

**贵州理工学院**

**本科毕业设计（论文）**

**设计（论文）题目**：**四轴八旋翼载重无人机智能移动抓钩控制软件设计**

|  |  |
| --- | --- |
| **学 院：** | **航空航天工程学院** |
| **专 业：** | **飞行器动力工程** |
| **班 级：** | **飞 动 191** |
| **学 号：** | **2019220109** |
| **学生姓名：** | **王 胜 贵** |
| **指导教师：** | **杨 雄** |

2023年6月2日

# 目 录

[目 录 II](#_Toc136378053)

[摘 要 IV](#_Toc136378054)

[关键词 IV](#_Toc136378055)

[Abstract V](#_Toc136378056)

[第一章 绪论 1](#_Toc136378057)

[1.1研究的目的与意义 1](#_Toc136378058)

[1.1.1 设计目的 1](#_Toc136378059)

[1.1.2 设计的意义 1](#_Toc136378060)

[1.2 国内外研究现状 2](#_Toc136378061)

[1.3主要设计内容 3](#_Toc136378062)

[1.4 论文结构安排 4](#_Toc136378063)

[1.5本章小结 4](#_Toc136378064)

[第二章 控制模块选型和软件控制原理 1](#_Toc136378065)

[2.1 单片机型号选择 1](#_Toc136378066)

[2.2 基本控制理论 1](#_Toc136378067)

[2.2.1 PID控制 1](#_Toc136378068)

[2.2.2 卡尔曼滤波 2](#_Toc136378069)

[2.3嵌入式系统简介 2](#_Toc136378070)

[2.4 本章小结 3](#_Toc136378071)

[第三章 软件设计流程与程序设计 4](#_Toc136378072)

[3.1智能移动抓钩软件控制流程 4](#_Toc136378073)

[3.2通信接口设计 5](#_Toc136378074)

[3.2.1 IMU通信设计 5](#_Toc136378075)

[3.2.2 OpenMV通信设计 6](#_Toc136378076)

[3.3自稳云台装置设计 8](#_Toc136378077)

[3.4视觉识别模块设计 9](#_Toc136378078)

[3.5升降装置设计 9](#_Toc136378079)

[3.6抓钩移动设计 10](#_Toc136378080)

[3.6.1 单片机与飞行控制器的通信设计 10](#_Toc136378081)

[3.6.2 通信协议的分析与设计 11](#_Toc136378082)

[3.6.3 抓钩移动的实现 12](#_Toc136378083)

[3.7本章小结 13](#_Toc136378084)

[第四章 软件编写与调试 14](#_Toc136378085)

[4.1 基于OpenMV IDE的视觉识别代码实现 14](#_Toc136378086)

[4.1.1 颜色阈值设定 14](#_Toc136378087)

[4.1.2 中心坐标采集与传输 14](#_Toc136378088)

[4.2 基于STM32CubeMX的控制代码实现与调试 15](#_Toc136378089)

[4.2.1 系统初始化配置 15](#_Toc136378090)

[4.2.2 系统仿真 16](#_Toc136378091)

[4.3本章小结 16](#_Toc136378092)

[第五章 总结 17](#_Toc136378093)

[5.1 总结 17](#_Toc136378094)

[5.2 展望 17](#_Toc136378095)

[参考文献 18](#_Toc136378096)

[致谢 20](#_Toc136378097)

[附录 21](#_Toc136378098)

四轴八旋翼载重无人机智能移动抓钩控制软件设计

# 摘 要

随着微电子技术和软件控制技术的发展，嵌入式控制系统广泛应用于无人机、智能家居、工业控制等领域。本文针对载重无人机移动抓钩的精准定点以及自动抓放问题，以OpenMV、无人机、控制器等为硬件模块，软件部分以Keil MDK、OpenMV IDE为开发环境，编写软件控制代码，以实现抓钩的智能移动。

首先对控制系统进行分析，选择控制模块。其次，以PID控制算法为核心编写各模块的控制代码，将各模块数据进行结合，针对智能移动抓钩设计一种软件控制系统，使抓钩能够自动识别目标点，并自动将抓钩精准移动到相应位置，并可自动抓放重物。最后对控制软件进行模拟仿真，仿真结果表明，当待抓放重物在可识别范围内时，能够自动对准目标点，抓取目标点重物。

关键词：软件控制，移动抓钩，PID控制，自动对准

Four-axis eight-rotor payload drone intelligent mobile hook control software design

# Abstract

With the development of microelectronics and software control technology, embedded control systems are widely used in fields such as drones, smart homes, and industrial control. This article focuses on the precise positioning and automatic grasping and releasing of the mobile hook of a payload drone. OpenMV, drones, controllers, etc. are used as hardware modules, and Keil MDK and OpenMV IDE are used as development environments for the software part. Software control code is written to implement the intelligent movement of the hook. Firstly, the control system is analyzed and the control module is selected. Secondly, the control code of each module is written with the PID control algorithm as the core, and the data of each module is combined to design a software control system for intelligent movement of the hook, so that the hook can automatically recognize the target point, move the hook accurately to the corresponding position, and automatically grasp and release heavy objects. Finally, the control software is simulated and the simulation results show that the target can be automatically aligned with the target point and the target goods can be grasped within the recognizable range.

# 第一章 绪论

## 1.1设计目的与意义

### 1.1.1 设计目的

无人机技术的不断发展，衍生出适用于各种场合的无人机，如农业植保、航拍摄影、地形勘测、消防无人机等。但是关于载重无人机的研究较少，且大多为半自动，无法满足特定的需要。为了实现载重无人机自动化运输，使用自动移动的抓钩和外部传感器，实现载重无人机自动化运输，使载重无人机重物运输由半自动变为自动，可大大提高效率。

### 1.1.2 设计意义

随着现代化信息技术的不断发展，装置的设计不仅仅是结构上的设计，还包括电路部分的设计。为了使装置更加智能化，还需要进行软件设计。只有将这三者结合起来，才能够组成一个完整的系统。在这个系统中，控制器通过将各种传感器采集的数据进行融合和判断，能够让机构实时感知外界环境，做出相应的回避动作，防止机构运行过程中产生意外。

本课题的主要研究内容是抓钩的软件设计。为了使抓钩能够精准地和重物对接，可以采用手动的方式进行辅助。但是，在无人机运行中难免会产生意外。一方面是机械的损伤，例如在运行中，周围存在树木等障碍物，无人机将会触碰到障碍物，这将会使无人机直接坠落，造成非必要的经济损失。另外一方面，会对操作人员产生安全隐患。若无人机运行不平稳，有可能会对周围人员造成伤害。

因此，设计出能一键操作的软件控制系统，并且抗外界干扰能力强，这将会让无人机平稳运行。又因为有外部高速传感器的辅助，不仅能保证无人机和操作人员的安全性，同时无人机会自动将抓钩移动到要装载的货物上方，且能够自动抓放。与此前的半自动载重无人机相比，能够快速地运输并提高运输效率。

因此，对无人机设计出一种抓钩，并设计出抓钩的软件控制系统，使载重无人机向智能化发展，使载重无人机实现自动装载和运输是非常有必要的。本设计通过设计无人机控制辅助系统，对载重无人机进行辅助控制，解决现有载重无人机存在的不确定性和关键问题。

## 1.2 国内外研究现状

为了使无人机实现定点起降功能，国内外的许多专业人士做出了很多的研究和贡献，包括各种机械结构、控制电路、控制软件等。无人机能够做出各种高难度动作，离不开各种巧妙的机构，硬件传感器和处理器的支持，同时也需要控制算法和控制软件才得以实现。智能移动抓钩的主要软件设计内容为借助控制器和各种传感器，通过软件控制算法对其控制，使其自动识别货物和自动将抓钩精确移动至货物上方。

