基于 Multiwii 的四轴飞行器

张馨元 17307110389

一．课题设计

Multiwii 是基于 Arduino 的开源飞控程序，飞控板上装载了 Arduino 核心主控芯片 ATMEGA328P，MPU6050 姿态传感器，HM-11 蓝牙模块以及 720 空心杯马达的接口和 N-Mos 管。多种元件高度集成在 PCB 板上能让飞行器小巧轻便，同时各个部件的工作原理透明清晰，飞行器不再是塑料壳子里的黑匣子，可以探究单片机与控制算法结合产生的神奇结果。研究内容主要分为两块，一是对硬件工作原理的学习，二是对程序控制语言的学习，并对输出数据进行简要分析。

二．实验过程

1. 硬件:

（1） MOSFET

MOSFET 是 metal-oxide-semiconductor 组成的结构的场效应晶体管，它的功能相当于一个微型开关。如图 1，G，S，D，B 分别是栅极，源极，漏极和衬底，黑色是金属，紫色是氧化物绝缘体，通常为 SiO_2，棕色和灰色为两种不同类型的半导体，源极和漏极是高度掺杂的。当栅极没有电压接通，源极和漏极的耗尽层位置局域在它们各自的周围，当栅极接通电压后（电压大小由 IO 口的 PWM 信号调控），耗尽层在源极和漏极之间延展开来，形成一个电子的导通沟道，从而导电。我们使用的 720 空心杯电机的工作电压是 3.7V，若直接接到 3.7V 电源上，它会不停大马力旋转，是无法控制电机旋转速度的，于是需要在电机和控制电路之间接一个 MOS 管来控制转速。

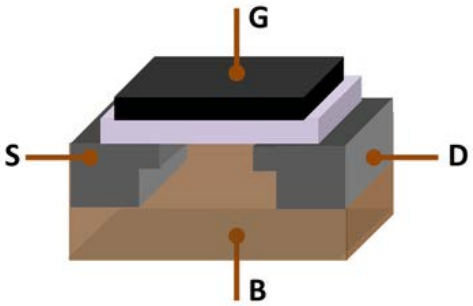


图 1：MOSFET 结构示意图

（2） MPU-6050

MPU-6050 包含 3 轴 MEMS 陀螺仪和 3 轴 MEMS 加速度计，陀螺仪工作原理是通过科里奥利力使微电机系统中电容极板间距发生变化，通过积分从而得到目前的姿态信息，稳定性高但有明显的累积误差。加速度计的测量也是通过电容极板之间距离的变化来测量加速度，重力加速度和三轴之间的夹角可以通过三角运算得到，加速度计得到姿态位置不需要积分就可以直接得到，没有积累误差，但缺点是稳定性差。使用 MPU-6050 测量飞行器姿态的时候需要使用滤波算法融合两组数据。

为了测试 MPU-6050 的测量效果，让程序输出了 MPU-6050 六轴（加速度计三轴+陀螺仪三轴）的原始数据，并根据 datasheet 上传感器的灵敏度取标度数据，得到对应的角度。受到环境和自身电信号的干扰，加速度计和陀螺仪每次输出的数据都会有一定附加的误差。对其进行零偏矫正，在初始化的过程中加入一个循环，在 MPU-6050 水平静止请款该读取零点的数据 200 次，估计误差为 200 次数据的平均值，在之后的数据处理中减去零偏误差。并进行简单滤波观察角度测量值随时间的漂移程度和对震荡的敏感程度，最终角度测量的效果通过 processing 软件进行可视化。

2. 软件:

除了飞控板之外，需要下载 MultiWii 源码压缩包和 Multiwiiconf.exe，这是一个显示和调控各项参数的飞行器界面。

(1) MultiWii 开源程序压缩包

解压压缩包，可以看到里面有许多的文件（图 2），Config 文件中选择无人机类型（四轴），传感器类型（MPU6050），飞控板类型，控制算法类型，IMU 函数计算姿态和高度，Output 文件将控制算法计算值转换为 PWM 波输送给电调控制电机。对飞行器实现 PID 算法控制和转速控制的主程序在 MultiWii.cpp 文件中。

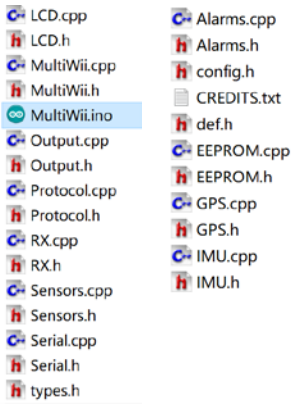


图 2：MultiWii 文件

（2）.PID 控制系统

PID 是一个工业中常用的控制系统，系统持续计算所控制量的当前值（PV）和设定值（SV）之间的误差，并利用比例控制（proportional），积分控制（integral），微分控制（derivative）来控制误差，控制回路如图 3 所示[1]。

