**封闭环境无人自动巡检系统**

**（CEAGVIS）**

**学校：** 广东工业大学

**学院：** 机电工程学院

**指导老师：** 于兆勤

**参赛队员：** 陈永铎 吴显 刘洁宇 黄泊凯

**电邮:** 3347620766@qq.com

**日期:** 2024年 7 月 31 日

# **摘要**

本项目的目标是建立一个**封闭环境无人自动巡检系统**，并通过包括物联网技术在内一系列方案，实现仅需数名人员就能够低成本高效率对相对封闭的空间区域进行监测管理的效果，具体应用场景包括：仓库、办公室、实验室、工地等等。本项目能**快速部署**于各类各型仓库等封闭环境中，在无需改变室内的布局的情况下同时实现对制定空间区域的高效管控监测。

由于篇幅限制，本报告书只对该项目中的重点核心模块进行解释

**关键词：物联网系统、AGV、边缘计算、消防安全**

# **项目背景**

在当前物流需求快速增长的背景下，仓库建设面积自2018年的2.54亿平方米增长至2023年的4.04亿平方米。然而，大多数仓库建设都发生在工业4.0和人工智能技术普及之前，导致这些设施大多缺乏现代化配套设施，如全面的监控覆盖和高质量的监控设备。为解决这些挑战，我们提出了一个创新的解决方案——名为CEAGVIS的封闭环境无人自动巡检系统。

CEAGVIS整合了物联网技术、机械结构设计、电路控制、路径规划与避障算法、云计算以及人工智能技术。这一系统的主要优势在于实现了仓储管理的全面自动化和无人化，通过引入AGV和工业机械臂等设备，完全替代了传统的人工巡检和装卸作业，显著提升了工作效率和操作安全性。

不仅如此，CEAGVIS还具备极高的可操作性和适应性。它不仅适用于各类仓储环境，还可以轻松应用于办公室、封闭体育场馆等类似场所。这种多功能性使得CEAGVIS具备了广泛的应用潜力和良好的可扩展性，为企业节省大量成本和时间，同时避免了仓库改造过程中可能带来的业务中断和不必要的费用支出。

总之，CEAGVIS不仅是一个技术上的突破，更是一个能够显著提升生产效率、降低运营成本并改善工作环境的重要创新方案。

# **一、方案介绍**

## **1.1 整体：**

整个项目由5个模块组成，分别是：与用户交互的网页（Web）和移动（Mobile）前端、负责数据指令处理储存并作为用户端和硬件端通信中枢的后台数据库（本地服务器）、无人自动导引车（AGV）、监控网络、充电台。

该系统整体运作流程如下：

1. 由技术团队根据实际情况安装部署硬件设备，包括服务器（也可以是直接利用现有服务器进行部署）。

2. 按照客户预设方案进行巡检监测，方案可由技术团队按需求直接存储在服务器本地数据库中或者由用户通过不同前端来确定。

3. 当AGV的车载传感器或者服务器后台处理程序检测到异常数据时，立即按照相关紧急预案进行处置，并通过不同前端向用户发出警告。处置完毕后，系统复位并按照“2”中步骤继续执行。

## **1.2 组成模块：**

### **a） 无人车(**AGV**)**

项目硬件主要载体,由HT32F52367作系统主控芯片，接收外部模块的信号，并由主控处理数据后判断进行决策，下达控制指令，控制无人车的自动巡检与电量检测和充电。并辅以ESP8266作为物联网模块连接物物联网平台，将事件情况反馈到负责人终端，在矩阵式红外测温模块监测到有火源后由K210（后期会迭代升级）判断火源位置并将灭火炮口对准火源，启动舵机旋紧螺丝柱按压灭火器开关进行灭火，同时响起蜂鸣器。

为了方便适用多种场合，无人车搭载的灭火器可以更换成市面上装有不同灭火材料的灭火器。

后期迭代，将使用搭载有华为海思Hi3519DV500芯片（ARM架构SoC）边缘智能工作站IPV09A或者华为昇腾Atlas 200I DK开发板替换K210,另外会引入英特尔L515激光摄像头，以实现更好的识别效果和更多的功能。

### **b） 充电台**

在无人车电量较低时无人车会自动行驶到此充电，并上传硬件传感器监测数据。

### **c） 监控网络**

由无线视频摄像头、无线红外摄像头（甚至烟雾传感器等等）组成，并于服务器后台连接，将监测数据实时上传。

### **d） 本地服务器（后台数据库）**

搭载有后台处理程序和数据库的工作站（规模大小视实际情况而定，可以就是一台个人小型计算机，也可以是大型计算集群）。这将被作为与用户交互的前端和硬件端的通信中枢，使用户能够及时查看现场包括硬件的实际状况，并能将用户的指令传送至硬件端。