二十一世纪是电子技术快速进步的世纪，各种传感器层出不穷，应用于无人机实现各种功能，且计算机视觉识别的应用越来越广。早在上世纪九十年代，国外一些大学与科研机构就已经将计算机视觉应用于无人机领域，相关技术发展至今已经比较成熟。与国外相比，国内无人机技术起步较晚，在自主着陆控制技术上相对落后，目前技术还未成熟，处于起步发展阶段。目前计算机视觉在无人机领域的研究主要集中在无人机的视觉导航以及相对位姿的估计两个方面[1]。

近年来，高性能处理器不断发展，用来处理视频的MCU越来越多，如OpenMV、OpenCV等传感器的应用。通过装载视觉传感器对目标进行位置检测，与无人机进行配合，移动到检测目标上，实现点对点的货物装卸。目前国内实现精准对点的方案主要有两种，基于RTK-GPS的精准对接、基于图像识别（视觉）的精准对接[2]。在实际使用中，一般使用双目相机成像进行点位测量，双目视觉测量是基于视差的原理，利用两个相机在同一时刻从不同位置采集被测目标的两幅图像信息，通过计算图像中对应视点的位置来获取目标在空间中的位置[3]。

在对精准定点起降的研究中，国内学者李勤文对基于视觉反馈的自动对准系统有过关键技术研究，重点解决了测量系统中两个非公共目标相机位姿关系的标定问题，基于DLT－LM算法提出了双相机在线标定方法。其结果表明基于DLT－LM算法的双目相机标定方法求出的目标点距离均方差优于０.２ｍm[4]。吴文秀，关志全等人提出了一种GPS和视觉识别相结合定点降落方法。首先通过pixhawk飞控模块配合M8N模块来进行GPS定位，使无人机到达目标附近，然后通过再启用视觉辅助(OpenMV)实现较精准的降落。实验结果表明，无人机能够准确识别充电基站从而实现定点降落，OpenMV搭载到无人机上时，要注意使用自稳云台，确保无人机在调控姿态时摄像头始终是垂直向下的，使目标始终在摄像头检测区域。自稳云台的控制可设计目标跟踪控制器，通过对云台模型的分析，使用PID控制规律设计云台的目标跟踪控制器，控制云台的转角，保持相机光轴实时指向地面目标。为了确保正确跟踪目标，对云台角度跟踪控制的要求主要有两个方面：一是快速、平滑的响应，超调量要小；二是跟踪精度要高[5]。

国内对于视觉精确定点研究的成果还有很多,也可以看出我国的研究爱好者们对于无人机精确定位的研究具有浓厚的兴趣。

过去，国外许多研究人员已经对基于视觉的着陆系统进行了详细研究和解决。2011年，Olson等人使用AprilTag标识符来构建基于视觉的着陆检测的独特平台。2019年，Kashish Gupta等人在基于视觉的着陆系统中根据目标的定向快速和旋转BRIEF特征，提出了一种强大的视觉数据处理系统，以实时估计无人机的三维姿态。结果显示该系统能够基于协作标记直观地定位和识别独特的着陆平台，该标记在横滚、俯仰和偏航角的错误率为1°或更低。结果显示该系统可提高基于视觉的系统的可靠性，且不会显著改变无人机的计算负载，使其在低速处理器中得到广泛应用[6]。

## 1.3主要设计内容

本设计旨在设计一款智能移动抓钩，使用stm32f103单片机作为主控，协助飞控系统完成抓钩的智能移动，设计难点在于抓钩的智能移动和精准对接。具体工作包括：

1. 模拟无人机信号接收器，将各个传感器的输入信号进行融合和判断，得到PWM控制信号并输出到无人机进行方向控制；
2. 采用视觉识别传感器采集目标，进一步处理得到具体坐标数据，通过USART将数据发送至处理器，通过处理器控制无人机飞行，以实现无人机对目标的识别和跟踪；
3. 使用双轴云台并对其进行水平角度控制，以保证传感器始终指向目标；
4. 编写子无人机升降控制软件，以实现对抓钩的智能移动；
5. 计算无人机和目标的偏差，将无人机移动至目标点上方，以实现精准抓取；
6. 建立飞行控制器和辅助系统的信息交流、双向通信、协同工作，以实现系统的智能化；
7. 综合分析各个传感器的功能、通信协议和接口，设计出所需的控制软件，以实现系统的高效稳定运行。

## 1.4 论文结构安排

本设计旨在设计一款智能移动抓钩，使用stm32f103单片机作为主控，协助飞控系统完成抓钩的智能移动。具体工作包括：模拟无人机信号接收器，将各个传感器的输入信号进行融合和判断，得到PWM控制信号并输出到无人机进行方向控制；采用视觉识别传感器采集目标，进一步处理得到具体坐标数据，通过USART将数据发送至处理器，通过处理器控制无人机飞行，以实现无人机对目标的识别和跟踪；采用双轴云台并对其进行水平角度控制，以保证传感器始终指向目标；编写卷扬机和子无人机升降控制软件，以实现对抓钩的智能移动；计算无人机和目标的位置偏差，将无人机移动至目标点上方，以实现精准抓取；建立飞行控制器和辅助系统的信息交流、双向通信、协同工作，以实现系统的智能化；综合分析各个传感器的功能、通信协议和接口，设计出所需的控制软件，以实现系统的高效稳定运行。本设计报告分五个章节，其内容如下：

第一章，阐述无人机智能移动抓钩的设计背景以及研究的意义，引出载重无人机的智能移动抓钩，随后对视觉识别的算法和方案进行分析和总结。

第二章，结合设计需要对主要元器件和模块选型，分析无人机和飞行控制器的控制原理以及软件控制软法，对控制系统选择合适的控制算法。

第三章，整理设计思路，画出控制系统的流程图并根据流程图设计出合适的控制算法。分析硬件模块的通信协议，用单片机设计出相应的控制软件，分别设计各模块的控制软件，再融合各传感器数据对整个控制系统进行优化。

第四章，针对所选模块和控制器，选择相应的代码编写软件，编写软件控制代码，使用编译软件的仿真功能跟踪和监视系统的运行。

第五章，对设计的结论进行了总结，指出本次软件设计中的不足，对以后智能移动抓钩的发展趋势作简单分析和展望。

## 1.5本章小结

本章说明了课题研究的目的和意义，分析和总结国内外对无人机精准定点的研究成果和现状，阐述了课题主要的设计难点，并列举了主要的的具体设计内容，最后对论文进行结构安排。

# 第二章 控制模块选型和软件控制原理

结合课题需求，四轴八旋翼载重无人机要实现自动抓取货物，需要智能移动抓钩辅助完成，为了提高抓钩的精度，需要视觉识别模块对目标点进行定位，选用合适的单片机和控制算法，将抓钩移动到目标点上方，实现自动抓放货物的功能。

## 2.1 控制器型号选择

。STM32系列单片机具有高性能、丰富的外设、低功耗、易用性、可靠性等优点，适用于多种场景，满足不同领域需求。与其他类型单片机相比，STM32有多种软件库，如STM32标准库、LL库和HAL库，以及开发工具和编译器配合使用，如Keil、IAR、GCC等。此外，STMicroelectronics公司还开发了一款软件工具CubeMX，用于支持STM32系列微控制器的开发，可以帮助开发人员快速生成初始化代码和配置文件，以便更快地进行开发和调试。选择STM32F103C8T6单片机作为系统主控，具有高性能、丰富的外设、数据传输效率高、低成本、稳定可靠等优点，可以满足智能移动抓钩设计需要。

