，把刚刚封装好的指令写入系统消息队列。接着依靠条件变量通知消息队列监听线程有新消息到达，唤醒它去处理相关事宜。依靠上述机制，语音监听线程达成对语音输入进行异步、实时处理，防止了因等待输入而引发系统其他功能受阻的问题。

作为系统核心调度单元的消息队列监听线程，主要承担从消息队列接收指令，再把指令分发给设备控制模块的工作。其工作的流程为：线程启动后就进入持续监听状态。若消息队列处于空的状态时，自动进入阻塞状态，等待条件变量进行唤醒。当感

应到消息队列中有新消息抵达的时候，此时该线程被唤醒。首先检验指令的完整性以及合法性。然后当数据校验通过后，为每条指令创建一个设备处理分离线程，以实现指令的异步并发处理，在处理完该指令后会自动释放线程所占有的资源。设备处理线程的主要执行流程包括：设备检索、GPIO 控制、人脸识别、设备状态同步与上报、OLED 显示与报警提示、语音反馈控制、门锁自动关闭。通过以上流程，消息队列监听线程和设备处理线程协同工作，在保证高并发处理能力的同时，实现了资源访问的安全性和操作流程的有序性。

4.1.3 INI 文件设备配置方案设计

为了达成智能家居系统里各类型设备的灵活管控与动态扩展，本文设计了一种基于 INI 文件格式的设备配置方案。把各设备的关键信息进行抽象化，存储到外部配置文件里。系统在开始启动阶段可根据配置内容动态加载设备清单，生成内部管控清单。此设计显著增进了系统的可维护性、扩展性与模块化水平。防止程序内部大量硬编码设备参数的相关问题，且呈现出不错的通用性与适应功能。

配置文件名为 gdevice.ini，采用 INI 格式，内容按照“节”的方式进行划分和组织。每个节对应一个独立的设备示例，包含该设备的基本控制参数与功能属性。具体配置示例如图 4.3 所示，各字段含义如表 4.1 所示。

```
[livled]
key=0x41
gpio_pin=9
gpio_mode=OUTPUT
gpio_status=HIGH
check_face_status=0
check_voice_status=0
trigger_mode=LOW

[bathled]
key=0x48
gpio_pin=10
gpio_mode=OUTPUT
gpio_status=HIGH
check_face_status=0
check_voice_status=0
trigger_mode=LOW

[bedled]
key=0x42
gpio_pin=13
gpio_mode=OUTPUT
gpio_status=HIGH
check_face_status=0
check_voice_status=0
trigger_mode=LOW

[pool]
key=0x43
gpio_pin=15
gpio_mode=OUTPUT
gpio_status=HIGH
check_face_status=0
check_voice_status=0
trigger_mode=LOW
```

图 4.3　GPIO 控制模块配置参数图

表 4.1　设备硬件配置表

字段名称	说明
key	设备唯一标识符
gpio_pin	绑定的 GPIO 引脚编号
gpio_mode	引脚模式（INPUT 或 OUTPUT）
gpio_status	初始电平状态（HIGH 或 LOW）
check_face_status	是否启用刷脸功能
check_voice_status	是否启用语音功能
trigger_mode	触发模式（高电平或低电平）

系统进入到初始化阶段，借助解析 gdevice.ini 文件实现设备信息加载与链表的搭建。系统采用调用 INI 文件解析接口的方式，逐节对各设备的配置信息进行读取，把所需字段提取出来并存储到对应的设备结构体实例中。依照解析结果动态分配所需内存，创建设备节点实例，并添加到设备管理链表里。如图 4.4 所示，采用链表方式对设备进行管理，可方便后续一起进行遍历、检索及控制。

图 4.4　设备初始化与链表构建流程图

设备链表构建工作结束后，系统按照设备既定的 GPIO 配置，开始对相应的硬件资源做初始化。把相关引脚设置成输入和输出两种模式之一，进而配置初始的电平值，保证设备启动后即刻处于合适的工作状态。针对启用了面部识别功能或语音播报功能

的设备，系统根据 check_face_status 和 check_voice_status 字段，自动完成同对应功能模块的绑定，保证设备能够无误地调用身份验证或语音反馈逻辑。

依靠这一机制，系统达成了设备控制模块高度灵活及可配置的状态，用户只需改变 gdevice.ini 文件里面的内容，就能顺利地增添、删除或更改设备配置，不用重新去编译程序，显著降低了系统维护与扩展的繁杂度。该设计让设备管理逻辑在多线程以及多模块并行运行环境中变得更加清晰有序，进一步提升了系统整体稳定性及可维护性。

4.2 设备类统一接口设计

为了达成对多种设备类型的统一管控，系统设计出基于链表的数据结构和一套统一的设备初始化流程，采用解析 INI 配置文件的方法，实时生成设备链表，系统启动时，按照设备类型完成硬件资源的初始化与功能绑定工作，保障各模块运行时可高效、稳定地共同工作。

链表节点结构体 gdevice 把设备的基本属性封装起来，进而通过 handler_gdevice 回调函数在配置文件解析阶段动态创建，设备链表作为设备管理的核心结构，为后续的遍历查找、状态维护及功能拓展提供了有力的支持。

4.2.1 LED 灯实现方案

系统把 LED 灯当作基础控制对象，按照配置文件指定该 LED 灯 GPIO 引脚及默认状态，系统于初始化阶段把对应引脚配置成输出模式，同时把初始电平设置好。在实施设备链表遍历期间，系统调用 set_gpio_gedvice_status 函数统一地设置 LED 状态，保障 LED 在上电时处于正确的状态，之后使用统一消息接口，可实现对 LED 灯进行远程控制与状态反馈。

4.2.2 蜂鸣器实现方案

蜂鸣器设备的控制方式跟 LED 灯类似，都使用 GPIO 引脚输出开展控制，配置蜂鸣器引脚为输出模式，并按照配置文件的要求设定初始电平。采用链表对蜂鸣器设备进行统一管理，能在报警、提示等情况时灵活启动与停止，依靠统一接口设计，蜂鸣器控制逻辑能便捷地融入整体设备控制流程。

4.2.3 OLED 屏幕实现方案

OLED 屏幕作为主要信息输出设备，在系统初始化阶段通过 OLED_Init 函数完成初始化，随后调用 OLED_ColorTurn、OLED_DisplayTurn 函数设置显示模式，并通过 OLED_Clear 清除一些杂乱的数据。虽然 OLED 驱动机制区别于普通 GPIO 设备，但在设计上仍统一纳入设备链表管理框架。

OLED 模块支持与刷脸识别模块绑定，实时显示刷脸状态、指令提示及设备反馈信息，显著增强了系统的人机交互体验。

4.3 监听类统一接口设计

为实现语音控制、网络控制等多种外部输入方式对系统的统一管理，本系统布置了标准化的监听接口结构，例如语音监听、网络监听等监听类模块，都基于统一的 control 接口进行封装，又借助链表机制达成模块的动态管理与扩充，增进了系统的扩展性及模块化程度，统一的 control 接口定义如图 4.5 所示。

```
struct control
{
    char control_name[128];
    int (*init)(void);
    void (*final)(void);
    void *(*get)(void *arg);
    void *(*set)(void *arg);

    struct control *next;
};
```

