2019 年全国大学生电子设计竞赛

巡线机器人（B 题）

2019 年 8 月 10 日

赛区编号：

学校姓名：河北科技大学

队长姓名： 曹晓东

队员姓名： 张成 、 李圭印

指导教师姓名： 王彦朋 、 马金龙

2019 年 8 月 10 日

摘　要

通过对该系统结构与特点的分析，结合现代控制技术设计理念实现了以微控制器 TM4C123GH6PM 单片机为核心的基于四旋翼飞行器的寻线机器人。系统运行分为三个部分：起飞、自动巡检与降落。起飞与降落使用了 LC306 光流传感器以及 tfmini plus 激光测距模块达到定点升降。巡检的工作过程为：全程开启定高利用侧面放置的摄像头进行寻线以及查找线缆上的异物，当查找到异物后开启定点进行拍照，拍照后继续寻线直到查找到二维码后再次开启定点对二维码进行拍摄存储，接着继续寻线返回起飞点。控制器采用 PID 算法并结合自抗扰控制以提高系统的鲁棒性。

关键词：TM4C123GH6PM 单片机 自动巡检 摄像头 自抗扰控制 PID

一、前言

设计一个可巡线的四旋翼飞行器，能够巡检电缆及杆塔状态（二维码识别），发现异常时拍摄存储（条形码识别），任务结束传送到地面显示装置上显示，全程利用激光标定航迹。除此之外，要能够悬挂 100g 的重物进行悬停。

利用光流传感器和激光测距传感器来实现四旋翼的定高定点，保证其飞行过程中的稳定性。利用 OPENMV 来识别条形码和二维码并进行拍照存入 SD 卡中，以便在显示装置（STM32F103RBT6 核心板制作）中显示，同时以电缆为基准进行巡线，在巡检过程中若发现黄色凸起物，OPENMV 闪烁红灯以进行提示。飞行过程中的控制系统采用 PID+前馈+自抗扰控制，从而增强四旋翼的鲁棒性，飞行任务结束后，利用 OLED 来实时显示飞行信息以及记录黄色凸起物的位置信息。

二、系统方案论证

1.巡线传感器方案比较与选择

（1）光电传感器：一般由处理通路和处理元件 2 部分组成。其基本原理是以光电效应为基础，把被测量的变化转换成光信号的变化，然后借助光电元件进一步将非电信号转换成电信号。用光电传感器去检测电缆和杆塔，此办法虽然方便编程，但其灵敏度不够高，当四旋翼以较快的飞行速度偏离电缆时，光电传感器不能及时感应到，导致四旋翼不能够快速纠正航线，从而偏离航线。

（2）OpenMV：OpenMV 以 STM32F427CPU 为核心，集成了 OV7725 摄像头芯片，在小巧的硬件模块上，用 C 语言高效地实现了核心机器视觉算法，提供 Python 编程接口,能够很快捷简便的使用 OpenMV 提供的机器视觉功能。通过 OpenMV,可以实时获取四旋翼偏离航线的距离，从而及时纠正航迹，使其稳定巡线。

光电传感器虽然逻辑简单，编程方便，但灵敏度不够高。而 OpenMv 虽然内部代码复杂，但可用 Python 编程，提供了很方便的编程接口，并且处理速度以及灵敏度都很好，要比光电传感器更适合巡线。 因此 OpenMv 优于光电传感器。