### **e） 前端（ Web / Mobile ）**

通过网站或者手机等移动设备上的应用，使用户能够随时随地非常方便的操作系统，包括查看数据，下达指令等。

# **二、拟解决的关键问题**

## **2.1 边缘模型部署问题**

在我们所提出的项目部署运行过程中，对仓库的实时监测是重中之重，这就需要系统能够及时的对各传感器收集上传的数据进行分析处理。正如当下主流技术路线的做法，我们将引入AI模型到我们的系统中来处理这些数据并输出结果。经过分析，我们认为需要将模型部署在硬件端，这样硬件端无需传输巨量的信息到软件端的后台，避免了对网络（有线或者无线）的过度依赖，也降低了网络异常会导致系统故障甚至崩溃的可能。同时，将模型部署在硬件端使得边缘模型能够第一时间接受传感器的监测数据并处理。因此，将模型部署至硬件端，需要解决几个关键点：1、边缘硬件算力支持；2、模型剪枝优化；3、数据I/O及后处理。

## **2.2 无人AGV路径规划和避障问题**

在所部署的室内空间，如果未使用路径规划或者路径规划不当，小车可能会在狭窄的通道中频繁堵塞或因为无序躲避某些场景的障碍物而陷入死循环，反复巡逻单一区域，导致巡逻路径无法遍布整个仓库，另外，电量的过多消耗容易让小车充电次数过多，电池老化，降低巡逻效率。在激光雷达，矩阵温度芯片，视觉模块等外部传感器的加持下使用ROS系统综合优化的路径规划能让小车准确地到达目的地，从而提高巡逻效率。其次，有助于降低运营成本。通过精确的路径规划，物流小车能够更有效地利用能源，减少燃料或电力的消耗。规划出最短且最顺畅的路径，能延长电池的续航里程，降低充电次数和成本。而当火情发生时能够快速响应，迅速到达指定位置，高效灭火，减少火势蔓延的时间，提高灭火效率。

## **2.3 AGV机械结构设计问题**

本次比赛中，机械结构设计方面存在的难题是如何平衡质量、结构强度与体积之间的关系以及如何稳定触发灭火器开关。

手里拿着电子设备

中度可信度描述已自动生成首先，由应力分析可知小车应力主要集中于底盘与电机支座之间的连接处，由于小车需要搭载有一定质量的灭火器，并且各工件都是3D打印而成，材料强度有限，此处十分容易断裂，因此材料的选择和板材厚度的设计十分有考究。经过多次仿真模拟实验决定采用强度较ABS更高的PC材料打印支座（ABS的硬度约为70 Shore A，而PC的硬度约为80 Shore D），厚度选择3mm，从而成功解决这一问题。

图为电机支座断裂情况

其次，对于触发灭火器开关的问题，我们有着以下考量：由于灭火器有一定高度且触发时需要较大的力，因而需要考虑较大距离力的传输衰减这一难题。经过多种方案的筛选，我们选择采用电机带动绞盘卷绳拉动开关的方式。此时我们又面临着绳材的选择问题，如果选用的绳子弹性较大会导致绳受力拉长时，使电机空转，然而选用的绳子弹性差又容易导致绳子直接受力过大而崩断。我们通过多次实验，最终选择采用弹性适宜的钓鱼线来解决了这一问题。图片包含 桌子, 食物, 啤酒, 男人

描述已自动生成

图为绞盘绞断绳材情况

## **2.4 后台数据更新维护问题**

从各种传感器和设备传来的数据在经过预处理后，数据量仍然非常大并且在持续增加，但在需要实时处理时，系统应该如何快速响应并及时处理，并且在这个过程中能够保持数据的准确性和一致性，是数据处理的一大关键。毕竟，大量来自各种传感器和设备的数据对实时分析系统要求较高，需要系统能够灵活处理不同格式的数据，也就是拥有高效率和高准确性，而失去这种效果则会让收集处理数据带来极大的困难，也难以保证其安全性。对此，我们也需要实现技术的快速更新来提高系统能力，以实现高效，准确，安全的实时数据处理。

## **2.5 后台……**

## **2.6 前端交互 / 用户操作 问题**

# **三、工作原理**

## **3.1 边缘模型模块（卷积图像识别）：**

### **3.1.1 模块总述**

该模块功能实现目前基于K210。通过查询资料和分析，该模块被要求在无人车搭载的视觉识别开发版上运行，并要与AGV系统板进行交互通讯，属于典型的端侧模型部署运行。在该环境下，硬件性能极其有限（诸如算力、运行内存），算力等资源调配捉襟见肘，因此需要在软件方面尽可能进行优化并且裁剪模型大小。经过对比测试，选用Ultralytics团队开发的YOLO算法（You Only Look Once）。YOLO算法的核心思想是使用单个神经网络直接预测图像中所有物体的类别和位置，而不需要像传统目标检测算法那样进行区域提案。该算法模型中的第五代（YOLO-v5），提供了预训练模型和简单的训练接口，使得我们团队能够方便地在自己的数据集上进行训练。YOLO-v5是一种流行的目标检测算法模型，它在速度和准确性方面取得了良好的平衡，特别适合实时应用场景。YOLO-v5的基本原理是将目标检测任务视为回归问题，通过卷积神经网络直接预测目标的位置和类别。