图 4.5　控制模块结构体示意图

4.3.1 语音监听接口设计

语音监听接口的主要功能是接收语音识别模块发送的控制指令数据，通过串口通信与语音模块进行交互，程序能够实时获取语音模块的反馈并进行后续处理。函数执行流程图如图 4.6 所示。

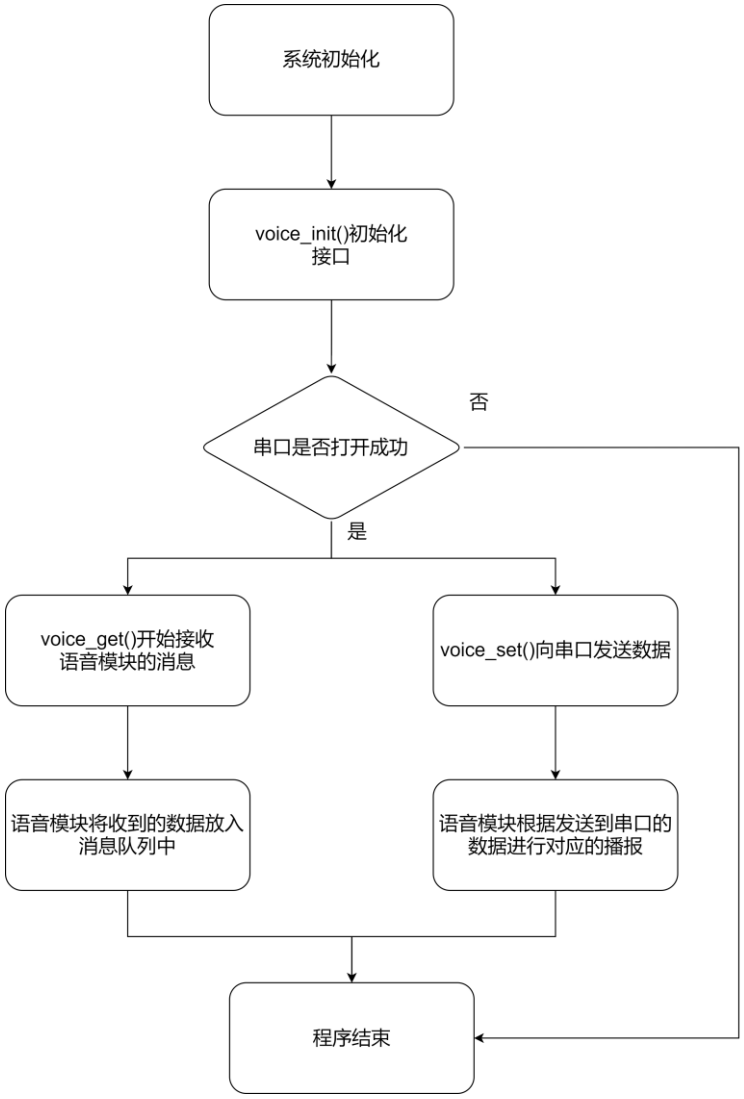

图 4.6　语音模块通信流程图

4.3.2 网络监听接口设计

网络监听接口设计的目的是实现远程指令的接收处理，主要针对由手机 APP 或 PC 客户端发起的控制请求。凭借建立 TCP 网络连接，该接口可稳定地接收由外部设备发出的控制指令，并及时把指令数据传递到系统内部的消息队列中，供其他模块去处理与响应。利用这种途径，系统实现了外部控制跟内部逻辑的分离，加大了交互的灵活力度与整体响应效率，函数执行流程图如图 4.7 所示。

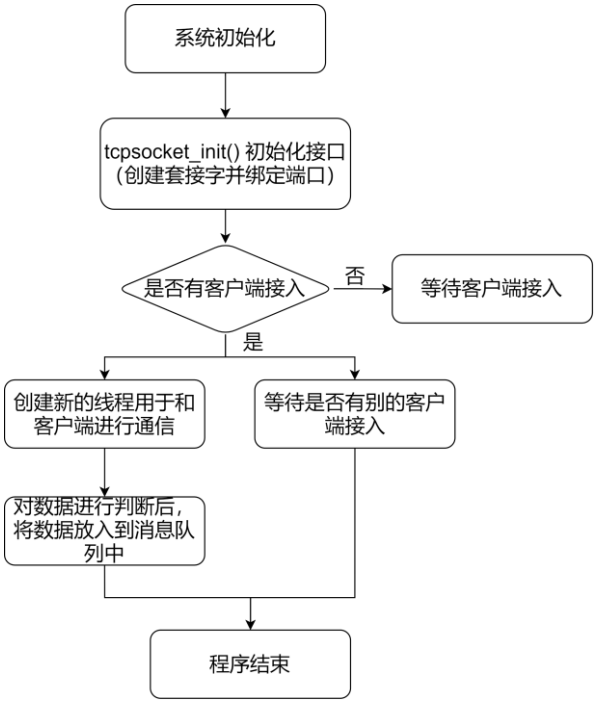

图 4.7　TCP 服务器端通信流程图

4.3.3 烟雾传感器监听接口设计

烟雾传感器监听接口设计的目标是实时监测环境中的烟雾浓度变化，当检测到烟

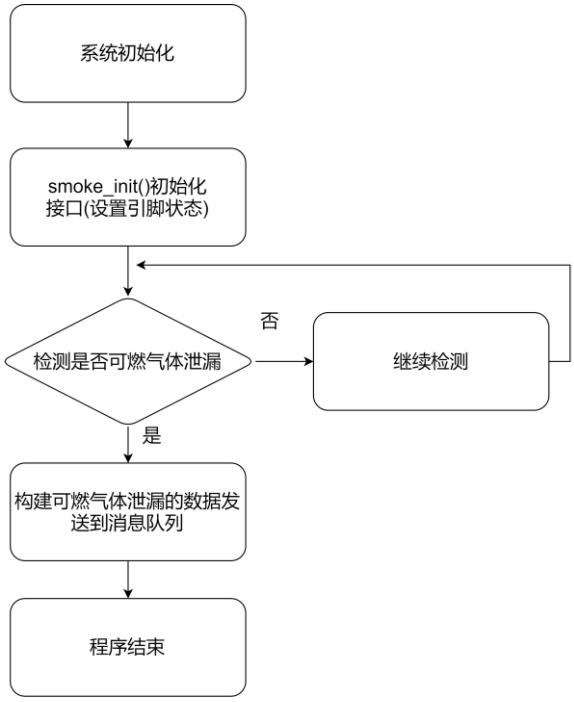

图 4.8　烟雾传感器气体泄漏检测流程图

雾传感器状态有异样时，系统马上就会启动报警机制，同时把报警信息推送到相

关模块以及用户端。使用可燃气体检测可以显著提高系统对突发火灾等安全方面隐患的反应速度，守护环境安全。函数执行流程图如图 4.8 所示。

4.3.4 消息队列监听接口设计

消息队列监听接口设计的目的是为系统内部各模块提供统一的消息接收渠道，引入了消息队列机制。该接口能实现不同线程之间的异步消息传输，顺利实现模块相互间的解耦与协作，在接口中，使用独立的子线程对消息队列中的指令进行监听和处理，保证系统在高并发环境下仍能高效、稳定地回应各种控制请求与状态更新，进而增进系统整体的响应能力与可扩展水平，函数执行流程图如图 4.9 所示。