## 2.2 基本控制理论

### 2.2.1 PID控制

PID控制理论是一种经典控制理论，适用于各种控制系统中，如电机调速、无人机、温控系统等，控制思想是检测系统误差，自动调整系统的输出信号，使误差尽可能减小，最终稳定在期望值附近。PID控制如图2-1所示。

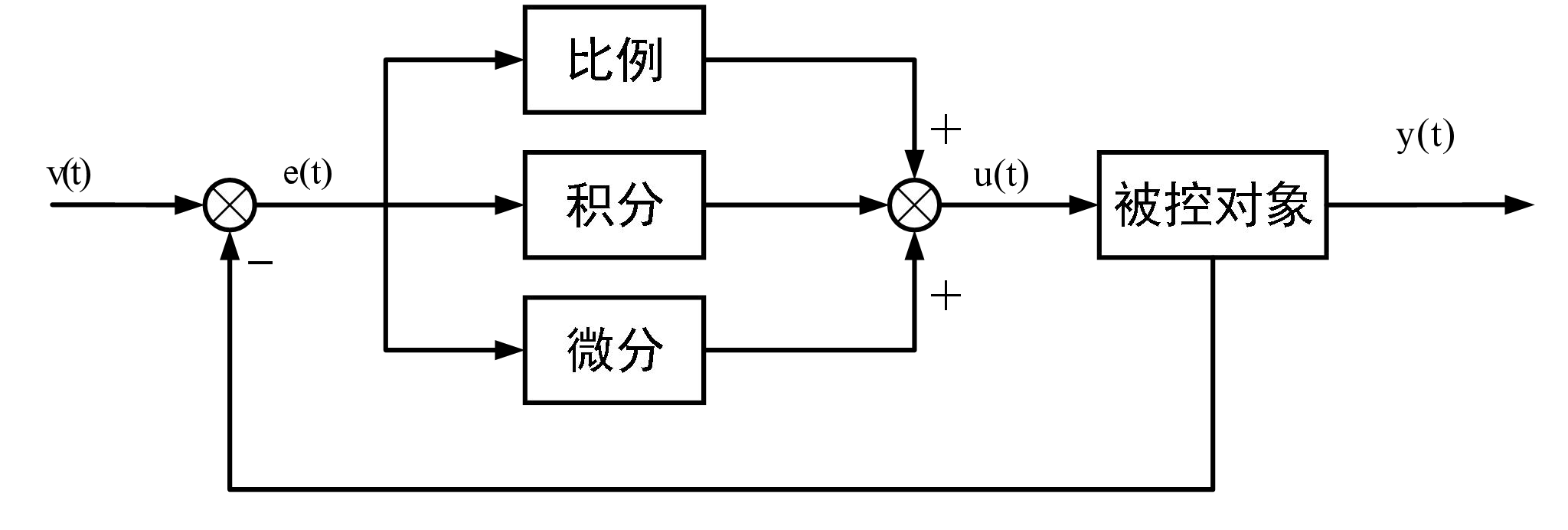


图2-1 PID控制器结构

图2-1中v(t)为期望值，y(t)为实际值，e(t)是期望与实际值的偏差：

(2-1)

PID将系统的输出经过比例、积分、微分三种运算后,叠加至输入中，以达到控制效果。在实际应用中，PID控制有两种，即位置式和增量式，位置式控制的表达式为：

(2-2)

增量式控制的表达式为：

(2-3)

由于控制过程中常带有非线性、时变性和不确定性，采用常规PID控制器往往不能获得令人满意的效果，且PID受参数整定的影响较大，若参数整定不良，将导致其控制性能下降，自适应性变差[7,22]。

### 2.2.2 卡尔曼滤波

1960年，Kalman发表了他的著名论文，描述了离散数据线性滤波问题[Kalman60]的递归解。从那时起，在很大程度上由于数字计算的进步，*卡尔曼滤波器*一直是广泛研究和应用的主题，特别是在自主或辅助导航领域[8]。卡尔曼滤波的应用非常广泛，在降噪和预测中都发挥出巨大优势，卡尔曼滤波器包括两个主要过程：预估与校正[9]，在本设计中，需要完成目标的识别和跟踪，使用卡尔曼滤波器对系统进行预测，可以提前对系统进行控制。卡尔曼滤波有两种建模方法：一种是基于控制理论通过运动模型根据前一时刻的状态来估计得到当前时刻的最佳估计值与协方差矩阵，以此类推进行下一时刻的最佳估计，这个建模求解的重点在于卡尔曼增益；另一种是利用一些概率性质进行建模的概率模型。通过运动系统方程、测量模型以及它们的误差来求得卡尔曼增益，随后用卡尔曼滤波后的值反过来修正误差，从而在行车过程中获得最佳的轨迹预测[10]。

## 2.3嵌入式系统简介

嵌入式系统是一种将计算机、通信和控制技术相结合的微处理器系统，是一种面向具体应用的将底层硬件、实时操作系统和应用软件相结合的专用计算机系统。嵌入式系统具有硬件集成度高、接口丰富、控制精度高、可靠性高、实时性强等特点，它已经用于各种工业控制中，如广泛应用于消费电子、通信、汽车、国防、航空航天、工业控制、仪表和办公自动化等领域。目前，在工业控制中嵌入式控制系统正朝着高速、高集成度和低功耗方向发展[19-21]。

## 2.4 本章小结

本章主要介绍了智能移动抓钩控制系统的硬件选型和软件系统的支持。在硬件方面，为了使抓钩能够稳定工作且降低设计成本，采用了stm32f103作为主控制器，该处理器具有高性能和丰富的外设资源和硬件接口，能够满足控制系统的要求。在货物识别方面，采用了视觉识别的方法，选用了开源平台OpenMV作为视觉识别元件，既能够提高精度，又能够降低开发难度和成本。在软件方面，介绍了无人机控制系统的控制软件和相关算法，比较了各种算法的优缺点和适用性，根据无人机所需不同的功能选取相应的算法，云台和追踪目标都可使用PID进行控制。

# 第三章 软件设计流程与程序设计

## 3.1智能移动抓钩软件控制流程

本系统使用四旋翼无人机挂载移动抓钩，通过控制无人机实现抓钩的智能移动。在运行时，当载重无人机飞至货物上方时，系统开始工作，利用OpenMV图像采集模块识别货物，并通过串行通信将货物所在相对坐标发送至stm32f103处理器。为保持图像采集模块平稳和平衡，采用二轴云台控制模块进行平衡控制；使用单片机的串口中断采集IMU数据（姿态角）作为舵机的控制量，通过两个SG90舵机分别控制X，Y轴的平衡。根据采集到的目标位置信息控制无人机的运动，使其移动到目标点上空，再通过超声波模块测量距离，控制卷扬机平稳移动抓钩到货物上，完成装载。

根据载重无人机智能移动抓钩系统的工作要求，软件控制流程可以根据各模块的工作，划分为以下几个结构：

1.SG90舵机云台：本系统采用SG90舵机云台，将其和IMU一起安装在四旋翼无人机上。当无人机受到外部干扰或者因移动产生倾角时，单片机通过采集IMU上的姿态角数据（0-90度）控制SG90舵机，以保持云台的工作面始终平放，确保图像采集的水平度和质量。

2.图像采集模块：采用OpenMV开源平台做视觉识别模块。在运行过程中，OpenMV通过采集无人机下方图像，识别特定波长的光源/颜色块，并检测该颜色块的相对坐标位置。然后将所检测出的坐标通过USART发送到处理器进行后续处理。

3.卷扬装置模块：本系统在载重无人机上安装卷扬机，并连接到四旋翼无人机上。使用HCSR04超声波测距模块测量四旋翼无人机到地面的距离，并反馈到卷扬机上，控制其放线长度，以使载重无人机不受载重绳的影响。