### **3.1.2 算法特点**

YOLO-v5当中的关键底层原理有：自适应锚框计算、自适应图片缩放、数据增强方式、网络结构、创新损失函数。关于自适应锚框计算方面，在自适应锚框计算方面，YOLOv5 采用了一种称为 k-means clustering 的方法来生成最佳的锚框尺寸，并在训练迭代过程中使用 IoU 替代传统欧几里德距离，能够更好的反映锚框与真实框之间的匹配程度。这种方法可以帮助模型更好地匹配不同大小的目标物体，并通过简化锚框配置，减少了模型训练和调整的复杂性，使得用户更容易使用和调整模型。关于自适应图片缩放方面，YOLOv5中的自适应图片缩放（Adaptive Image Scaling）是一种基于目标尺度的图像缩放方式，它可以自适应地缩放输入图像的尺寸，以适应不同尺度目标的检测。这种方法可以有效地解决目标检测中存在的尺度不一致问题，提高检测精度和鲁棒性。关于数据增强方式方面，为了提高模型的泛化能力和鲁棒性，YOLOv5在输入端采用了Mosaic数据增强方式。Mosaic是一种基于四张图片的数据增强方法，通过随机缩放、裁剪、排布和拼接四张图片，生成一张新的训练样本。这种数据增强方式不仅丰富了数据集，还增加了小目标的数量，使得模型在训练过程中能够学习到更多的目标特征，提高检测精度。关于网络结构方面，YOLO-v5的网络结构主要包括Backbone、Neck和Head三个部分。Backbone负责特征提取，Neck用于多尺度特征融合，Head负责预测。YOLO-v5采用了CSP-Darknet53作为Backbone，该网络在保持较高精度的同时，通过改进网络结构，提高了模型的推理速度。关于创新损失函数方面，YOLO-v5采用的是一种名为CIOU-LOSS的损失函数，它是IOU-Loss的改进版，通过计算预测框与GT框之间的最小外接矩形面积，再计算出矩形面积中不属于两个框的区域占矩形面积的比重，然后用IOU减去这个比重得到CIOU。这种损失函数可以更好地处理目标检测中的边界框回归问题。同时，通过转换模型文件格式，该模型将能够支持在多种平台和设备的部署，因此可以实现在嵌入式开发板环境下的端侧模型部署运行。

### **3.1.3 底层原理**

具体到原理层面上，需要结合人工智能领域中机器学习与神经网络方面的知识来说明。此次项目中的使用的YOLO算法模型属于典型的卷积神经网络模型（Convolutional Neural Networks – CNN），此类神经网络的网络结构一般由五个部分组成：输入层，该层通常处理的是图像数据，可以是单通道（灰度图像）或多通道（彩色图像）；卷积层，该层通过多个卷积核对输入数据进行卷积运算，提取出不同的局部特征。每个卷积核对输入数据的一个小区域（称为感受野）进行加权求和，得到特征图；激活函数层，在卷积层之后，通常会加入一个激活函数层，对卷积层的输出进行非线性变换，以YOLO-v5为例，当中使用了Mish函数、SiLU（Swish）函数、LeakyReLU函数、Sigmoid函数和Softmax函数，这些函数分布在模型的不同结构组件当中；池化层，池化层通常位于卷积层之后，用于降低特征图的空间维度，减少计算量，同时保留重要的空间信息。池化操作一般有两种，即最大池化和平均池化；全连接层，该层分布在卷积层和池化层之后，全连接层将提取的特征映射到样本标记空间，实现分类或回归等任务。具体到YOLO-v5卷积视觉识别模型当中，通过对上述五种网络基本结构不同设计接合，组成相应的基本算子组件：CONV、CSP、SPP、Focus、CPP、CBL等。这些算子组件再与处理算法组成YOLO-v5模型的核心模块：输入端（Input）、骨干网络（BackBone）、颈部网络（Neck）、头部网络（Head）。在模型训练过程中：首先进行前向传播处理，即输入数据经过卷积层、激活函数层、池化层等层层处理，最终得到网络的输出。这个输出与真实标签进行比较，计算出损失函数的值；随后根据该结果，再进行反向传播处理，根据损失函数的梯度信息，从输出层逐层向输入层反向传播误差信号。通过链式法则计算出每一层参数的梯度值；最后进行参数更新，即根据反向传播计算出的梯度，使用梯度下降算法更新网络中的权重和偏置参数；再将上述步骤不断重复，也就是不断训练迭代，使模型识别精度尽可能提高。最终将训练好的模型使用相关工具进行模型转换，部署到开发板上。