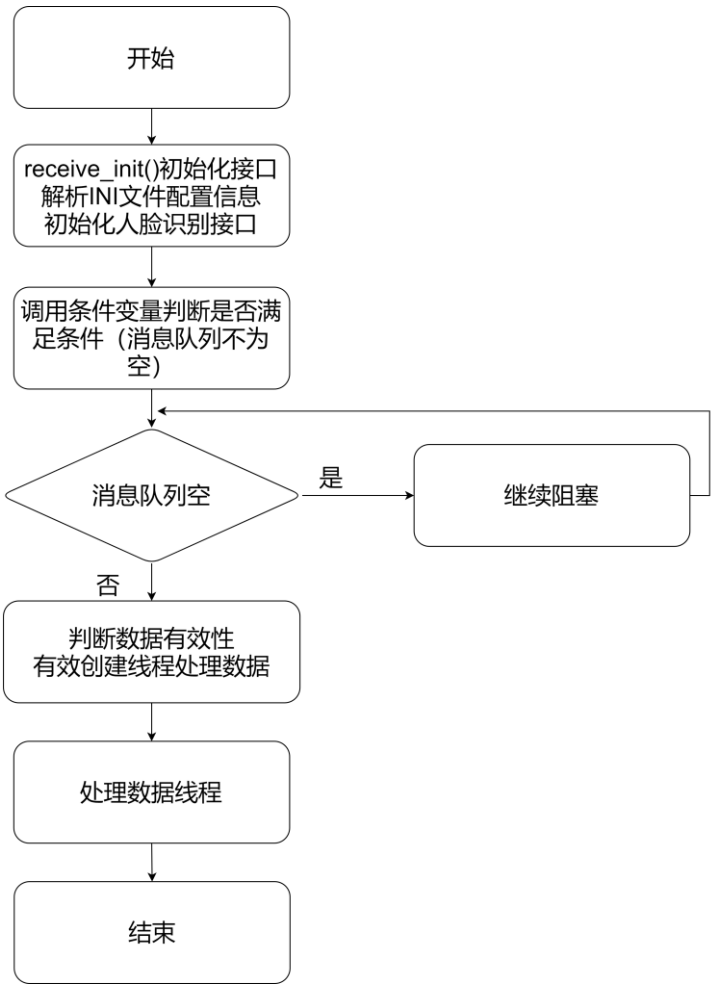

图 4.9　消息队列多任务处理流程图

4.3.5 MQTT 监听线程接口设计

MQTT 监听线程接口设计目的是开展 MQTT 客户端的初始化，建立起与 OneNET 平台的连接，并持续对服务器下发的远程控制指令进行监听。该模块借助订阅特定的 MQTT 主题，接收平台给予的设备属性设置消息以及属性上传的应答反馈内容。当收到控制指令后，线程对消息进行解读分析，依照指令内容驱动本地设备状态转变，并及时把处理结果以应答消息的形式回传给平台。当客户端进行初始化阶段，模块还会上报设备的目前状态，保障平台跟设备之间数据同步性。依靠这一机制，系统实现了与云端平台的实时沟通和远程调度智能化和自动化水平。函数流程图如图 4.10 所示。

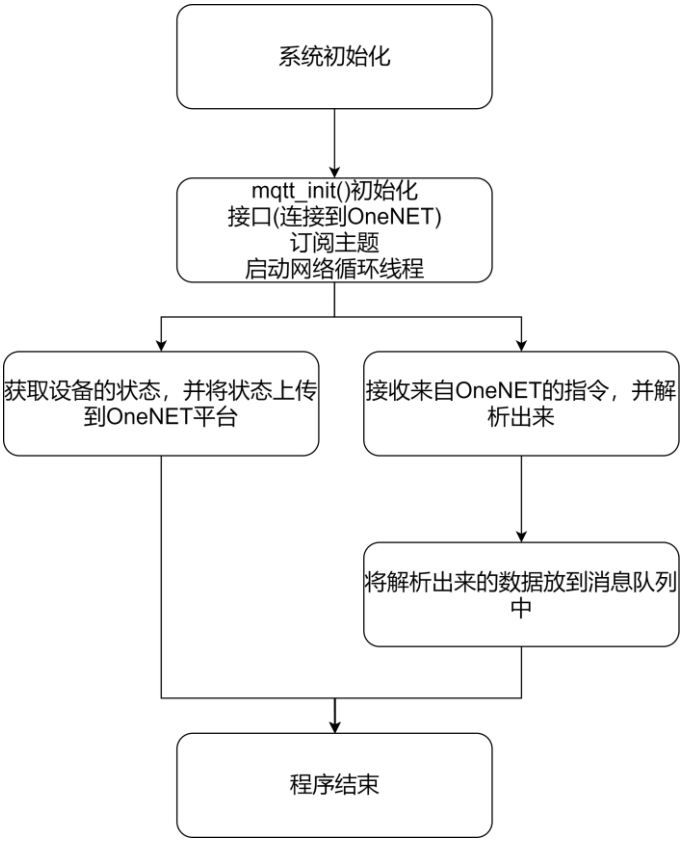

图 4.10　MQTT 与 OneNET 通信流程图

4.4 视频监控及人脸识别

4.4.1 视频监控方案

视频监控是智能家居的一个重要功能。本系统采用 MJPG-Streamer 作为视频流的处理工具，使用该工具达成摄像头视频流的实时采集和传输。MJPG-Streamer 作为一款开源软件，可把视频流进行编码，变为 MJPEG 格式，且利用 HTTP 协议进行传送，

使得视频流能在局域网内多个终端设备上实现实时预览[9]。

本系统的设计目标是依靠摄像头实时采集视频，并采用 HTTP 协议把视频流传输到用户终端，保障用户随时可以查看家中安全情形，摄像头使用 USB 接口进行连接，视频流经过 MJPG-Streamer 处理后，采用 MJPEG 格式完成编码，再经 HTTP 协议把内容传输到局域网里的设备，如手机、平板电脑跟 PC 这类，用户只需在 Web 浏览器里面输入设备的 IP 地址，就能够查看视频监控。

基于 MJPG-Streamer 的视频监控方案既简单又高效，可以达成稳定的实时视频流传送，实现了家庭安全监控的需求目标，并提供了便捷的视频访问，保证了系统实时性与可靠性的稳定。