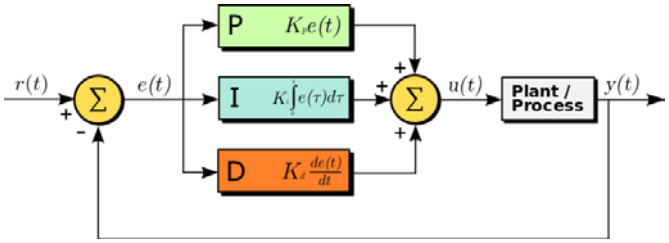


图 3：PID 控制系统流程示意图

根据 SV-PV 得到的误差大小，乘上一个比例因子，得到 P 项的修正，但 P 项只能缩小误差而不能使误差为零，因为误差为零时 P 项修正将不起作用，此时 I 项发挥作用，它是误差的累计值，当误差趋近于零时，积分值几乎不变，此时 I 项修正其主要作用。但是系统仍可能会出现修正过快或过慢的情况，此时 D 项发挥作用，D 项也被形象地称为“先期控制”，因为根据 error(t)项的变化快慢，它可以判断做出修正的大小。

PID 三项的协调过程被称为 loop tuning 。在 Multi Wii 中，PID 是根据角度来调整来调整螺旋桨转速。在程序中对应部分如图 4

[1] https://en.wikipedia.org/wiki/PID_controller

```
  //计算P项
  PTerm = ((int32_t) RateError * conf.pid[axis].P8)>>7;

  //计算I项
  //精度对I项很重要，因为I项可能会产生长时间积分引起的漂移，因此使用了32位积分
  errorGyroI[axis]  += (((int32_t) RateError * cycleTime)>>11) * conf.pid[axis].I8;
  //对I项还需加一个限制，防止累计数据过大，系统饱和
  errorGyroI[axis]  = constrain(errorGyroI[axis], (int32_t) -GYRO_I_MAX<<13, (int32_t) +GYRO_I_MAX<<13);
  ITerm = errorGyroI[axis]>>13;

  //计算D项
  delta          = RateError - lastError[axis];
  lastError[axis] = RateError;
  delta = ((int32_t) delta * ((uint16_t)0xFFFF / (cycleTime>>4)))>>6;
  //数据加和减少噪音影响
  deltaSum       = delta1[axis]+delta2[axis]+delta;
  delta2[axis]   = delta1[axis];
  delta1[axis]   = delta;
  DTerm = (deltaSum*conf.pid[axis].D8)>>8;

  //计算PID总和
  axisPID[axis] =  PTerm + ITerm + DTerm;
}
 mixTable();
// 在对姿态传感器进行校准时，不需要进行PID对电机的控制
if ( (f.ARMED) || ((!calibratingG) && (!calibratingA)) ) writeServos();
writeMotors();

prop2 = 128;
if (rcData[THROTTLE]>1500) {
  if (rcData[THROTTLE]<2000) {
    prop2 -=  ((uint16_t)conf.dynThrPID*(rcData[THROTTLE]-1500)>>9);
  } else {
    prop2 -=  conf.dynThrPID;
  }
}
```

图 4: 基于陀螺仪的俯仰角 pitch 和侧倾角 roll 的 PID 修正计算和对螺旋桨的控制

3. 飞行器安装步骤

a．首先将空心杯电机接线连接到 3.7V 电源上检查其是否能正常旋转，之后用借来的焊接枪，将四个电机的接线分别焊接到飞控板上。

b．螺旋桨分为两种，旋转起来向上吹风的螺旋桨和向下吹风的螺旋桨，相同类型螺旋桨安装在四轴飞控板的对角线上（图 5）。

c．将 Multi Wii 程序下载到飞控板上，下载手机控制软件，与飞控板蓝牙连接。

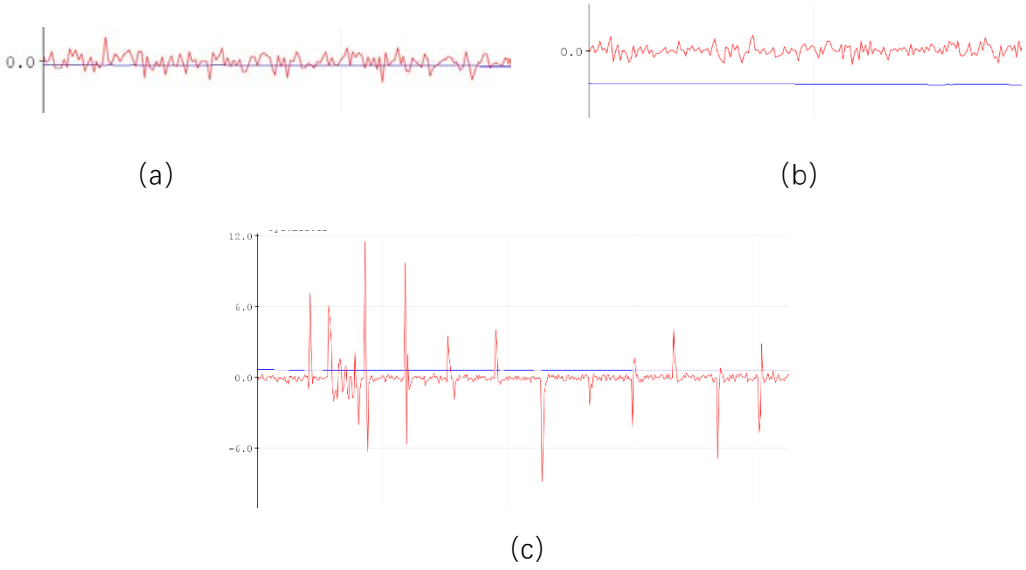
d．将飞控板水平放在桌面上一段时间，校准姿态传感器，之后使用手机 app 的案件就可以控制飞行器螺旋桨的转速和飞行方向了，根据飞行效果，可调整 PID 参数达到稳定飞行。