### **3.1.4 部署推理**

另外在实现部署层面上，我们首先在电脑本地Ubuntu22.04（Linux）上搭建开发环境来进行训练迭代和模型推理，这样便于同其他硬件进行适配和通信交互，比如AGV的运动路径规划是采用ROS（Robot Operation Systym）来实现，而ROS就是基于Linux环境来实现的，包括其他的单片机和开发板（包括 Raspberry-Pi / HUAWEI-Atlas-200I / IPV09A / ……），大多都是搭载Linux系统来操作运行。并且，在本地训练的时，考虑采用Pytorch框架（.pt）来进行开发，

当获得一个相对理想的模型文件和参数权重时，将其转换为ONNX格式（.onnx）,在根据实际情况来适配部署。例如：目前使用的K210开发板用相应的平台工具 ，将模型文件转换为.kmodel后缀格式文件，并配合相应固件库即可实现部署；

而华为海思IPV09A（Hi3519DV500）边缘计算工作站则需要将文件转换为离线模型文件格式（.om），并在工作站上安装相应编译链和依赖文件才能实现部署运行。

## **3.2 着火点识别与扑灭MS26M833(AMG8833)**

由倍创生产设计的BMS26M833矩阵式红外测温模块，采用AMG8833传感器开发而成。该模块是一款基于先进MEMS技术的高精度红外数组传感器，可直接输出8×8 (64pixels)温度矩阵。其侦测视角为60度，侦测距离可达到7米。此模块可通过BMCOM接口，使用I2C通信方式，实现读取红外温度数据等功能。在本项目中，该模块可以配合视觉识别多重判定是否为火源，提高判定火源的精确度。

该模块在使用过程中对普通物体的发热不敏感，测试温度需要补偿。但是在调试过程中，我们发现，该模块起到了核心作用。该芯片的MEMS技术对于火焰异常敏感，使用IIC协议读取8\*8温度矩阵再扫描其中的高温成分即可精准识别到火焰。

火焰温度识别算法

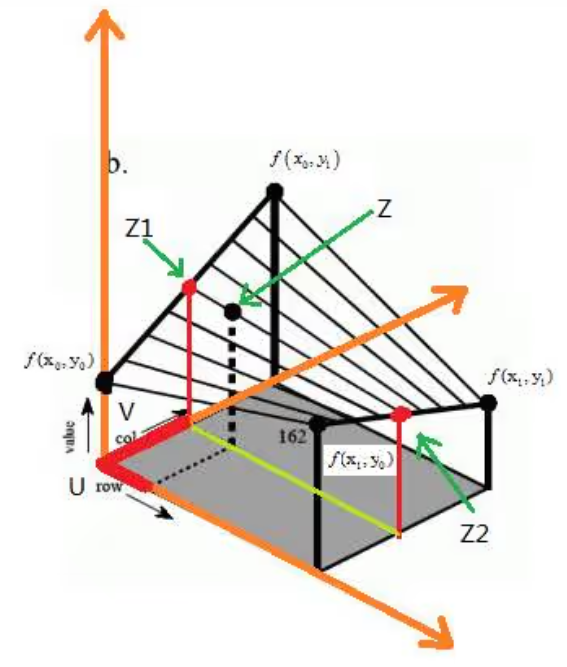
 当矩阵中监测到监测视野内有温度过高的区块，结合视觉识别所捕获到的大致区域，对其进行识别是否为火源。然而，该模块的分辨率较低，所以我们采用图像双线性插值处理，放大图像分辨率，用以提高识别准确度

图1：双线性插值示意图

双线性插值是一种图像放大算法，在图示中，所求目标点有四个相邻点，根据x轴上的对所求目标点的x坐标Xt对f(x0,y0)和f(x0,y1)两个相邻点的距离的图像温度值进行加权平均求得第一个点的温度值Z1，这种方法是单次线性插值。同理，对另外两个相邻点进行线性插值求得温度值Z2，最后根据两次线性插值所得的点与在y轴上对所求目标点的y坐标的距离的图像温度值再进行一次线性插值，可求得目标点的温度值。

我们使用这种方法，根据调试所得的最佳倍数，对8\*8温度矩阵进行放大。

主控接收到对应矩阵坐标的温度值和坐标并与视觉识别芯片K210进行拟合判断是否为火源，若呈现图像上和温度上的一致性，即触发灭火流程。该方法也同样适用于人形识别。

## **3.3 运动控制原理**

无人车运动控制

无人车采用履带式驱动以应对复杂环境，选择两个MG513霍尔编码器减速电机作为动力源，TB6612模块作为电机驱动，QMC5883三轴磁场传感器模块精准反馈无人车的偏航角。