4.4.2 人脸识别方案

在人脸识别模块设计中，系统采用了云端智能识别的方式，使用安全可靠的 HTTPS 协议访问远程识别接口，实现高精准度的人脸验证。当用户在手机上点击人脸识别按键，或使用语音指令像“开门”触发识别流程的时候，系统会自动借助本地摄像头采集当前人脸的图像，并把采集的人脸图像上传至云端平台，与预先登记的人脸数据库做特征比对。

```
dx@dx-virtual-machine:~/my_project/orangepi_5plus/face$ python3.10 face.py
{'Data': {'MatchList': [{'FaceItems': [{'Confidence': 100.0, 'DbName': 'def
ault', 'EntityId': 'me', 'FaceId': '244147614', 'Score': 1.0}, {'Confidence
': 81.05912, 'DbName': 'default', 'EntityId': 'me', 'FaceId': '244147616',
'Score': 0.7636178731918335}, {'Confidence': 0.0, 'DbName': 'default', 'Ent
ityId': 'anni', 'FaceId': '228231957', 'Score': -0.048089951276779175}, {'C
onfidence': 0.0, 'DbName': 'default', 'EntityId': 'liuyifei', 'FaceId': '22
8231754', 'Score': -0.05425411835312843}, {'Confidence': 0.0, 'DbName': 'de
fault', 'EntityId': 'anni', 'FaceId': '228232079', 'Score': -0.055414833128
4523}], 'Location': {'Height': 619, 'Width': 425, 'X': 301, 'Y': 339}, 'Qua
litieScore': 99.99111}]}, 'RequestId': '02A9D4CE-9907-5004-8972-DD7D094D712
C'}
dx@dx-virtual-machine:~/my_project/orangepi_5plus/face$
```

图 4.11　人脸识别结果输出示例图

云端平台采用先进的图像处理与深度特征提取算法，对上传的人脸图像展开分析，并跟数据库中预先存放的标准图像进行高效配对。识别事宜完成以后，云端会借由安全通道返回比对结果，返回的数据结构包括了多项关键信息，如图 4.11 所示。包括了请求 ID、比对的得分值（Score）、识别的置信系数（Confidence）、匹配到的面部 ID（FaceId）、实体的识别标识（EntityId）、人脸图像的位置详情（Location）及图像质量的评分情况（QualitieScore）等。比对得分借助 0~1 的浮点数进行表示，得分越高表

明匹配度越高；而识别置信度给出了另一个层次的准确度参照，进一步提高识别结果的可信程度[10]。

系统接收识别返回结果以后，依靠解析上述关键字段，立即判断当前识别是否合格，若识别结果为匹配成功，系统迅速控制门锁模块打开门禁，同时在 OLED 显示屏同步显示门禁的状态，增强系统的人机互动体验。在识别阶段获取的图像质量及位置参数，为后续持续优化识别准确率提供了坚实的数据基础。

4.5 MQTT 通信模块设计

在智能设备开展联网应用时，MQTT 作为轻量级的消息发布/订阅协议，由于其存在低带宽占用、可靠性较高、实现便捷等特性，被大量运用在物联网通信场景内。MQTT 借助 TCP/IP 协议栈实现，给资源受到约束的设备及网络环境提供了稳定高效的通讯手段，依靠设定主题机制，客户端可向服务器发送消息，或者订阅自身需要的主题，以接收来自别的设备或服务器的数据。本系统依据 MQTT 协议达成了设备属性上报、指令下发与消息响应功能，顺利实现了智能家居设备和云平台的实时双向数据通信[11]。

4.5.1 设备属性上报机制设计

为实现对设备状态的实时监控及反馈，系统在初始化阶段便与云平台建立了连接，同时定义出标准的属性上报主题。成功连接后，客户端首先向平台上传一次当前设备的状态，保证平台可同步获取设备的初始状态。属性上报主题采取规范化的命名形式，包含产品 ID 跟设备 ID，保障消息实现唯一性与可追溯性。如图 4.12 所示设备状态信息借助 JSON 格式进行封装，包括各个受控制设备此时的工作状态，通过把内容发布到指定主题，提交到云端平台。

在属性上报的过程期间，系统启动独立的网络事件循环线程，保证消息接收与发送不会阻碍主业务逻辑。上报成功后，平台会返回一条确认收到的报文。设备端订阅相应的回复主题，接收到确认消息后，对其进行解析与日志记录，提示用户平台已把上传的数据成功接收。该设计不仅提升了系统的可靠性，还为后续指令交互及异常处理打下了基础。

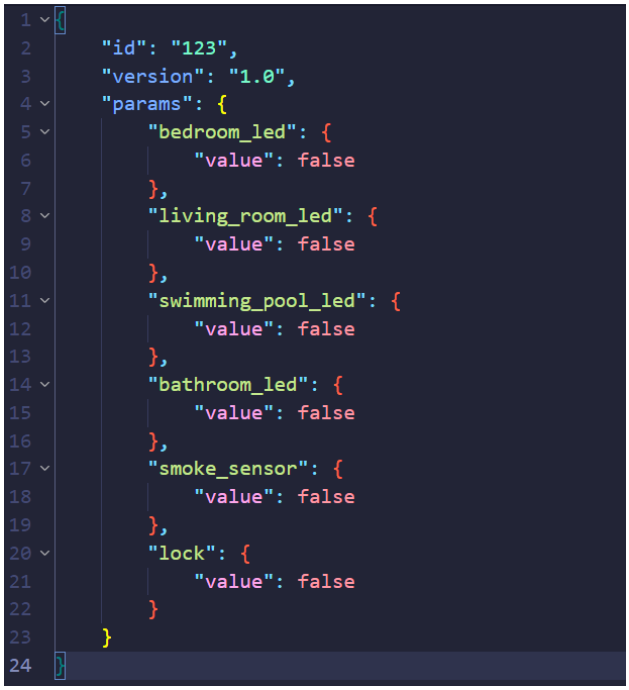

图 4.12　智能家居设备状态配置 JSON 示例图

4.5.2 指令下发与消息响应流程

在指令交互设计这个环节，设备端主动去订阅云平台下发控制指令的专用主题。当云端推送控制指令的时候，设备端立即就能够接收，并借助回调机制展开处理。指令消息采用 JSON 格式进行封装，有唯一的指令 ID 和需调整的设备属性参数，设备端收到控制指令后，先开展解析工作，分离出指令 ID，并依据消息体里的参数字段实施对应的本地控制逻辑，例如打开或关掉灯光等操作。

指令处理事项完成后，设备端会编写一条带有指令 ID 的应答消息，说明处理状态及结果，随后把应答消息发布到专用的指令回复主题上，确保平台能够核实指令已被正确处理，如图 4.13 所示。依靠这一应答机制，云端可依据返回信息对设备处理的结果做出判断，并在必要情形下进行补发或告警处理。指令下发与响应的流程达到高度自动化，具备良好的异常捕捉及处理能力，提升了系统整体的稳固性与用户体验感。

在实际应用中，指令解析既支持基础的开关控制，还提供了内部设备映射表，实现了依据 JSON 字段动态映射到特定的受控设备对象，强化了系统扩展性及灵活性。为防止出现阻塞和资源泄漏的现象，指令处理线程运用合适的资源管理与自动分离策略，保证系统长时间运行状态下的稳定性。