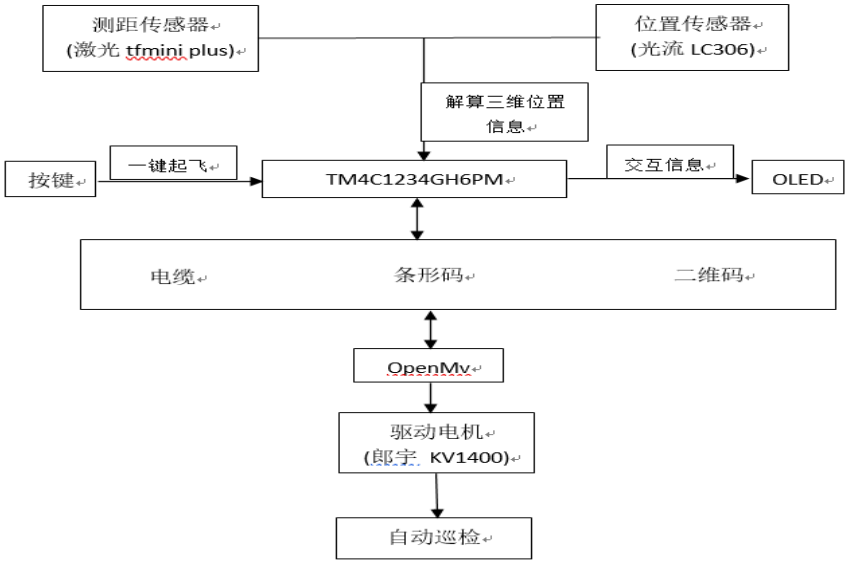
2.巡线方式方案比较与选择

（1）上方巡线：四旋翼在电缆上方执行任务，其优点是 OpenMv 背景单一简单，便于巡检电缆和检测凸起物，但由于塔 B 上的二维码位于塔 B 的侧方，镜头朝向无法移动。若解决此问题需添加云台结构，保持 OpenMv 的稳定性以及可旋转性，但这样做会大大增加程序的编写难度以及占据机体过大空间。

（2）侧方巡线：将 OpenMv 的镜头朝向机体左侧，进行侧方巡线，该方法即避免的镜头朝向的改变，又能够正常巡迹。虽然背景不可控，但可通过背景虚化算法对无用背景进行滤波，从而稳定识别电缆、条形码和二维码。

侧方巡线相比上方巡线，在取消云台结构的同时，能够检测到二维码，能够更易于程序的编写以及各个传感器在机体上的分布设计，因此侧方巡线方案优于上方巡线方案。

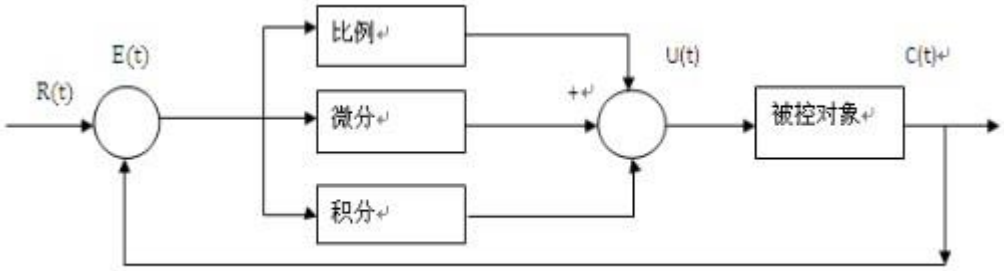
综上所述，我们最终选择的方案是利用 OPENMV 进行侧方寻线。系统框图如下：



三、理论分析与计算

1.控制算法

连续 PID 控制器也称比例—积分—微分指控器，即过程控制是按误差的比例、积分和微分对系统进行控制。



PID 算法的控制数学模型为：

$$u(t)=k_p[e(t)+\frac{1}{T_i}\int_0^t e(t)dt+T_d\frac{de(t)}{dt}] \text{ 即 } G(s)=\frac{U(s)}{E(s)}=k_p(1+\frac{1}{T_i s}+T_d S)$$