### **3.3.1 速度控制**

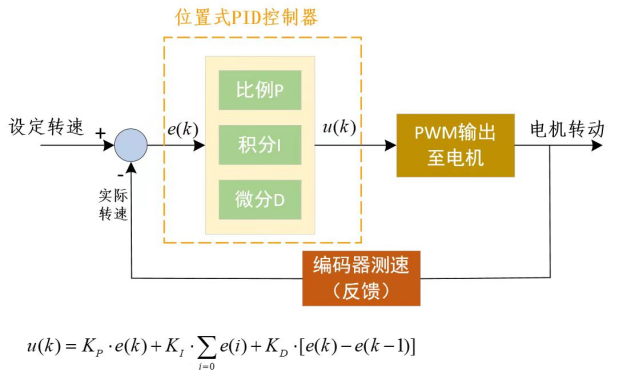


图3：PID算法示意图2

单片机输出pwm的占空比控制电机转速，占空比越大电机转速越快，采用pid算法使速度控制更为平滑，响应速度更快，通过单位时间内霍尔编码器产生的脉冲数，利用PID对其运行速度进行精确调控。

### **3.3.2 无人车航向控制**

无人车的转向采用履带差速实现，左右两个履带上电机的速度一致，转动方向相反即可实现无人车原地转向。

使用QMC5883三轴磁场传感器模块获取无人车实时偏航角数据，采用pid控制使无人车维持在某一角度方向上的直线行驶，或者原地转向某一目标角度。

### **3.3.3 无人车向可能着火点导航的功能实现**

#### **3.3.3.1 安保台识别着火点位置**

在安保台云台摄像头识别到画面中有可能的着火点后，两轴云台上的舵机控制摄像头转动，将摄像头中心点对准画面中的着火点，并回传舵机此时的角度数据，并且通过云台上的超声波模块测得着火点的距离信息，通过角度和距离信息计算得到着火点相对于安保台的水平方位，如图方位用极坐标表示：

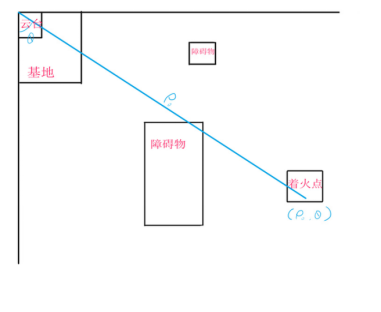


图2：PID算法示意图1

#### **3.3.3.2** 无人车避障并前往目标位置

该部分分为两种控制模式，即车载摄像头无法看见着火点和车载摄像头可以看见着火点两种情况，由于无人车高度较小，视野容易被障碍物遮挡，一开始无人车采用第一种不可见的控制模式，到靠近到可见着火点后使用第二种模式，并消除第一种模式带来的误差。

（1）车载摄像头无法看见着火点，在该情况下无人车目标航线始终为一开始云台所反馈的目标航线，当车载激光雷达识别到前方障碍物时采取避障动作。

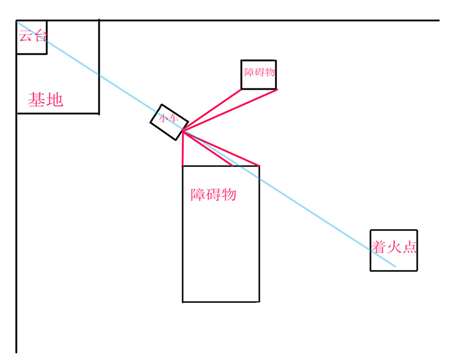


图4：PID算法示意图3

如图所示，选取激光雷达在正前方100度范围内探测到的五十个平均点作为避障数据，获取障碍物的相对位置。

之后无人车开始避障动作，即原地转向90度，朝垂直于目标航线的方向前进，前进的距离为无人车相对于目标航线的偏移量，由编码器测速然后再对速度积分得到该偏移量。

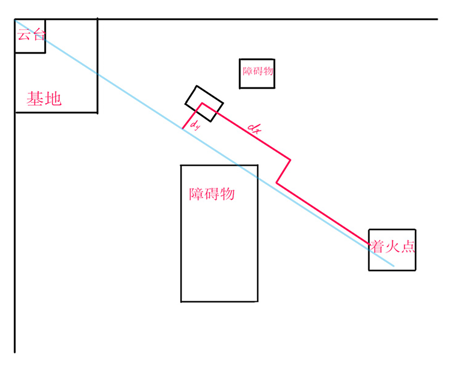


图5：PID算法示意图4

（无人车的运动都为原地转向加直线运动，方便进行编码器测速积分出位移）

如图所示，无人车避障动作后再原地转向，进入一条与原目标航线平行，距离为偏移量的航线，沿该航线前进到越过障碍物的位置，越过后再次原地转向朝着垂直于原航线的方向前进，前进距离为之前所测得的偏移量，通过该动作回到原目标航线上，但由于偏移量存在误差，当前所处航线实际上只是与原航线平行，而这种误差会随着避障次数的增加而变大。为了避免该情况，我们引入了第二种控制逻辑，即车载摄像头能看见着火点

（2）车载摄像头可见着火点，如图，在完成第一种逻辑的一部分避障动作后，着火点出现在车载摄像头视野内，此时控制无人车原地转向，使得着火点对准摄像头中心，此时车头朝向即为新的目标航线。通过在避障动作后对目标航线进行更新可以消除（1）中的误差，并且得到更短的路径。