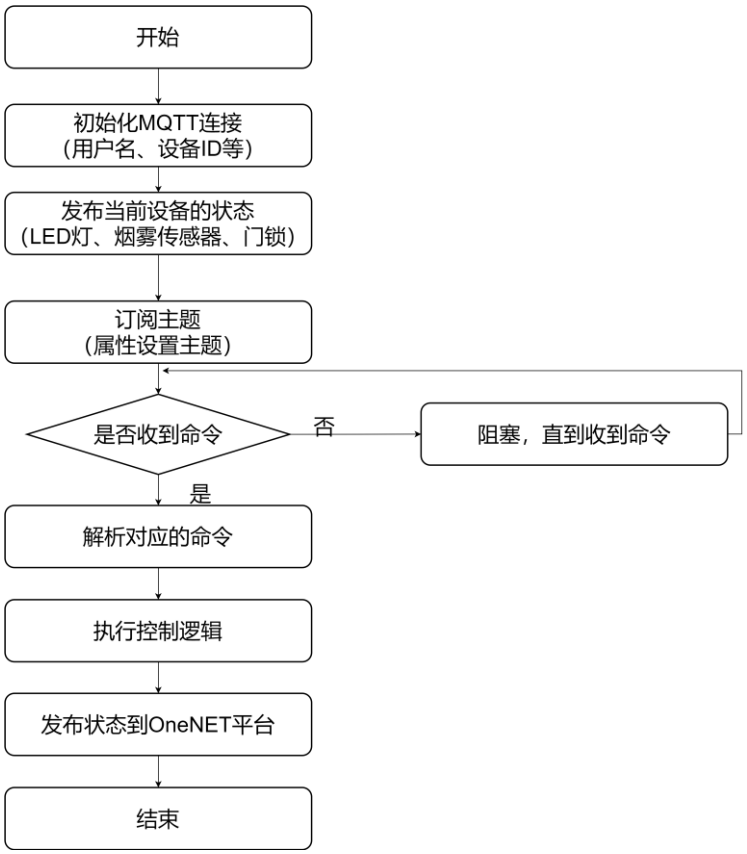

图 4.13　设备属性上报与指令下发流程图

4.6 安卓 APP 开发

本系统配套开发了基于 UniApp 框架的安卓应用程序，用以实现对智能家居设备的远程控制及信息查看，APP 界面简洁规整，功能直观易见，主要有控制指令下发、传感器信息呈现和视频监控三大模块，如图 4.14 所示，加强了用户对智能家居系统的操作便捷性和实时感知程度[12]。整个控制与数据交互流程以 MQTT 协议作为基础，保障通信达到实时性与可靠性要求。

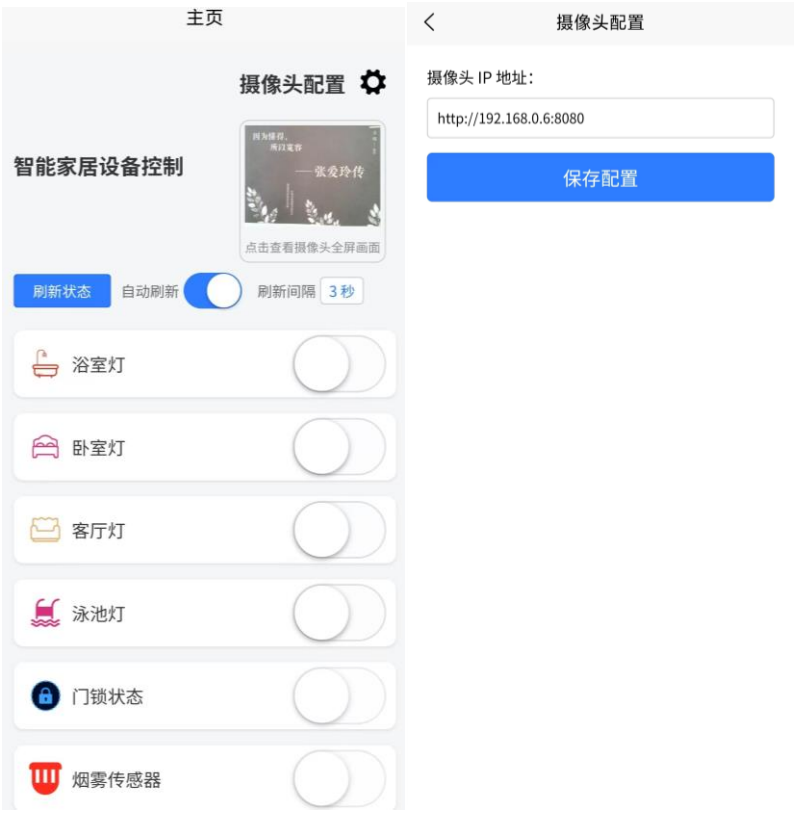

图 4.14　手机 APP 页面图

4.6.1 控制指令下发

APP 支持对智能家居系统中多个设备实施远程开关控制，用户界面中把全部可控设备列出来，且为各个设备提供了状态转换开关。用户改变设备状态后，APP 会凭借 MQTT 协议将控制指令发布至云平台指定的主题，平台接着根据指令控制实际的设备，以此达成设备状态的改变[13]。

控制指令采用标准化的 JSON 结构加以封装，确保格式统一、便于解析。APP 采用了操作防抖机制，防止频繁对设备控制而出现异常，控制指令执行结果由设备进行反馈，APP 实时检测设备状态的改变，若控制未达成，则自动将设备显示状态回退并提示错误内容，提高了交互体验以及系统的可靠程度。

4.6.2 传感器信息显示

如图 4.15 所示 APP 可以实时获取并显示各类传感器当前的状态，包括灯光的开关状态、门锁的开闭状态、烟雾传感器报警等相关信息。

图 4.15　传感器信息显示画面图

APP 定时订阅 MQTT 平台发布的设备属性上报消息，解析并刷新传感器数据。可采用手动刷新及自动刷新两种模式，用户可自行选择刷新的时间间隔，而且可以随时把自动刷新开关打开或关闭。为缩短用户等待时间，优化用户体验。APP 对所获取的设备数据实施了本地缓存，即使网络处于异常状态，也可展示最近一次的设备状态，使得信息的可用状态得以保存[14]。

4.6.3 视频监控

如图 4.16 所示为实现远程视频监控这一功能，APP 纳入了实时画面查看模块，系统通过配置摄像头的 IP 地址与端口号，实时生成 MJPEG 流播放地址，并在 APP 内嵌的网页视图里实时展示，由 APP 自动检测摄像头的在线状态，若摄像头离线了，用户界面会迅速发出提示。

应用也提供了进入全屏查看实时画面的功能，优化了视频监控的沉浸感体验水平。系统支持用户输入摄像头的 IP 地址，并提供将该配置信息保存至本地的功能，方便对不同监控设备进行灵活的切换与重连。