4.处理器模块：使用stm32f103作为控制系统的主控制器，负责整个控制系统的运行。通过使用其内部通信总线、定时器、硬件接口，完成包括数据读取、数据传输、数据处理等功能。

软件控制流程图如图3-1所示。

图示

描述已自动生成

图3‑1 软件控制流程

## 3.2通信接口设计

### 3.2.1 IMU通信设计

通用同步异步收发器（Universal Synchronous/Asynchronous Receiver/Transmitter，USART），简称串口，为与使用工业标准 NRZ 异步串行数据格式的外部设备之间进行全双工数据交换提供了一种灵活的方法[11]。它是一种通用的串行收发器，支持双向数据传输，可以同时发送和接收数据。它包括一个发送器和一个接收器，可以通过配置设置来控制数据的传输方式和参数。USART使用多个控制位来定义传输的数据位数、停止位数、校验位类型等。

在异步通信模式（UART模式）下，USART通过一个数据线（TX）和一个接收线（RX）实现数据的逐位传输。数据以字节为单位进行传输，并通过波特率（Baud Rate）来控制数据传输的速率。同步模式下，通信具有更高的传输速率和更低的误码率，适用于对数据传输速度和精确性要求较高的应用场景。

USART常用于嵌入式系统、通信设备、传感器和其他需要进行串行数据传输的应用中。它是一种灵活、可配置和广泛支持的串行通信接口，具有较高的可靠性和兼容性。USART通信协议时序图如图3-2所示。



图3‑2 USART通信协议时序图

本设计采用ATK-MS901M模块来检测无人机的姿态角，并将其用于控制云台。该模块通过USART协议与单片机通信，数据传输时按照特定的帧格式进行。通信帧可以分为两类：模块主动上报的数据帧和上位机发送的指令帧。ATK-MS901M模块默认会通过USART不断向单片机上报内部传感器测量并经过初步处理后的数据。数据帧的格式包括自姿态角、四元数、陀螺仪和加速度计、磁力计、气压计、端口状态等信息，具体的帧格式详见表3-1。

表3-1 ATK-MS901M模块主动上报数据帧格式

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 帧头 **1** | 帧头 **2** | 帧 **ID** | 数据长度 | 数据 | 校验和 |
| 1 字节 | 1 字节 | 1 字节 | 1 字节 | N 字节 | 1 字节 |
| 0x55 | 0x55 | ID | N | DATA | SUM |

ATK-MS901M 模块主动上报姿态角数据帧的帧格式，具体的帧格式详见表3-2。

表3-2 模块主动上报姿态角帧格式

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 帧头 1 | 帧头 2 | 帧 ID | 数据长度 | 数据 | 校验和 |
| 0x55 | 0x55 | 0x01 | 0x06 | DATA | SUM |

根据硬件条件和项目需求，可对ATK-MS901M模块的USART接口进行以下设置：将波特率设置为115200，无校验位，1位停止位，以同时满足速度要求和STM32串口支持的波特率。系统在接收姿态角数据时，需要对数据帧进行校验，以确保数据在传输过程中的准确性。此外，需要对USART接收中断设置合理的优先级，既不占用大量CPU资源，也不会使系统运行出错。还需要设置合理的接收缓冲区，使系统有一定的RAM余量，确保系统有足够的运行空间。

### 3.2.2 OpenMV通信设计

视觉定位采用开源视觉模块OpenMV，OpenMV是一种低成本、低功耗，基于stm32和摄像头模组的小型机器视觉跟踪模块，是一种开源和性能高效的机器视觉跟踪模型[12]。模块事物如图3-3所示。依靠其高性能、低成本、开源的优点，常用于嵌入式系统，能够使用Python语言实现图像检测、目标追踪、智能寻迹等功能[13]。



图3‑3 OpenMV模块实物图

在使用OpenMV采集图像和识别时，需要和单片机进行数据通信，可在OpenMV上配置USART串口用来输出坐标位置。可自定义具体的帧格式，详见表3-3。

表3-3 数据输出帧格式

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 帧头1 | 帧头1 | X高8位 | X低8位 | Y高8位 | Y低8位 | 帧尾 |
| 0x2C | 0x12 | DatxH | DatxL | DatyH | DatyL | 0x5B |

OpenMV在图像采集和识别过程中，需要将识别结果通过USART串口发送给单片机进行处理。OpenMV最小图像分辨率为320x240，模块在识别目标时坐标数据可能大于8位（28-1），如一次检测中坐标信息为（260，180），而USART每次只能发送一个字节，因此大于一个字节的数据需要分割成两个char类型多次发送。接收端需要进行数据解析和帧判断，将两个char类型数据转换成int数据，才能正确获取发送端输出的数据。

## 3.3自稳云台装置设计

舵机也称伺服马达通过控制摆臂旋转精确的角度达到控制目的，其具有闭环控制系统的机电结构[14]。使用舵机内部的位置闭环控制可以使舵机稳定在一定角度，从而稳定云台，云台安装于四旋翼无人机下方，用于稳定OpenMV模块，云台由两个转动轴x-y组成，通过两个SG90舵机分别控制其转动角度，保持OpenMV的平衡，舵机实物如图3-4所示。根据舵机的控制原理，使用单片机定时器功能输出PWM信号对舵机进行控制，PWM的英文全称是”Pulse Width Modulation”，中文翻译为"脉冲宽度调制"[18]。通过改变PWM的占空比即可控制角度，控制信号的脉冲宽度和对应角度如图3-5所示，输出PWM可以使用stm32定时器功能输出，SG90的控制一般需要输出50Hz的PWM波，脉宽范围为500us-2500us，分别对应角度0-180。云台的控制需要使用姿态角（通常包括俯仰角、横滚角和偏航角横滚角（roll）、俯仰角（pitch）和航向角（yaw）[15]）来描述物体的旋转状态。因为姿态角可以避免欧拉角存在的万向锁问题，并且在计算机图形学和机器人控制等领域得到广泛应用。在云台控制中，需要实时获取云台的旋转状态，并根据需要进行调整，使用姿态角可以更加准确地描述云台的旋转状态，从而实现更加精确的控制。同时，使用姿态角也可以方便地进行数学计算和处理，提高控制的效率和精度。利用ATK-MS901M IMU模块获取的姿态角控制舵机角度，舵机只需分别控制云台的横滚角和俯仰角，即可保持云台在水平方向上的平衡。

图片包含 游戏机, 电缆

描述已自动生成

图3‑4 SG90舵机实物图

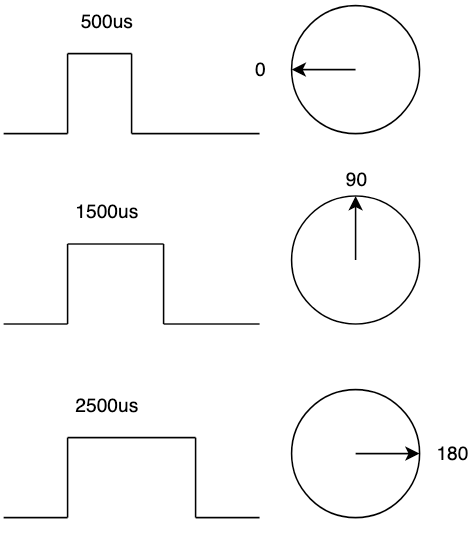


图3‑5 舵机PWM控制信号

## 3.4视觉识别模块设计

视觉识别模块选用OpenMV，模块安装于四旋翼无人机下方，用于采集目标点坐标。OpenMV模块采用高分辨率摄像头进行图像采集，支持实时图像处理和视觉识别算法。图像处理流程包括图像预处理、特征提取、目标检测和跟踪等步骤，以获取无人机相对于降落区域的位置和姿态信息。