图 5：安装好的四轴飞行器

三．结果与简要分析

（一） MPU-6050 数据稳定性和准确性

用 Arduino IDE 自带的串口监视器，可以观察到绕 x 轴旋转角度随时间的变化情况如图 6 所示。



（a） （b）

（c）

图 6：红色是加速度计测得的绕 x 轴旋转角，蓝色是陀螺仪测得的绕 x 轴旋转角。（a）刚刚初始化，零偏矫正保证两个数据的平均值都在 0 点附近，但加速度计测得的数据波动很大。（b）一段时间后（半分钟左右），陀螺仪测得数据出现了明显的累积误差。（c）用手指轻轻敲击传感器，发现加速度计对震动十分敏感，而陀螺仪对小振动基本免疫。

接着对数据进行滤波，直接对加速度计数据进行低通滤波，即 Angular = K*gyroangular + P*accangular ，对三组不同的 K/P 值进行了测试，发现 K/P=4 时数据稳定性已经很好（图

7)，同时 K 又不会太大导致数据漂移严重（表 1）。

T=30s	K/P=24	K/P=4	K/P=1.5
AngularX shift /°	0.27	-0.13	0.04
AngularY shift /°	0.88	0.57	0.73
AngularZ shift /°	0.73	0.35	0.42

表 1：不同 K/P 值在初始化到半分钟之后数据的漂移情况。K/P=4 和 1.5 时漂移较少。但 K/P=1.5 时数据不稳定。

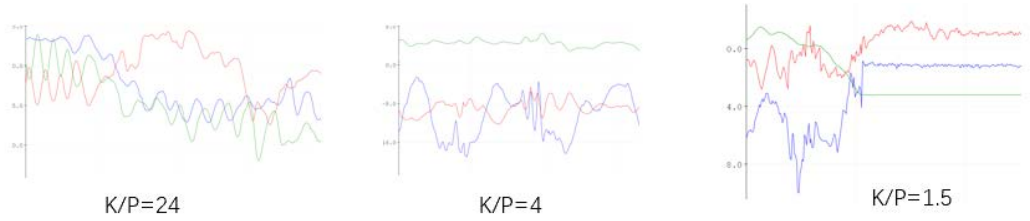
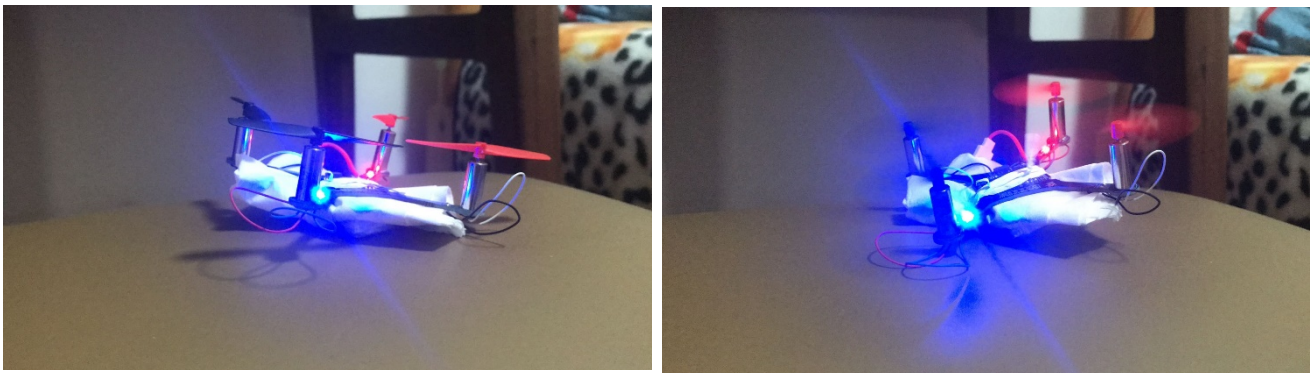


图 7：不同 K/P 值下角度-时间变化曲线的效果，K/P=24 和 4 时，曲线较为平滑。

（二） 飞行器稳定性

姿态传感器校准完成后，连接蓝牙，接通电源，测试飞行情况和姿态矫正情况（图 8），算法能够很敏感的矫正偏离水平的飞行器。



（a） （b）

图 8:四轴飞行器的飞行情况（a）矫正完成，将四轴飞行器倾斜放置（b）开小油门，发现四轴飞行器往倾斜的反方向倾斜，这是 PID 控制的结果。