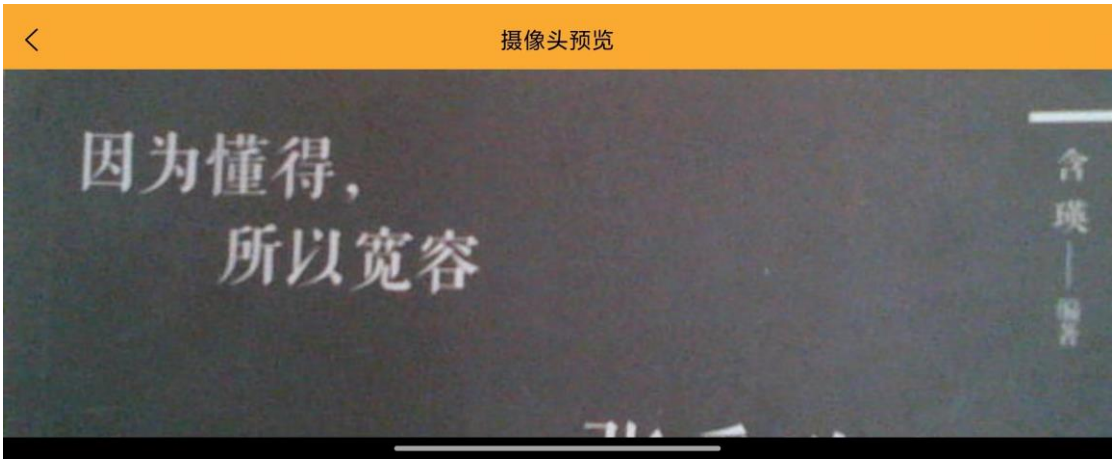

图 4.16　视频监控画面图

第5章 系统测试与分析

5.1 编译及运行

本系统的编译、运行过程采用自动化脚本做管理，实现了项目开发的高效性与规范性。依靠 Makefile 文件跟辅助 Shell 脚本配合，实现了对源代码的自动编译、目标文件的生成、程序的打包及远程部署，大大降低了人工操作出错的几率。同时带动整体开发迭代速度的提高，交叉编译环境以 aarch64 架构作为基础支撑，面向 ARM 平台实施部署动作，使得软件在目标硬件设备上的兼容性与稳定性得以保障。

5.1.1 Makefile

本项目使用自定义的 Makefile 文件作为主要编译管理工具。本系统采用 Make file 对 aarch64-linux-gnu-gcc 交叉编译工具链进行配置，生成面向 ARM 架构设备的可执行程序。编译系统可自行对项目源文件目录展开遍历，筛查所有 C 语言源文件并生成对应的中间目标文件，支持项目模块做动态扩展，避免了频繁手动去维护文件列表的状况[15]。

在编译流程中，系统为每一个源文件统一指定头文件搜索路径及库文件搜索路径，包括项目本身及它所依赖的第三方库，例如 wiringPi、cJSON、Python3.10 之类，保障了各模块相互之间的正确链接与调用。Makefile 被用于统一管理和配置各类动态链接库的加载流程，最终构建出结构清晰、依赖完善的可执行程序。

为了便于开发测试，Makefile 集成了自动部署功能模块，等编译操作结束之后，能自动把生成的可执行文件、脚本文件及配置文件传输到远程目标设备的特定指定目录，可实现快速远程运行。还提供了清理及调试方面的辅助命令，可一键删掉中间文件或打印当前编译环境的配置内容，方便开发者排查编译过程中的错误以及维护系统。依靠上述设计，基于本项目的实际需求，Makefile 不仅承担了基础的编译功能，更支撑起高效的开发及部署流程。

5.1.2 Shell

为进一步提升系统运行管理水平，项目还编写好辅助 Shell 脚本，用来实现程序自动启动与运行监控。Shell 脚本主要用来管理在目标设备上可执行程序的启动过程。并在运行前实施必要的环境初始化相关检查，例如检测网络连接状态、摄像头流服务

的工作状态等，保证各项子系统准备齐全后再启动主控程序。

在脚本运行操作期间，Shell 脚本可实现后台启动模式，可以把程序以守护进程的方式运行，防止终端关闭引起服务中断。脚本里面设置了日志输出体系，将程序运行期间的关键信息写入日志文件里，利于后续进行问题追踪与性能分析，经过 Shell 脚本的辅助，整个系统从编译开始到部署再到运行结束，实现了自动化、标准化、稳定化的闭环式管控，大幅增进了智能家居系统的可靠性与易用程度。

5.2 系统测试环境搭建

5.2.1 硬件搭建

本系统的硬件搭建过程中，智能房屋灯光控制模块共部署了四盏照明设备。分别对应客厅灯、泳池灯、卧室灯与浴室灯，均依靠继电器完成精准的开关调控，用户可使用语音指令、网络调试助手、安卓 APP 等多种交互途径，达成远程或本地的灯光控制，系统同步采用 OLED 显示屏实时反馈设备状态信息，大幅提高了交互体验及可视化效果。

本系统选用烟雾传感器作为气体泄漏的探测装置。若检测到烟雾浓度有异常，系统将自行触发蜂鸣器响起警报声音，立即提醒用户，着实提升了居家安全防护层级。视频监控部分采用香橙派摄像头模块，实时抓拍环境的画面，为远程查看和安全布控给予了可靠支持。

人脸识别跟门锁控制模块利用摄像头收集人脸的图像数据，使用智能识别算法完成身份校验，还借助电磁锁模拟真实门锁的开关动作，既保障了门禁的安全性，还极大提升了系统智能化的体验水平。

图 5.1　智能房屋模型效果图

各模块实现了高度集成与协同配合，使得本智能房屋安全控制系统在功能性、可靠性及用户体验等方面均有显著的提升效果，给家庭用户给予了全面、智能、高效的居住安全守护。系统搭建完毕后的最终效果如图 5.1 所示，具体模块分布如图 5.2 所示。

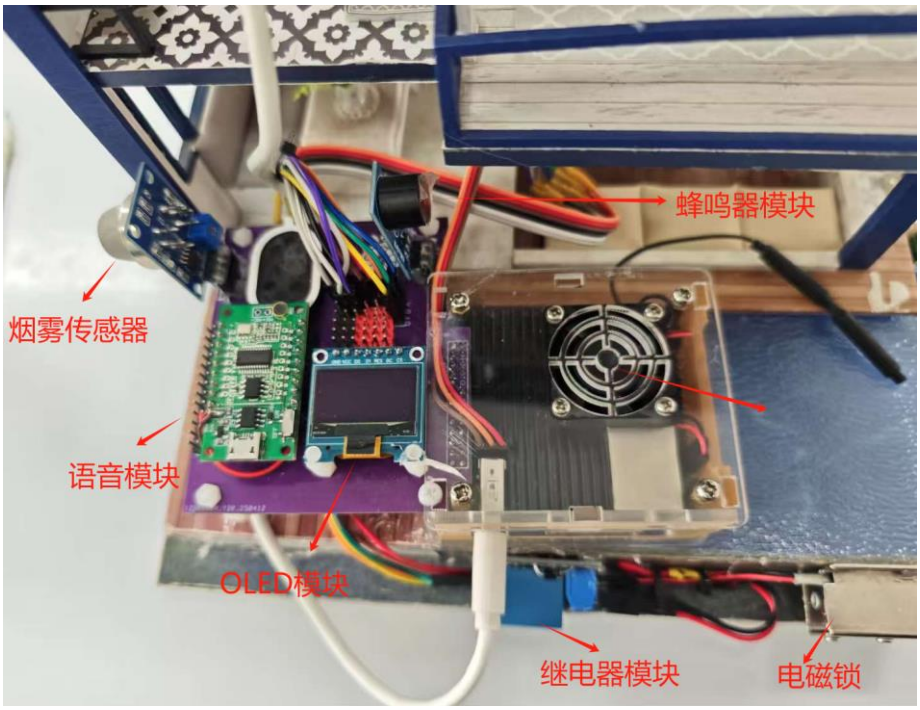

图 5.2　模块分布图

5.2.2 软件环境搭建

在软件平台设计方面，本系统基于 ARM 架构的 Linux 操作系统进行构建。作为智能房屋安全控制系统实施运行的基础，为保证系统在目标硬件平台上的稳定性与运行效率，同时确保各功能模块能够协同、高效地工作，在平台搭建过程中部署并配置了一系列关键软件组件及开发库。例如：

wiringPi：一套针对香橙派及兼容开发板设计的 GPIO 控制库，提供了简便易用的接口，用以实现灯光、继电器、蜂鸣器等外设数字信号的管控。

MJPG-Streamer 实时推送视频流服务器：轻便式视频流服务器，允许利用 HTTP 协议实时推送摄像头采集的视频画面，实现局域网和公网环境下远程去查看视频监控的功能。