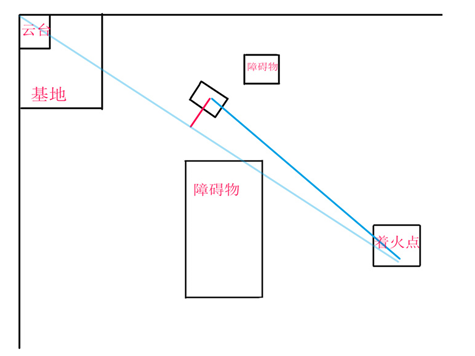


图6：PID算法示意图5

## **3.4 后台数据库及处理程序（本地服务器）**

### **3.4.1 与硬件端的通信交互**

这套物联网系统的核心在于能够收集并有效处理大量来自各种传感器和设备的数据。首先，物联网设备，比如温湿度传感器、光照强度探测器等，持续不断地监测其周围的环境变化，并通过无线通信技术（如Wi-Fi / Zigbee / LoRaWAN /……）或者有线通信技术（比方说在充电台充电的时候）将这些数据发送出去。在某些场景下，这些数据可能会首先被发送到一个网关设备，在那里进行初步的数据处理或过滤，即数据清洗，格式转换等预处理，然后通过互联网将数据转发给后端服务器。这一过程中，确保数据的安全至关重要，这种情况下通常采用SSL/TLS加密来保护数据在传输过程中的安全。或者硬件端可能会直接与后台进行通信交互。这二者具体采用哪一种方式视实际情况而定。在本地服务器收发硬件端的数据信息的这一过程中。通常是通过实现 RESTful API 来完成的，这些 API 允许外部设备通过 HTTP 请求发送数据到服务器。服务器端接收到的数据随后会被存储在数据库中，以便于后续的处理和分析。MySQL是这类应用中常用的数据库选择，因为它提供了可靠的数据存储能力和丰富的功能集，适合存储结构化的数据，也同时是因为它是开源数据库，便于获取，而且相关社区配套资源丰富全面。为了提高数据检索的速度，通常会对数据库中的关键字段建立索引。除了数据上传，后端服务器也可以向硬件端发送指令，控制设备的操作（如调整温度，控制开关等），实现双向的通信交互。

### **3.4.2 本地处理**

而在服务器（后台数据库）数据本地处理储存阶段。需要先进行预处理，对收集的数据进行清洗，去除无效或重复数据，整合来自不同数据后。再根据应用的需求，将数据处理分为实时处理和批量处理两种类型。实时处理通常涉及到对数据进行即时分析并作出响应，比如触发警报或调整设备设置。这可以通过使用流处理框架如 Apache Kafka 和 Apache Flink 来实现。Flink的状态后端还提供持久性和恢复功能，可以避免数据丢失。而直接或者结合使用这些框架，可以提高实时数据流的处理效率。另一方面，批量处理则更侧重于定期对数据进行汇总分析，如统计分析、数据挖掘等，这可以通过使用 Hadoop MapReduce 或 Apache Spark 等工具来完成。这两个工具都支持数据的并行处理，通过多个节点来提高处理速度。在处理数据的同时，还需要实现相应的业务逻辑，以满足具体的应用需求，对异常情况及时反应。这可能包括但不限于阈值检测、异常报警、数据分析等功能。将数据进行压缩和分层存储，按照访问频率的不同存储在不同的空间里，也可以确保数据质量，节约成本。最后也要建立监控系统，实时监控数据处理和存储的性能，以便及时发现并解决潜在问题。

### **3.4.3 与前端的通信交互**

为了辅助前端使用户能够方便的操作系统（包括查看数据、下达指令），前端与后台之间需要有一个接口以供通信交互，相互之间收发数据信息。在这个物联网系统中，前端与后端之间的通信是通过 RESTful API 实现的，客户端通过HTTP请求与服务器进行交互。前端使用现代Web框架如React、Angular或Vue.js构建用户界面，并通过Ajax技术发起异步请求，无需刷新页面即可与后端交互。这些请求通常携带JSON格式的数据，以GET方法获取资源或以POST方法发送资源。后端使用Spring Boot等框架处理这些请求，定义控制器来匹配特定的URL路径和HTTP方法。后端执行业务逻辑，可能涉及数据库操作、数据验证等，并将结果封装成JSON格式通过HTTP响应返回给前端。前端接收到响应后，通过JavaScript解析JSON数据，并利用解析后的数据更新用户界面。此外，前端还可以使用状态管理库如Redux或Vuex来管理应用状态，以及使用Axios或Fetch API来发送HTTP请求。后端则使用MySQL存储结构化数据，并通过Spring Security等工具来实现安全性。整体而言，前端与后端之间通过HTTP请求和响应进行数据交换，实现了物联网系统中数据的有效处理和展示。