可在OpenMV IDE中对特定颜色块进行识别和学习，得出该颜色的RGB组成，设置合适阈值，写入FLASH中，在系统启动时OpenMV将自动识别该颜色，并实时获取该颜色的中心坐标，通过通信接口向控制器上传坐标信息。OpenMV支持多种分辨率QVGA(320x240)、VGA(640x480)、HD(1280x720)等，系统运行中，需要实时获取目标点信息，过大的分辨率设置会影响处理速度，会使信息获取严重滞后，导致系统效率降低，因此可将图像分辨率设置为QVGA(320x240)，既保证图像清晰又提高运算速度。使用过程中，可能会因外界光线干扰而导致识别出现问题，在处理时可以寻找面积最大的色块作为最终输出，可避免因干扰产生非必要的误差。由于OpenMV采集坐标的数据可能会受到机械振动和不可预测因素的干扰或数据传输错误，使坐标信息产生高频波动，而信息采集频率比较低，干扰信号的频率一般相对较高，采用低通滤波器来滤除干扰是最容易想到的办法[23]，一阶低通滤波的算法公式为：

(3-1)

式(3-1)中：*c*为滤波系数；*q(n)*为本次采样值；*p(n-1)*为上次滤波输出值；*p(n)*为本次滤波输出值[24]，滤波系数*c*需根据系统频域分析得出，一般取值范围为0.5-0-8。

## 3.5升降装置设计

卷扬机作为载重无人机和移动抓钩的连接，需要实时控制，对相应速度有极高要求，为了满足卷扬机的速度要求，可以采用高效的电机和传动装置，同时结合PID控制算法进行精准控制。在卷扬装置的设计中，还需要考虑载重无人机和移动抓钩的重量和尺寸等因素，以确保卷扬机的承载能力和稳定性。此外，卷扬机的安全性和可靠性也是设计中需要重点考虑的因素，可以采用多重保护措施，如过载保护、断电保护等，确保系统的安全运行。

卷扬机构设计需要考虑起重绳索的长度，同时，可以结合传感器和智能控制算法，实现对卷扬机的自适应控制和优化，提高系统的性能和可靠性，考虑到低成本需求，采用大功率带有编码器的有刷直流电机。编码器又称作码盘，是目前广泛使用的测速测位移传感器[16]，它可以通过检测旋转轴的位置和方向，输出相应的脉冲信号，用于控制和调节系统。所用的编码器为电磁编码器，电磁编码器主要由磁阻传感器（霍尔原件）、充磁磁鼓（码盘）、信号处理电路组成。其工作原理是利用磁阻或霍尔元件检测角度和位移的变化[17]。磁盘上有磁极交替排列而成的磁场，当电机带动磁盘旋转时，两个霍尔元件会感应到磁场的变化并产生电信号输出。这两个电信号相位差90度，可以用于确定物体的位置和运动状态，输出信号如图3-6所示，使用stm32定时器的编码模式可读取输出脉冲数，用于表示电机旋转的角度。

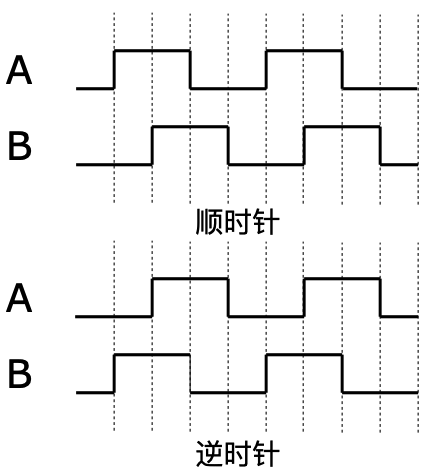
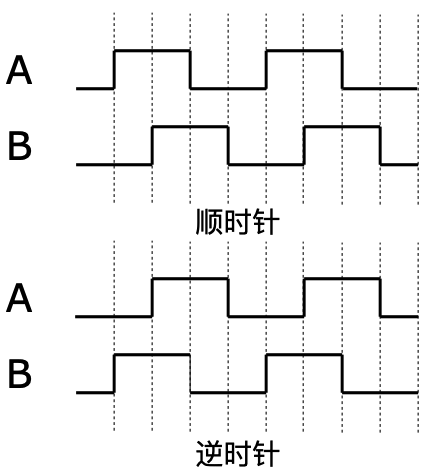


图3‑6 编码器信号输出

## 3.6抓钩移动设计

抓钩移动的控制是本软件系统的核心，包含多个模块的协同控制，为满足精度需求，使用多种控制算法进行融合控制。

### 3.6.1 单片机与飞行控制器通信设计

PWM是一种常用于无人机接收器无线电频率进行与遥控器的通信，本设计使用PWM通信方式的飞行控制器和接收器。在PWM波形中，通过遥控器改变信号的脉冲宽度来控制无人机的各个通道，每个通道对应一个控制量（如油门、方向、俯仰和横滚等）。PWM通信方式的波形通常具有如下特点：

(1)脉冲周期固定，通常为50Hz。

(2)脉冲宽度在0.5毫秒至2.5毫秒之间，通常对应控制量的最小值和最大值。

(3)每个通道的脉冲宽度与控制量成正比。

结合设计需求，PWM波形的方式满足无人机控制的基本需求，能够实现基本的运动（升降、旋转、俯仰和副翼），且PWM方式控制简单，通过单片机定时器功能和I/O即可输出信号，每个通道对应一个功能，抗干扰能力强。

通过实际测试得出如图3-7所示的飞控的PWM控制信号，其频率为47Hz（21.2766ms），脉宽范围为0.811ms-2.02ms，使用4通道PWM接口控制。stm32单片机具有PWM输出功能，通过使用内部定时器产生PWM。

在stm32f103中，定时器时钟频率为72MHz，通过公式(3-1)

计算可将定时器预分频器（Prescaler）设置为72-1，自动重装载寄存器（Auto-Reload Register）设置为21277，定时器将输出47Hz的PWM脉冲。具体配置如图3-8所示。

图3-7 PWM控制脉宽

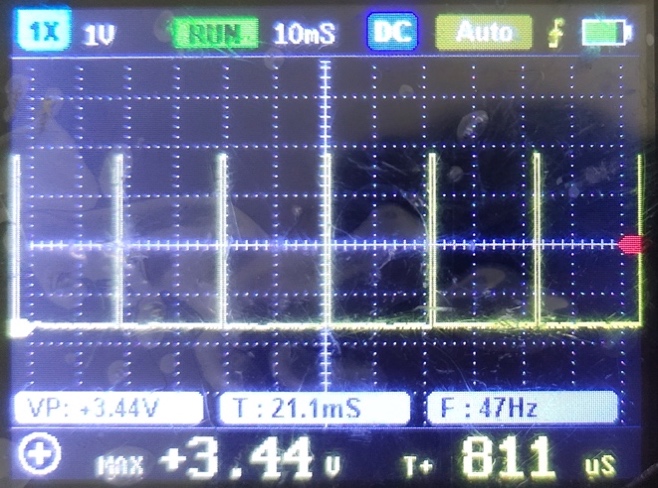
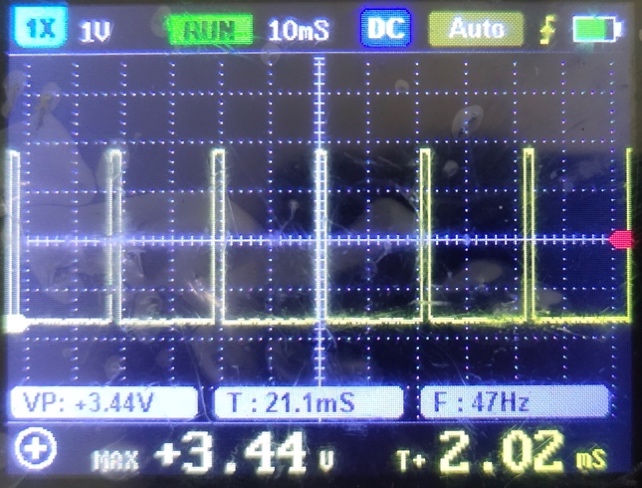