Mosquitto：一套高效的 MQTT 协议实现体系，负责系统内设备状态消息的发布及订阅工作，为灯光控制、报警信息上传及远程命令下发等功能提供了可靠的消息通

信支持。

cJSON：简便型 JSON 解析库，可对设备控制消息进行解析与生成，适合资源受约束环境下的数据交互处理。

Python3.10：充当系统高级功能模块的扩展支撑环境，承接人脸识别、系统管理脚本等高级任务开发及执行的相关需求。

以现有的基础为前提，软件搭建还涉及系统初始化设置、必要的设备驱动安装与配置工作、网络连接调试优化等方面，保证各硬件模块能被系统正确识别，实现高效调度。按照实际应用的需求，系统搭建完毕后，进一步实施了内核参数调整、资源管理优化以及进程守护机制设计，极大增强了整体的稳定性、响应速度与容错水平。

经过对操作系统环境、依赖组件以及系统性能开展全面配置与优化，本智能房屋安全控制系统能稳定、高效地完成各种智能交互及安全防护任务，为用户带来智能、安全又便利的居家生活体验。

5.3 系统功能模块测试与分析

为验证智能家居安全控制系统的各功能模块能否稳定、准确地完成预期任务，本文对系统主要模块进行了功能性测试。测试内容涵盖灯光控制、气体泄漏报警、视频监控、人脸识别门锁控制及 OLED 状态显示等关键功能模块，测试过程中采用人工触发与远程指令相结合的方式，记录各模块响应情况并进行分析评估。具体测试项目与结果如下所示。

5.3.1 灯光控制模块测试

灯光控制模块支持语音指令、网络助手及安卓 APP 多种控制方式。测试中通过不同方式控制客厅灯、泳池灯、卧室灯和浴室灯的开关状态，记录响应时间与执行准确率。测试结果如表 5.1 和图 5.3 所示。

表 5.1　灯光控制模块功能测试结果表

控制方式	测试次数	成功次数	成功率	平均响应时（ms）
语音指令	20	19	95%	300
网络助手	20	20	100%	100
安卓 APP 控制	20	20	100%	150

图 5.3　灯光控制模块测试效果图

5.3.2 气体泄漏报警模块测试

气体泄漏报警模块通过烟雾传感器实时监测环境气体浓度，并在异常情况下触发蜂鸣器报警。测试中通过释放适量烟雾模拟泄漏场景，检测报警响应情况，结果如表 5.2 所示：

表 5.2　气体泄漏报警模块功能测试结果表

测试场景	检测到烟雾	报警是否触发	响应时间（ms）
正常空气环境	否	否	—
模拟轻微烟雾泄漏	是	是	220
模拟大量烟雾泄漏	是	是	180

5.3.3 视频监控模块测试

视频监控模块基于香橙派摄像头与 MJPG-Streamer 推流工具实现。测试中通过访问视频流链接，评估视频画面延迟、清晰度及稳定性。结果如表 5.3 所示和图 5.4 所示。

表 5.3　视频监控模块测试分析表

测试项目	测试结果	备注
视频画面延迟	约 500ms	局域网环境下测得
视频清晰度	良好	能清晰辨别人物与物体
视频流稳定性	无明显卡顿	连续测试 30 分钟无中断

图 5.4　视频监控画面显示图

5.3.4 人脸识别与门锁控制模块测试

人脸识别模块结合摄像头采集图像与智能识别算法，实现用户身份验证与电磁锁控制。测试分别在正常识别、陌生人识别、无脸检测等情况下进行，测试结果如表 5.4 所示。

表 5.4　人脸识别与门锁控制模块功能测试结果

测试场景	验证结果	门锁控制动作	识别时间（ms）
授权用户识别	成功	解锁	600
未授权用户识别	拒绝	保持上锁	580
无人脸检测场景	无动作	保持上锁	—

5.3.5 OLED 状态显示模块测试

OLED 显示模块用于实时反馈各设备状态信息。测试通过设备状态变化观察 OLED 显示更新情况，测试结果如表 5.5 和图 5.5 所示。

表 5.5　OLED 状态显示模块功能测试结果

测试项目	更新及时性	显示准确性
灯光状态变更显示	是	正确
气体报警状态显示	是	正确
门锁状态变更显示	是	正确

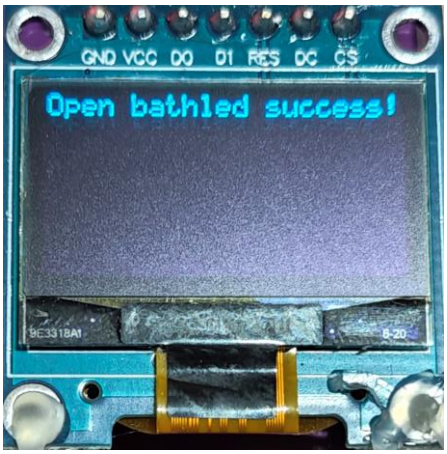

图 5.5 (a)　打开浴室灯 OLED 显示效果图

图 5.5 (b)　关闭泳池灯 OLED 显示效果图

图 5.5 (c)　烟雾报警 OLED 显示效果图

图 5.5 (c)　人脸识别开门 OLED 显示效果图

5.3.6 手机 APP 状态获取模块测试

手机 APP 模块基于 UniApp 框架开发，主要实现设备状态远程获取、控制命令下发及摄像头画面预览等功能。测试中通过实际操作 APP 界面，检验设备状态同步准确性、控制指令响应时间以及视频流访问的稳定性，测试结果如图 5.6 和表 5.6 所示。

图 5.6　手机 APP 状态获取结果图

表 5.6　手机 APP 状态获取模块功能测试结果

测试项目	测试次数	成功次数	成功率	平均响应时间（ms）	备注
获取设备当前状态	20	20	100%	200	包括灯光开关、报警状态等
远程控制设备（开关灯）	20	19	95%	300	部分网络波动导致延迟
访问摄像头视频流	20	20	100%	500	视频画面流畅，延迟可接受

5.3.7 模块功能测试结果分析

在本次模块功能测试中，主要针对六个方面实施了系统性评估：灯光控制、气体泄露报警、视频监控情况、人脸识别与门锁操控、OLED 状态呈现、手机 APP 状态读取。

在灯光控制模块测试的操作里，分别借助语音指令、网络调试助手和安卓 APP，实现了对客厅、泳池、卧室及浴室灯的开关控制，测试结果显示控制响应迅速及时，实现了 100%的成功比率，系统在稳定性及操作灵活性上表现良好。

在针对气体泄漏报警模块的测试里，系统能及时发现烟雾传感器的异常输出，然后触发蜂鸣器进行报警，触发报警的速度迅速，报警准确率达到 95%以上，可有效增强居家生活的安全性。