2.控制器的设计与仿真

四旋翼的控制是通过调节四个旋翼的转速实现的。一般通过两个控制回路对四旋翼实现控制。一个是内环，用于控制四旋翼的姿态；一个是外环，用于控制四旋翼的位置。

整体仿真框图如图 1 所示。被控对象四旋翼的模型由.m 文件编程实现，输

入为四个电机的转速 w1、w2、w3、w4，输出为 3 个位置加速度和 3 个角加速度。Rotor 模拟电机，输入为 4 个转速给定信号输出为四个转速。

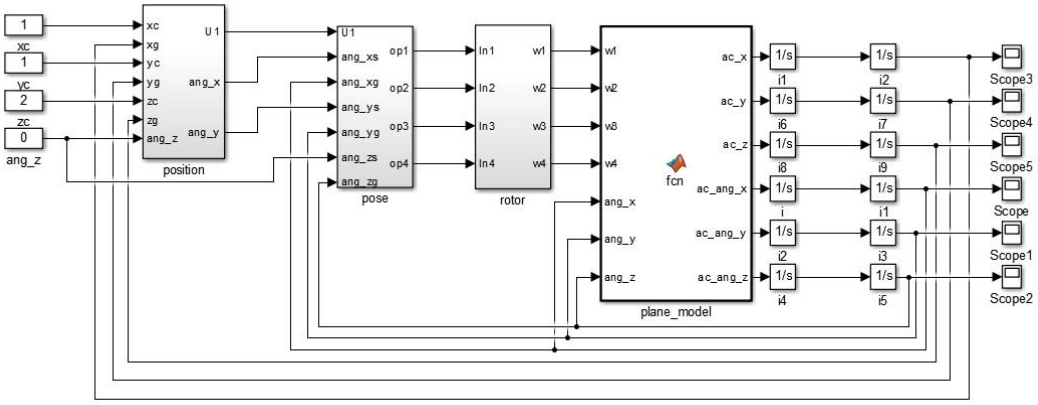


图 1 整体仿真框图

通过观察 XYZ 位置响应曲线以及偏航、俯仰、滚转响应曲线，我们经过仿真调参最终都达到了稳定、静差小，同时，响应也较为迅速。

四、电路与程序设计

1. 电路设计

（1）核心 MCU 电路如图 1

此次设计我们采用 TM4C123FH6PM 单片机，它的优势在于能够方便的运用多种 ARM 的开发工具和片上系统(SoC) 的底层 IP 应用方案，以及广大的用户群体。另外，该微控制器使用了兼容 ARM 的 Thumb®指令集的 Thumb2 指令集来减少存储容量的需求，并以此达到降低成本的目的。