### **3.4.4 数据安全**

还需要补充的是安全也是物联网系统必须考虑的因素。为了保证数据的安全和隐私，系统需要实施有效的身份验证和授权机制。这意味着只有经过验证的用户才能访问敏感数据或控制设备。此外，系统的监控和维护也非常重要。定期检查系统的运行状态，并对数据库进行必要的维护（如备份、恢复、性能调优等），能够确保系统的长期稳定运行。

### **3.4.5 可扩展性 / 可迭代性**

另外，考虑到未来系统需要升级迭代，亦或者是为了适应相似但又有不同的应用场景，系统的设计必须考虑到可扩展性、容错性和高可用性。这意味着系统需要能够在负载增加时自动扩展资源，系统本身会被设计成模块化结构，模块本身会被封装，只提供接口，这意味着如果因实际需求而要修改系统时，只需要针对相关模块进行修改即可。同时还要具备处理故障的能力，确保即使在部分组件出现故障的情况下也能继续提供服务。为了实现这些目标，通常会选择使用容器化技术和编排工具，如Docker和Kubernetes，来管理和部署应用程序。

## **3.5 前端交互（ Web / Mobile ）**

在物联网系统中，我们需要提供一个简洁易用的界面给用户来进行操作，使用户无需为此进行多少专业的培训就能够熟练操作系统，参看需要的数据信息，并根据实际需要下达指令。就目前的技术趋势而言，不管是local还是Web还是Mobile，全部都采用图形化界面（GUI），我们也不例外。Web前端和移动端（Android）是前端两个主要组成部分，它们负责与用户的交互，并展示来自后端的数据。

### **3.5.1 Web**

关于Web前端工作原理及技术栈，Web前端主要关注的是通过浏览器来展示信息并与用户交互。在物联网系统中，Web前端负责接收来自后端的数据，并将其呈现给用户。为了实现这一点，Web前端通常使用三种技术栈： HTML / CSS / JavaScript ，这是Web前端的基础技术栈，用于构建网页布局、样式和动态功能。同时为了提高开发效率和构建复杂的用户界面，我们会使用现代前端框架如React、Angular或Vue.js。这些框架提供了丰富的功能，如组件化开发、虚拟DOM等，可以帮助我们能够更快地构建高质量的应用，而无需重复低效的复现相同的功能，即所谓的“造轮子”。使用这些前端框架提供的组件构建用户界面，并展示数据。这些框架支持事件监听和动态渲染，使得用户界面能够响应用户的操作。为了更好地管理应用的状态，我们会使用Redux、Vuex等状态管理库，来保证程序能够及时的响应新的状态变化。在与后台通信的过程中，前端通常使用 HTTP 请求库的Axios或Fetch API来与后端通信，发起HTTP请求以获取数据或提交数据。Web前端通过发送HTTP请求到后端API，获取设备数据或其他相关信息。这些请求通常使用GET或POST方法，并且数据通常以JSON格式返回。前端接收到数据后，使用JavaScript对其进行解析，并将其转换为可视化的形式。这可能包括数据的筛选、排序、格式化等。同时对于需要实时数据更新的应用场景，我们会使用WebSocket技术实现实时双向通信。具体到应用开发层面，为了加快开发速度并保持一致的视觉风格，我们将使用Bootstrap、Material-UI、Ant Design等UI组件库。

### **3.5.2 Mobile**

Android移动端前端工作原理及技术栈，Android移动端前端是指用于构建Android应用的用户界面的部分。与Web前端相比，Android前端更多地依赖于原生技术，但是如果有必要在非Android IDE下进行开发，也有一些跨平台的解决方案可以选择。我们将使用Java语言进行开发。这些语言是Android官方推荐的语言，提供了完整的原生开发API集合来构建高性能的应用。针对用户界面展示的具体实现来说，我们将使用Android的布局文件（XML）来定义用户界面，并使用Java或Kotlin代码来填充数据。这些布局文件支持各种控件，如TextView、ImageView等。如果在实际开发中发现有必要，将会使用React Native、Flutter等跨平台框架，在代码库中编写应用，然后编译为Android或iOS应用。这些框架使用JavaScript语言，提供了丰富的组件库和API，可以快速构建应用，实现跨平台开发。在Android应用中，我们使用OkHttp、Volley等HTTP库中的请求库来发起HTTP请求，获取数据。并且与Web端相似的是，Android端将用同样的方法和格式，来发起或响应请求以及处理数据，包括专门针对需要实时更新数据的情况中，也采用WebSocket技术。在Android原生开发过程中，我们将使用Data Binding Library来实现数据与视图的绑定，简化界面更新逻辑。Android应用需要处理不同的生命周期状态，如启动、暂停、停止等。我们需要理解这些状态，并正确处理相关事件，以实现对生命周期的管理。对于复杂的应用，我们依然要实现稳定的状态管理。我们在Android应用中将通过保存状态或使用MVVM模式来管理应用的状态。