图3-8 定时器配置

### 3.2.3 抓钩移动的实现

该系统使用分布式控制系统，系统各模块分布情况如图3-9所示。为了实现抓钩智能移动的功能，需要使用多个模块的数据，并对这些数据进行计算和处理。基于传感器模块提供的数据，进行计算和处理以实现智能移动。具体的计算步骤将取决于所需的功能和算法。通过不断监测和更新传感器数据，实时调整控制指令，以实现闭环反馈控制，确保抓钩的准确移动和操作。

抓钩智能移动的控制核心是位置式PID控制，其控制表达式为：

(3-2)

使用C语言编写PID控制算法时需要对表达式进行离散化处理,将该公式转换成C语言可表示的公式，可得到：

其中Err为当前系统误差，SumErr为积分项，LastErr为上次系统误差，K为系统增益系数，OpenMV采集的坐标点作为控制系统的输入量，PWM信号脉宽作为输出量，在控制过程中，比例控制量、积分控制量和微分控制量的权重可以根据实际需求进行调整和优化，以达到更好的控制效果。同时，该设计还采用了卡尔曼滤波算法，对目标物体进行识别和跟踪，增加系统的可靠性，提高了系统的智能化水平。该方案的实现可以为载重无人机的应用提供一种可行的智能移动抓钩软件设计方案。

图示

描述已自动生成

图3‑9 系统模块分布

## 3.7抓取装置设计

抓取功能是系统运行的最后一步，该装置使用可正反转的直流电机进行控制。超声波模块循环检测抓钩高度，模块使用HC-SR04，其测距原理为超声波发生器发出一个超声波信号，当这个超声波信号遇到被测物体反射回来后，就会被超声波接收器接收到，模块工作时序如图3-10所示。

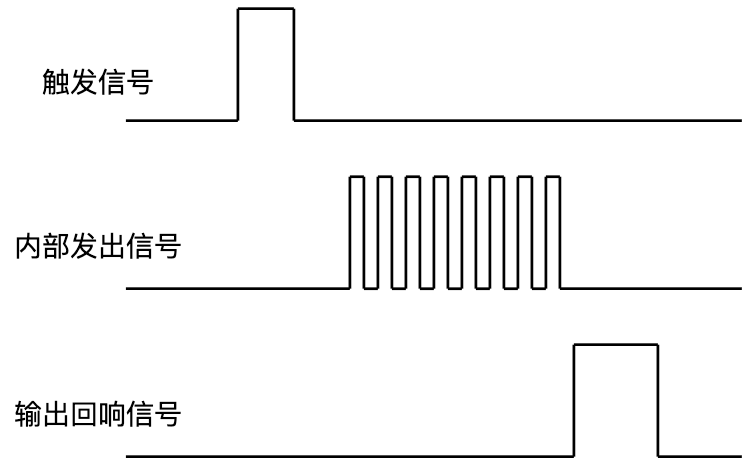


图3-10 超声波工作时序图

模块只需要测量从发送器发送开始，到接收器接收到的时间间隔即可计算出目标距离。采用IO口TRIG触发测距，给至少10us的高电平信号，便会触发检测功能，随后ECHO返回一个脉冲，其宽度与检测距离成正比，通过脉宽即可检测出距离[25]。当检测距离（高度）小于5cm时抓钩向下移动到合适位置，电机旋转触发抓钩抓取功能，完成货物的抓取。

## 3.8本章小结

本章介绍了移动抓钩软件系统的具体工作流程，

# 第四章 软件编写与调试

## 4.1 基于OpenMV IDE的视觉识别代码实现

### 4.1.1 颜色阈值设定

通过调用sensor.set\_pixformat(sensor.RGB565) 语句设置相机模块输出的像素格式为RGB565，即每个像素用16位表示，调用sensor.set\_framesize(sensor.QVGA) 语句设置相机模块输出的帧大小为QVGA(320x240)。在OpenMV IDE中可使用阈值编辑器对所识别颜色进行阈值设定，识别特定颜色块。如图4-1所示，设置识别浅黄色颜色块的阈值设定。



图4‑1 颜色块阈值设定

### 4.1.2 中心坐标采集与传输

颜色块中心坐标使用blobs = img.find\_blobs([threshold\_01])和max\_b = find\_max(blobs)得到，cx=max\_b[5]、cy=max\_b[6]、 cw=max\_b[2]、ch=max\_b[3]，cx表示目标颜色色块的中心点的 x 坐标，Cy表示目标颜色色块的中心点的 y 坐标，cw表示目标颜色色块的宽度，ch表示目标颜色色块的高度。如图4-2所示为实时识别颜色块坐标信息。

坐标数据可通过UART/USART串口进行传输，可将串口波特率设置为115200，8位数据位，无校验，1位停止位，并以特定周期向单片机发送坐标数据。



图4‑2 识别颜色块坐标点

## 4.2 基于STM32CubeMX的控制代码实现与调试

### 4.2.1 系统初始化配置

根据stm32特性，选用8MHz晶振作为系统时钟源，产生稳定的振荡信号来驱动系统的时钟，通过如图4-3所示的配置，将系统时钟配置为72MHz。

图示

描述已自动生成

图4‑3 系统时钟配置

### 4.2.2 系统仿真

在Keil MDK中，"Watch"（监视）是一个调试工具，可用于查看和监视变量、寄存器和表达式的值。使用Watch功能，可以在程序执行过程中实时查看这些值，以帮助调试和分析代码。

在程序运行过程中，可以实时查看OpenMV采集的目标点信息，如图4‑4所示，采集到的坐标信息为X=278,Y=84。

表格

描述已自动生成

图4‑ 接收OpenMV坐标

## 4.3本章小结

# 第五章 总结

## 5.1 总结

本文介绍了一种基于嵌入式系统控制的智能移动抓钩的载重无人机系统，该系统包括多个模块，如四旋翼无人机、控制器、OpenMV视觉识别模块、超声波测距模块、升降装置、载重抓钩、IMU和自稳云台等。这些模块共同协助载重无人机自动装卸货物，使得载重无人机更加智能。本文从视觉识别入手，以视觉识别目标位置为软件系统的输入量，最终输出到四旋翼无人机，控制无人机自动对接和自动抓放。各模块通过不同的通信方式进行数据传输和控制指令的发送，需要进行相应的软件设计和优化，以保证系统的性能和稳定性。同时，本文也介绍了系统的不足之处，如仅使用单目视觉识别方法、系统精度有待提高、仅使用简单的PID控制算法和卡尔曼滤波算法等。因此，该系统还有进一步优化和完善的空间。

## 5.2 展望

智能移动抓钩是一种基于机器人技术的智能装备，其应用不限于无人机，在农业和工业领域具有巨大的应用空间，如工业生产、物流配送、仓储管理等领域得到广泛应用，但是在无人机的应用中，由于没有固定着力点，抓钩应具有更高的稳定性和鲁棒性。为了解决这个问题，可以考虑引入机器学习算法进行优化。通过对无人机飞行数据和抓钩状态数据的收集和分析，可以训练出一个机器学习模型，用于预测无人机在不同飞行状态下的抓钩稳定性和鲁棒性。同时，可以通过实时监测无人机和抓钩的状态，对机器学习模型进行实时更新和优化，以保证系统的性能和稳定性。