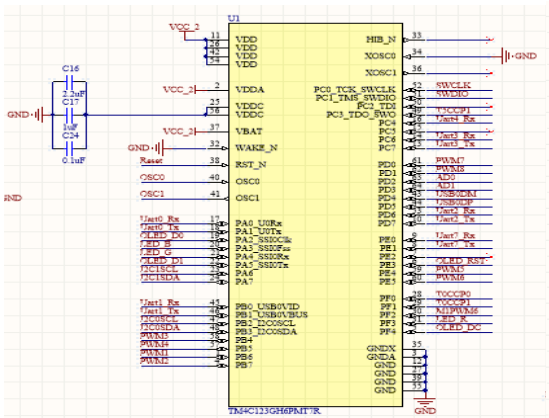


图 1

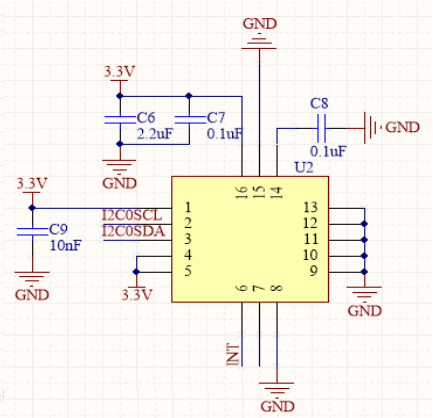


图 2

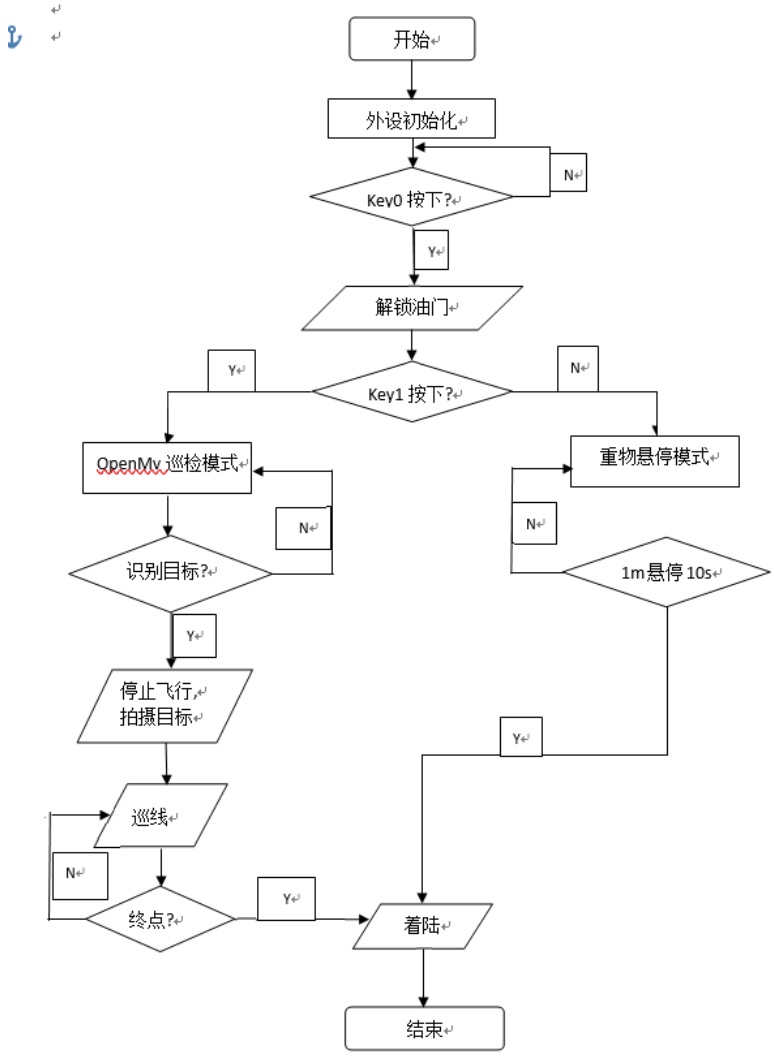
（2）IMU 模块电路如图 2

2.程序设计

（1）设计工具及原理

程序设计工具采用 KEIL MDK5 编译器，匿名地面站、山外地面站等调试软件；程序设计采用了卡尔曼、巴特沃斯、一阶互补滤波、PID+前馈+自抗扰控制、陀螺仪姿态解算等算法。

（2）程序框图如下：



五、系统测试

1.测试使用的仪器设备

ARM 仿真器、万用表、示波器、锂电池充电器、调试架

2.系统测试

第一:根据规则要求搭建实验平台，制作出规则要求的环形圆板并在该板上进行配重悬停实验。

第二:在加上 100g 重物的情况下，去整定高度加速度环 PID 参数观察其各参数对悬停效果以及抗干扰能力的影响

表 1　测试高度加速度环对悬停的影响

序号	PID 参数	高度波动范围（cm）	波动中心与目标高度（1m）的距离(cm)	飞行至目标高度的时间（s）
1	P=500，I=0,D=1000	19	15	2.5
2	P=700，I=0,D=1000	25	13	1.8
3	P=500，I=0,D=2000	10	12	2.4
4	P=500，I=500,D=1000	27	4	2.1

3.测试结果分析

由以上实验发现：

（1）高度加速度环 P 值影响起飞的速度，越大越快，但会降低悬停稳定性；

（2）I 值越大，与目标间的静差越小，但也会引起震荡降低悬停稳定性；

（3）D 值有抑制震荡的作用，适当提高可增强稳定性；

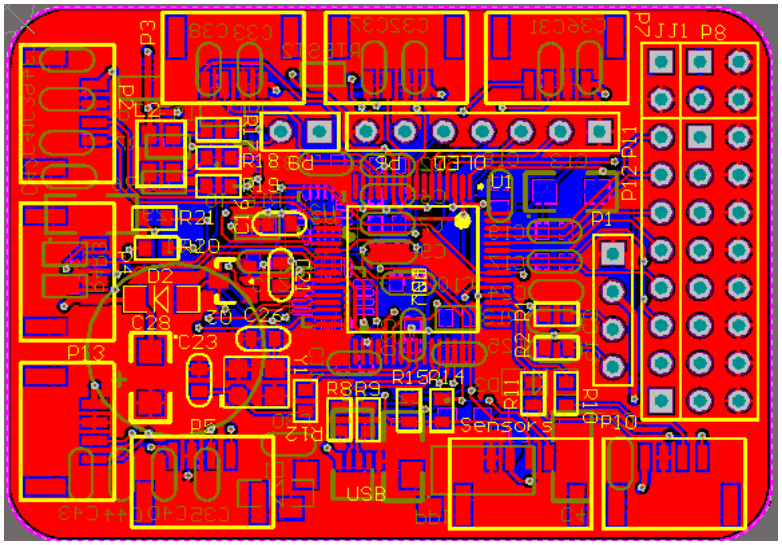
经过以上实验，我们发现通过整定高度加速度环 PID 参数可以让系统的响应有所提高，但是还不够稳定，所以我们又加入了前馈控制以提高系统的响应以及抗干扰能力。