另外，后期如有可能，我们将考虑针对其他移动端平台进行兼容适配工作，例如 华为 Harmony OS 、苹果 IOS 等系统

# **四、测试方法**

对于BMS26M833的测试，在室内2m外点燃火焰，所得到的矩阵温度中高温直接超过测量范围。且仅有该值饱和，其他矩阵表格则不受影响。

对于物联网平台的测试，主控芯片HT32F52367使用串口与ESP8266进行通讯，使用AT指令控制其开始MQTT协议初始化并连接物联网平台，实测中初始化的速度与网络质量相关。

对于K210模块的测试，在上面部署YOLO-v5模型，识别火焰图像，并且使用串口向合泰主控通信，发出识别到的火焰图像的中心坐标

# **五、作品结论**

## **5.1 总述**

本项目已初步实现封闭环境无人自动巡检系统（CEAGVIS）搭建，利用物联网技术和其他先进技术，以低成本、高效率管理相对封闭空间。未来，我们计划优化系统界面设计、PCB布局，并修复或更换故障部件。同时，我们引入了华为海思Hi3519DV500芯片的IPV09A工作站和昇腾Atlas 200I DK开发板替换初始的K210模块，实现系统计算能力和智能化水平的提升。这些改进措施旨在进一步增强系统稳定性和功能性，为用户提供更高效、可靠的无人自动巡检解决方案。

## **5.2 系统核心创新点**

**1. 技术整合、应用范围广：**

多项先进技术共同应用，为仓库、办公室和实验室提供全方位的自动化监测解决方案，涵盖物联网、AGV、边缘计算及消防安全等。为仓库、办公室、实验室等多场所提供全面的自动化监测解决方案。

**2. 火灾响应优化：**

结合传感器和边缘模型预测技术，实现火灾源头迅速定位和灭火响应的高效方法。

**3. 完善的系统架构：**

设计了完整的软硬件体系，包括交互界面、后台数据库和监控网络，支持远程监控和实时数据传输，确保系统运行的稳定性和可靠性。

## **5.3 项目技术难点：**

**1. 控制精度提升：**

采用PID算法实现AGV精准导航和避障，确保其在复杂环境中运动控制的可靠性.

**2. 火灾识别升级：**

在边缘计算模块中引入YOLO-v5模型，用于火焰图像识别，显著提升了火灾预警的准确性和响应速度。

**3. 高效数据通信：**

开发了稳定的物联网平台，优化AGV与服务器间的数据传输，支持实时监控和远程控制功能的顺畅运行。

## **5.4 项目不足之处：**

**1.用户体验升级方向：**

未来将进一步完善物联网平台的火情通知前端界面，以提供更加友好和直观的用户体验。

**2.PCB布局优化策略：**

目前的PCB布局存在改进空间，计划重新设计以提升系统的可靠性和维护性，确保长期稳定运行。

**3.技术更新路径：**

当前云台芯片出现故障，将更换新芯片以恢复正常功能，并加强硬件设施，进一步增强系统整体可靠性和性能表现。

# **六、参考文献资料**

[1]灭火器的使用方法系列.中华人民共和国中央人民政府

[2]于群, 戴敏达, 余书慧. 基于二维双线性插值的流量计量温度补偿算法研究[J]. 自动化仪表, 2020, 41 (11): 28-32.

[3]《神经网络与深度学习》邱锡鹏 飞桨教材编写组 著

[4] Ultralytics团队 YOLO-v5 官方文档（GitHub）

[5] HUAWEI-HISILICON IPV09A-04D 智能工作站官方文档

[6] 华为昇腾社区（CANN）文档

# **七、其他及附录**

## 附件一：作品中使用的硬件模块清单

|  |  |  |  |
| --- | --- | --- | --- |
| 作品使用硬件模块名称、型号、规格 | 硬件厂牌 | 数量 | 用途 |
| BMS26M833 | 合泰半导体 | 1 | 温度传感器，接收视野内温度 |
| BMP73T104 | 合泰半导体 | 1 | 驱动电机 |
| K210 | 嘉楠 | 1 | 图像识别 |
| MG513编码电机 | 无 | 3 | 运动控制、灭火器阀门按压 |
| D500激光雷达 | 乐动 | 1 | 自动巡航的地图获取 |
| L515激光雷达 | Intel | 2 | 路径规划/避障 |
| HUAWEI Atlas 200I DK A2 开发板 | 华为 | 1 | 边缘模型部署  硬件电路控制 |
| HUAWEI OrangePi KunPeng Pro 开发板 | 华为 | 1 | 边缘模型部署  硬件电路控制 |
| 华为海思 IPV09A 计算盒子 | 英码信息 | 2 | 边缘模型部署 |

## 附件二：参赛队伍合照

从左到右：刘洁宇、陈永铎，吴显，黄泊凯

人们站在一起合影的一群人

描述已自动生成