另外，为了进一步提高系统的智能化水平，可以考虑引入深度学习算法进行目标识别和跟踪。通过对大量图像数据的训练，可以训练出一个深度学习模型，用于实现无人机对目标的自动识别和跟踪。这样可以大大提高系统的自动化程度和操作效率，同时也可以减少人为操作的风险和误差。

综上所述，智能移动抓钩的应用具有广泛的应用前景，但是在实际应用中需要考虑多种因素，包括通信协议、软件设计、机器学习算法和深度学习算法等。只有综合考虑这些因素，才能实现一个高性能、高稳定性和高智能化的智能移动抓钩系统。

# 参考文献

1. 姜运宇. 旋翼无人机跟踪地面移动目标的视觉控制[D].哈尔滨工业大学,2014.
2. 薛菁,李帅,代妮娜,雷国平,张望.无人机精准降落研究及应用发展综述[J].现代信息科技,2021,5(15):134-136.
3. 靖明阳. 面向多旋翼无人机的智能抓捕与收纳机器人研究[D].西安电子科技大学,2020.
4. 李勤文. 基于视觉反馈的自动对准系统关键技术研究[D].中国科学院大学(中国科学院长春光学精密机械与物理研究所),2022.
5. 吴文秀,关志全,汪磊,郑军斌,阳璞琼.基于openmv的无人机定点降落视觉识别技术研究[J].智能计算机与应用,2021,11(01):41-45.
6. Gupta K, Emran B J, Najjaran H. Vision-based pose estimation of a multi-rotor unmanned aerial vehicle[J]. International Journal of Intelligent Unmanned Systems, 2019, 7(3): 120-132.
7. 许海洋. 四旋翼无人机飞行姿态控制方法研究[D].西南科技大学,2021.
8. Welch G, Bishop G. An introduction to the Kalman filter[J]. 1995.
9. 彭丁聪. 卡尔曼滤波的基本原理及应用[J]. 软件导刊,2009,8(11):32-34.
10. 邱润,黎敬涛,李孝疆,杨改娣,宋开雨.基于卡尔曼滤波算法的轨迹预测[J].电视技术,2022,46(06):24-28.
11. 黄克亚,余雷,李晓旭.STM32与PC机USART通信教学实验设计[J].现代电子技术,2022,45(10):21-25.
12. 刘钱强,钟小勇.基于OpenMV四旋翼无人机视觉跟踪系统设计[J].自动化与仪表,2023,38(05):5-8+40.
13. 姚志强,赵恩铭,杨松,杨燕婷,刘光宇,周豹,王辰,朱晓栋.基于OpenMV的目标方位追踪算法[J].牡丹江师范学院学报(自然科学版),2021(03):14-18.
14. 陈孟臻. 三轴WiFi摄像头舵机云台的设计与实现[D].广西大学,2015.
15. 刘凯峰. 多旋翼无人机姿态解算的研究[D].西安建筑科技大学,2019.
16. 赵晋秀,刘文杰.全向底盘机器人智能定位和姿态检测系统——基于正交编码器和陀螺仪[J].工业技术创新,2020,07(05):33-37.
17. 赵映川.编码器原理与应用分析[J].无线互联科技,2018,15(22):67-69+78.
18. 申庆华.基于STM32的高精度PWM信号发生器的设计[J].电子测试,2021(17):25-26+52.
19. 陈星光. 工业控制中嵌入式控制系统的应用研究[J]. 电子技术与软件工程,2023,(06):79-82.
20. 覃颖. 嵌入式控制系统在工业控制中的应用[J]. 电脑知识与技术,2010,6(05):1227-1228.
21. 徐晨俊琦. 微型无人直升机嵌入式控制系统设计[D].南京航空航天大学,2021.
22. 龚茂林,张凤. 一种基于STM32的PID直流电机控制系统[J]. 电子产品世界,2022,29(02):33-35.
23. 许正望,张小华. 一种动态低通滤波器[J]. 自动化与仪器仪表,2007,(04):51-53.
24. 张小明,王坤坤,于纪言. 旋转倒立摆串级PID低通滤波控制系统的设计[J]. 现代电子技术,2020,43(04):21-24.
25. 苏琳. 基于HC-SR04的超声波测距器的设计[J]. 科技信息,2012,(09):125+124.

# 致谢

四年大学生涯匆匆结束，在此，我要向所有在我完成毕业设计期间给予帮助和支持的老师和同学表示最真诚的感谢。

首先，我要感谢我的指导老师杨雄老师。在我完成毕业设计的过程中，他给予了我无私的指导和帮助，在学术上给予了我很多启发和建议，没有他的支持和鼓励，我无法完成这项毕业设计。

其次，我要感谢我的家人和朋友。他们在我完成毕业设计的过程中一直支持和鼓励我，给予我精神上的支持和帮助。他们的陪伴和鼓励让我感到无比温暖和幸福，让我坚持到毕业设计结束。

此外，我还要感谢实验室的学长和学姐。他们是我完成毕业设计重要的资源，给我提供经验和建议，他们在我完成实验和数据处理的过程中给予了我很多帮助和支持，让我能够更加顺利地完成毕业设计。

最后，我要感谢学校和学院的领导和老师们。他们为我提供了良好的学习和研究环境，让我能够充分发挥自己的才能和潜力。毕业设计的完成也标志着大学生活的结束，在社会上不管扮演什么角色，都需要努力和学习，学无止境，祝大家身体健康，工作顺利。

# 附录

1. 软件主程序代码
2. OpenMV识别代码
3. 核心算法代码

附录一：软件主程序代码

//定义姿态角结构体

static atk\_ms901m\_attitude\_data\_t attitude\_dat;/\* 姿态角数据 \*/

//主函数入口

void mymain()

{

int OpenMV\_X,OpenMV\_Y; //定义OpenMV坐标变量

bool Dir; //定义编码器方向变量

uint16\_t Encode; //定义编码器计数变量

uint16\_t startHigh,High; //定义超声波高度变量

//开启定时器TIM3 PWM功能（控制舵机X，Y）默认设置PWM频率50Hz(定时器计数值20000)

HAL\_TIM\_PWM\_Start(&htim3, TIM\_CHANNEL\_1);

HAL\_TIM\_PWM\_Start(&htim3, TIM\_CHANNEL\_2);

//初始化将云台置于水平

sg90\_XY\_Control(0.0f,0.0f);

//开启定时器TIM4 PWM功能（控制飞控）默认设置PWM频率50Hz(定时器计数值20000)

HAL\_TIM\_PWM\_Start(&htim4, TIM\_CHANNEL\_1);//偏航

HAL\_TIM\_PWM\_Start(&htim4, TIM\_CHANNEL\_2);//油门

HAL\_TIM\_PWM\_Start(&htim4, TIM\_CHANNEL\_3);//俯仰-x

HAL\_TIM\_PWM\_Start(&htim4, TIM\_CHANNEL\_4);//横滚-y

//初始化使无人机处于静止状态

UAV\_Control(0,0,0,0);

//开启串口接收中断，用于读取IMU数据

HAL\_UART\_Receive\_IT(&huart2, &g\_uart\_rx\_fifo.usart\_receive, 1);

//开启串口接收中断，用于接收OpenMV发送的坐标

HAL\_UART\_Receive\_IT(&huart3, &openmv\_rxType.rece, 1);