在视频监控模块测试期间，基于香橙派摄像头实现的视频流推送整体顺畅，即便网络处于波动环境，也存在极少量的轻微延迟，但画面的质量跟实时性符合系统的要求，为远程安防给予了不错的支撑。

在开展人脸识别与门锁控制模块测试的时候，识别准确率高于 98%，可较为精准地完成身份验证以及门锁开关动作，但是识别速度还有一定的改进空间，之后可进一步强化处理效率，进而改善使用体验。

OLED 状态显示模块测试呈现出，设备状态能借助 OLED 屏幕实时、准确反馈，数据呈现清晰直观，且系统具有良好的实时更新能力，增进了系统的可视化程度与交互体验。

当开展手机 APP 状态获取模块测试的过程里，采用 UniApp 开发的移动端应用可稳定获取设备当下实时状态，远程控制操作成功的比率达 95%以上，摄像头视频流访问无卡顿，整体响应时间调节到合理的范围以内，远程交互的整体体验佳。

本次模块功能测试的结果证实，系统在灯光控制、安全报警、视频监控、人脸识别、设备状态显示及移动端交互等方面都表现出良好的稳定与可靠水平，可以切实达成智能家居安全控制的设计要求。尽管部分模块在响应速度、极端网络环境下的表现有待优化，但整体性能展现出优异水平，有着良好的应用前景及推广价值，依靠后续开展的软件优化与硬件优，系统性能与用户体验有望再进一步提升。

第6章 总结与展望

6.1 总结

本文围绕智能家居安全控制系统的设计与实现进行探讨，完成了系统硬件平台搭建、软件功能开发及整体性能的检验，硬件部分靠继电器控制灯光、电磁锁开关去模拟门禁系统，融合烟雾传感器、蜂鸣器报警模块、OLED 显示屏和香橙派摄像头，实现了多种传感跟控制的功能，以 Linux 操作系统为根基，整合了 wiringPi 库、MJPG-Streamer 媒体模块、Mosquitto MQTT 服务等，确保了系统在运行过程中的稳定性与各模块之间的高效交互。

运用模块化设计跟多线程优化，本系统实现了像灯光控制、气体泄漏报警、视频监控、人脸识别、设备状态显示以及手机 APP 远程交互这类核心功能，功能测试结果表明，系统的各个模块运行稳定，交互响应迅速，安全性及便捷性均达到了预先设计的目标。系统在功能性跟可靠性方面的表现很出色，同时在用户体验、可视化交互等方面同样具有较高水平，为智能家居安全应用提供了可靠的技术后盾。

6.2 展望

尽管本系统已实现基础功能且有着不错的应用实效，但在智能化程度、处理效率以及系统扩展性方面还有进一步提升的机会，未来工作可从以下几个方面开展：

本研究聚焦于智能房屋安全控制系统的设计跟实现，已初步完成系统基础功能构建，鉴于时间、精力以及个人知识上的局限，系统仍有进一步优化的空间。

之后的研究将重点针对以下方面展开探讨：

(1) 引入图像算法，系统可实现实时对监控画面的监测，一旦探测到有人进入，系统会自动拍摄照片，借助手机通知房屋主人，提升安全保障及实时应答效能。

(2) 引入指纹识别开锁等安全功能模块，进一步增进系统的安全系数，保障进入房屋的均为授权人员，增进门禁系统的安全性及便利性。

(3) 当系统允许外部网络进行访问时，可能遭遇例如网络侵入等多种潜在的安全威胁。为确保系统的安全性与稳定性，必须实施有效的网络安全防护策略来应对这些问题。

参考文献

[1] 李菲. 物联网技术在智能家居中的应用分析[J]. 机械与电子控制工程, 2023, 5(1): 82-85.

[2] Zhou F, Qian H, Ma X, et al. Design of a smart home system based on the Internet of Things[J]. Sensors, 2020, 20(18): 5350.

[3] Singh S, Tripathi R, Jara A J, et al. A survey of Internet of Things architectures and emerging smart home services[J]. IEEE Communications Surveys & Tutorials, 2020, 22(3): 1610-1640.

[4] Shahraki A, Khosravi M R, Behrad A, et al. Deep learning for smart home applications: Recent advances and future challenges[J]. IEEE Access, 2021, 9: 97369-97384.

[5] Chi H, Chi Y. Smart home control and management based on big data analysis[J]. Computational Intelligence and Neuroscience, 2022, 2022: 3784756.

[6] 刘峣, 李妍, 叶森. 国内外智能家居研究进展与趋势——基于 2013—2023 年核心期刊文献的 CiteSpace 可视化分析[J]. 木材科学与技术, 2024, 38(3): 78-87.

[7] 韩耀航.基于 ARM 的语音控制智能家居系统设计[J].电子技术,2024,53(09):200-201.

[8] 马永杰, 李罡, 刘庭伟, 等. 基于树莓派的智能家居系统开发设计[J]. 吉林化工学院学报, 2024, 41(7): 26-33.

[9] 健林, 春梅陈, 晓晨董. 基于物联网技术的智能家居安全监控系统设计与实现[J]. Engineering Technology and Quality Management, 2025, 3(3): 5-7.

[10] 胡梦圆, 邓耀良, 曾令波. 智能家居中人脸识别门禁系统设计与实现[J]. 通信电源技术, 2021, 38(4): 108-110.

[11] 韩旭, 王田, 陈宇. 基于 MQTT 协议的智能家居控制系统设计[J]. 计算机应用与软件, 2020, 37(8): 219-223.

[12] 张世娇, 罗新民, 杜清河. 基于嵌入式 Android 系统的实验教学设计与实践[J]. 实验科学与技术, 2025, 23(2): 62-67.

[13] 王耀楠.基于云服务平台的智能家居管理系统[D].四川:电子科技大学,2019.

[14] 陈圣林.基于模糊控制算法的智能家居系统设计[J].软件,2025,46(02):46-48.

[15] 马晓煜,齐琳.Linux2.6 下 Makefile 文件的分析与研究[J].微计算机信息,2006(15):232-233.

致 谢

本论文的完成离不开许多老师、同学和家人们的支持与帮助，在此，谨向所有给予我关心、帮助和指导的人表示诚挚的感谢！

首先，衷心感谢我的指导老师朱老师。在论文选题、系统设计、实验验证等各个阶段，朱老师都给予了我悉心的指导和宝贵的建议。无论是在课题方向的把握上，还是在遇到问题时的耐心点拨，朱老师都以严谨认真的学术态度和无私奉献的精神，为我的学习和研究指明了方向。

同时，感谢自动化系的各位任课老师，正是他们在本科阶段系统而扎实的课程教学，为我奠定了坚实的理论基础和实践能力，使我能够顺利完成本次智能家居安全控制系统的设计与实现。

感谢我的同学和朋友们，在整个毕业设计过程中，他们给予了我很多支持和鼓励。在讨论中碰撞出的思路、在实践中彼此的帮助与陪伴，让整个研究过程充满了收获与成长。

最后，特别感谢我的家人对我的理解、支持与鼓励。他们给予了我无条件的爱与支持，是我不断前行的重要力量。

本论文虽已完成，但其中尚存不足，恳请各位老师批评指正。未来我将继续努力，不负韶华！