在实验过程中，我们还发现飞行器的定高和定点经常会出现异常，后来发现是传感器数据异常，导致这种情况的原因来自于光照对于光流传感器的影响以及地面平滑度对激光传感器的影响。所以在后续实验都选择在光照好、地面有纹理的地面上进行。

六、结束语

通过本次巡线机器人的设计与制作，实现了以微控制器 TM4C123GH6PM 单片机为核心的基于四旋翼飞行器的寻线机器人的目标。我们团队既有失败后收获的教训，也有取得测试成功的经验。从开始到结束，经过一步步的摸索、测试、摔倒、再测试，经过无数次的失败和调整以及这些日子里队友们夜以继日的奋力执行和操作，我们在一步步地接近目标。正如古人语：交得其道，千里同好，固于胶漆，坚于金石。我们之所以取得收获，离不开我们团队的团结合作，各取所长，合作共赢。当然，我们也会总结成功的经验和失败的教训，进一步改进、提升。

附　录



附录一　主控板 PCB 图

附录二　主函数：

```
#include "Headfile.h"
int main(void)
{
  HardWave_Init();//芯片资源、飞控外设初始化
  while(1)//主循环
  {
    Get_Battery_Voltage();//测量电池电压
    Key_Scan(Key_Right_Release);// 按键扫描
    QuadShow();//OLED 显示
    Vcan_Send();//山外地面站（多功能调试助手）
    ANO_SEND_StateMachine();//ANO 地面站发送
    Accel_Calibartion();//加速度计 6 面校准

Mag_Calibartion_LS(&WP_Sensor.mag_raw,Circle_Angle);//磁力计椭球校准

RC_Calibration_Check(PPM_Databuf);//遥控器行程校准
    Save_Or_Reset_PID_Parameter();//运用地面站，修改控制参数
  }
}
```

附录三　部分子程序：

```
uint8_t  move_with_target(float  pos_x_target,float  pos_y_target,Duty_Status  *Status,uint8_t  *Start_flag)
{
    //static  Vector2f  pos_base;
    ncq_control_althold();//高度控制依然进行
    if(*Start_flag==1)
    {
        //pos_base.x=OpticalFlow_SINS.Position[_PITCH];
        //pos_base.y=OpticalFlow_SINS.Position[_ROLL];
        *Start_flag=0;
    }

    if(Status->Start_Flag==0)
    {
        OpticalFlow_Pos_Ctrl_Expect.x=OpticalFlow_SINS.Position[_PITCH]+pos_x_target;
        OpticalFlow_Pos_Ctrl_Expect.y=OpticalFlow_SINS.Position[_ROLL]+pos_y_target;
        Status->Start_Flag=1;
    }

    if(Status->Start_Flag==1
         &&Status->Execute_Flag==1
              &&Status->End_flag==1)
    {
        OpticalFlow_Control_Pure(0);//完成之后，进行光流悬停
        return  1;
    }
    else
    {
        if(pythagorous2(OpticalFlow_Pos_Ctrl_Expect.x-OpticalFlow_SINS.Position[_PITCH],
          OpticalFlow_Pos_Ctrl_Expect.y-OpticalFlow_SINS.Position[_ROLL])<=8.0f)
        {
            Status->Execute_Flag=1;
            Status->End_flag=1;
            //pos_base.x=0;
            //pos_base.y=0;
            OpticalFlow_Control_Pure(1);//完成之后，进行光流悬停
            OpticalFlow_Pos_Ctrl_Expect.x=0;
            OpticalFlow_Pos_Ctrl_Expect.y=0;
            return  1;
        }
        else
        {
            Status->Execute_Flag=1;
            OpticalFlow_Pos_Control();//光流位置控制
            OpticalFlow_Vel_Control(OpticalFlow_Pos_Ctrl_Output);//光流速度控制
            return  0;
        }
    }
}
```