//开启定时器2编码器模式,用于卷扬装置

HAL\_TIM\_Encoder\_Start(&htim2,TIM\_CHANNEL\_1 | TIM\_CHANNEL\_2);

//开启电机控制PWM

HAL\_TIM\_PWM\_Start(&htim1,TIM\_CHANNEL\_1);

TIM1->CCR1=0;

//初始化IMU

atk\_ms901m\_init();

while(1)

{

//获取姿态角数据，最大等待时间：100ms

atk\_ms901m\_get\_attitude(&attitude\_dat, 100);

//控制云台舵机

sg90\_XY\_Control(attitude\_dat.pitch,attitude\_dat.roll);

//获取目标点相对坐标

OpenMV\_GetXY(&OpenMV\_X,&OpenMV\_Y);

//卡尔曼滤波器

OpenMV\_X=kalmanFilter(&KFP\_TypeX,OpenMV\_X);

OpenMV\_Y=kalmanFilter(&KFP\_TypeY,OpenMV\_Y);

//PID控制

int32\_t Xout=pid\_ctr(&High\_PID,OpenMV\_X,2,0.4,1);//X控制

int32\_t Yout=pid\_ctr(&High\_PID,OpenMV\_Y,2,0.4,1);//Y控制

//输出限幅

if(Xout>2000) Xout=2000;

else if(Xout<1000) Xout=1000;

if(Yout>2000) Yout=2000;

else if(Yout<1000) Yout=1000;

//输出到飞控

UAV\_Control(0,0,Xout,Yout);

//获取超声波高度

High=GetHigh();

High\_PID.SetPoint=startHigh-High;

//获取编码器值

GerEncode(&Dir,&Encode);

High\_PID\_Con(Encode);

HAL\_Delay(10); //延时10ms

}

}

//串口接收中断回调函数

void HAL\_UART\_RxCpltCallback(UART\_HandleTypeDef\* huart)

{

//IMU接收中断

if(huart->Instance==USART2)

{

ATK\_MS901M\_UART\_IRQHandler();

}

//OpenMV接收中断

if(huart->Instance==USART3)

{

OpenMV\_UsartHandler();

}

}

附录二：OpenMV识别代码

# 导入了一些模块，这些模块提供了相机模块，图像处理，时间，数学和外围设备的功能。

import sensor, image, time,math,pyb

from pyb import UART

#导入json模块，用于处理JSON格式的数据。

import json

#导入ustruct模块，用于将Python数据结构打包成C语言的结构体，以便在串行通信中进行传输。

import ustruct

sensor.reset() #重置相机模块的设置，将其恢复到默认值。

sensor.set\_pixformat(sensor.RGB565) #设置相机模块输出的像素格式为RGB565，即每个像素用16位表示。

sensor.set\_framesize(sensor.QVGA) #设置相机模块输出的帧大小为QVGA(320x240)。

sensor.skip\_frames(time = 2000) #跳过2秒钟的帧以便相机模块稳定。

sensor.set\_auto\_gain(False) #禁用自动增益，这是颜色跟踪所必需的。

sensor.set\_auto\_whitebal(False) #禁用自动白平衡，这是颜色跟踪所必需的。

#设置颜色阈值

red\_threshold\_01=(49, 98, 2, -69, 67, 6)

clock = time.clock()

uart = UART(3,115200) #定义串口3变量

uart.init(115200, bits=8, parity=None, stop=1) # init with given parameters

def find\_max(blobs): #定义寻找色块面积最大的函数

max\_size=0

max\_blob=0

for blob in blobs:

if blob.pixels() > max\_size:

max\_blob=blob

max\_size = blob.pixels()

return max\_blob

def sending\_data(cx,cy,cw,ch):

global uart;

#frame=[0x2C,18,cx%0xff,int(cx/0xff),cy%0xff,int(cy/0xff),0x5B];

#data = bytearray(frame)

data = ustruct.pack("<bbhhhhb", #格式为俩个字符俩个短整型(2字节)

0x2C, #帧头1

0x12, #帧头2

int(cx), # up sample by 4 #数据1

int(cy), # up sample by 4 #数据2

int(cw), # up sample by 4 #数据1

int(ch), # up sample by 4 #数据2

0x5B)

uart.write(data); #必须要传入一个字节数组

while(True):

clock.tick()

img = sensor.snapshot()

blobs = img.find\_blobs([red\_threshold\_01])

max\_b = find\_max(blobs)

cx=0;cy=0;

if blobs:

#如果找到了目标颜色

cx=max\_b[5]

cy=max\_b[6]

cw=max\_b[2]

ch=max\_b[3]

img.draw\_rectangle(max\_b[0:4]) # rect

img.draw\_cross(max\_b[5], max\_b[6]) #在图像上绘制一个十字形标记

FH = bytearray([0x2C,0x12,cx,cy,cw,ch,0x5B])

uart.write(FH)

time.sleep\_ms(10)

else:

FH = bytearray([0x2C,0x12,0,0,0,0,0x5B])

uart.write(FH)

time.sleep\_ms(10)

附录三：核心算法代码

KFP\_Structure KFP\_TypeX = {0, 0, 0, 0, 0.001, 0.543};

KFP\_Structure KFP\_TypeY = {0, 0, 0, 0, 0.001, 0.543};

/\*\*

\*\* @param kfp: [输入/出] 卡尔曼滤波器结构体指针

\*\* input: [输入] 待滤波的数据

\*/

float kalmanFilter(KFP\_Structure\* kfp, float input)

{

//预测协方差方程：k时刻系统估算协方差 = k-1时刻的系统协方差 + 过程噪声协方差

kfp->Now\_P = kfp->LastP + kfp->Q;

//卡尔曼增益方程：卡尔曼增益 = k时刻系统估算协方差 / （k时刻系统估算协方差 + 观测噪声协方差）

kfp->Kg = kfp->Now\_P / (kfp->Now\_P + kfp->R);

//更新最优值方程：k时刻状态变量的最优值 = 状态变量的预测值 + 卡尔曼增益 \* （测量值 - 状态变量的预测值）

kfp->out = kfp->out + kfp->Kg \* (input - kfp->out); //因为这一次的预测值就是上一次的输出值

//更新协方差方程: 本次的系统协方差付给 kfp->LastP 威下一次运算准备。

kfp->LastP = (1 - kfp->Kg) \* kfp->Now\_P;

return kfp->out;

}

//\* 使用一阶低通滤波器对单个数据进行多次滤波

//\* @param alpha 滤波器时间常数

float lowpass\_filter(float x, float alpha)

{

static float y = 0; // 上一次滤波的输出值

float filtered\_data = alpha \* y + (1 - alpha) \* x;

y = filtered\_data; // 更新状态变量

return filtered\_data;

}

PID控制代码

PID\_TypeDef High\_PID;//高度PID控制

PID\_TypeDef X\_PID;//对准PID控制

PID\_TypeDef Y\_PID;//对准PID控制

void pid\_init(void)

{

//PID控制初始化

}

float pid\_ctr(PID\_TypeDef\* PID, float Feedback\_value,float kp,float ki,float kd)

{

PID->Error = PID->SetPoint - Feedback\_value; /\* 计算偏差 \*/

PID->SumError += PID->Error;

if(PID->SumError > 500) /\* 积分限幅 \*/

{

PID->SumError = 500;

}

else if(PID->SumError < -500)

{

PID->SumError = -500;

}

PID->ActualValue = kp \* PID->Error /\* 比例环节 \*/

+ ki \* PID->SumError /\* 积分环节 \*/

+ kd \* (PID->Error - PID->LastError); /\* 微分环节 \*/

PID->LastError = PID->Error;

return ((float)PID->ActualValue); /\* 返回计算后输出的数值 \*/

}