**控制算法**

**1.用于0-360角度的均值函数**



python实现代码：

def mean\_sin(deg):

return sum(map(sin, map(radians, deg)))/len(deg)

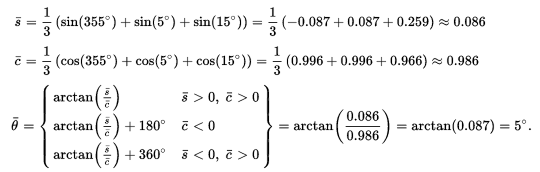
def mean\_cos(deg):

return sum(map(cos, map(radians, deg)))/len(deg)

def mean\_angle(deg):

return degrees(atan2(mean\_sin(deg), mean\_cos(deg)))

应用：



参考资料：

<http://blog.csdn.net/xiahouzuoxin/article/details/38472845>

<https://en.wikipedia.org/wiki/Circular_mean>

**2.编程实现微分积分技巧**

将连续过程转化为离散的过程，用短时间内的减法表示微分，用乘法或者加法表示积分。

以PID中的微分和积分举例如下：

pid.SetSpeed=speed; // 设置目标速度

pid.err=pid.SetSpeed-pid.ActualSpeed; // 目标速度与当前速度之间的差值

pid.integral+=pid.err; // 积分就是把这个误差不断的累计（加法）

pid.voltage=pid.Kp\*pid.err+pid.Ki\*pid.integral+pid.Kd\*(pid.err-pid.err\_last);

// 控制量，微分部分直接做差

pid.err\_last=pid.err; // 储存当前差值，用于下一次微分

pid.ActualSpeed=pid.voltage\*1.0; // 假设控制量的作用是立即的，当前速度立即改变

return pid.ActualSpeed;

**3.九轴的校准**

（1）陀螺仪的零偏

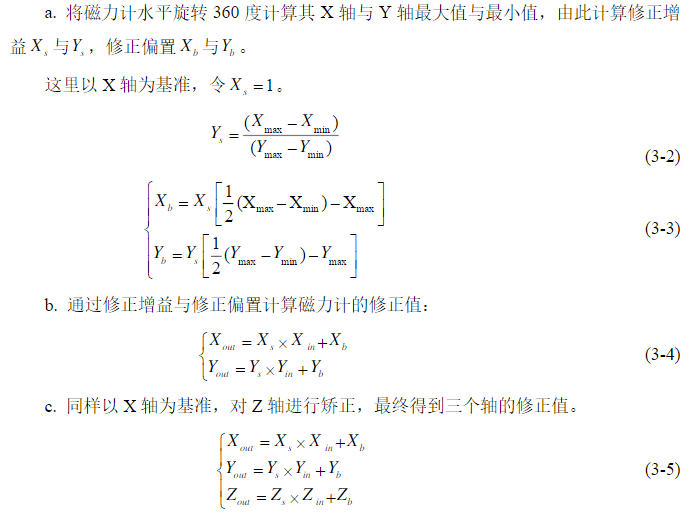
陀螺仪存在静态偏置即零偏，因此在每一次系统初始化时需要对其零偏进行计算并在之后的计算除去零偏。先将陀螺仪静置，连续读取200次陀螺仪的测量值进行均值滤波，并将其作为陀螺仪的零偏常量。

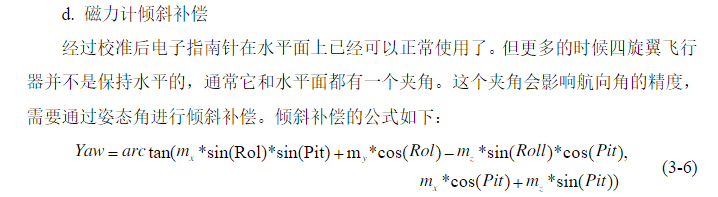
（2）加速度计校准及滤波

由于加速度传感器的测量值必然存在误差，因此，有必要在使用前对其进行校准，使用时对其进行滤波。这里使用均值滤波及滑动平均滤波的方式对加速度计进行简单处理。事实上，针对加速度计校准较好的算法为最小二乘法。通过将加速度计三个轴得到的理论值的平方和与加速度计三个轴得到的测量值的平方和进行做差得到重力加速度的平方的相反数，以此来构造平方误差，进而使用最小二乘法对误差进行估计。

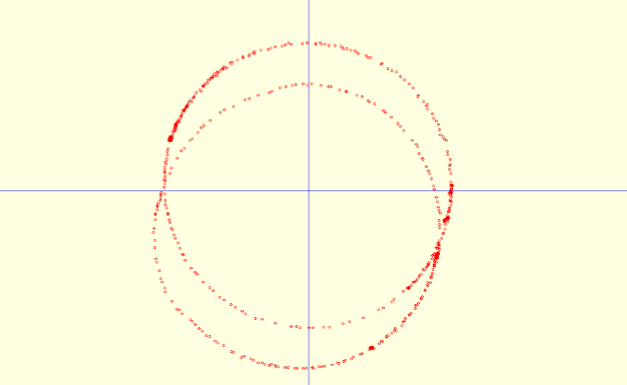
（3） 磁力计偏心矫正及倾斜补偿

理论上，磁力计的三个轴的测量值范围是对称的，即在空间中呈球体分布。然而，由于传感器的误差以及外界磁场的干扰，本文所得到的磁力计测量值往往是一个中心不在原点的椭球体，因此需要对其测量值进行偏心校正，基本步骤如下：





下图是校准之前和校准以后的对比效果：



参考资料：<http://wenku.baidu.com/view/3940f1a2910ef12d2bf9e71a.html?from=search>

**4.  PID算法中积分饱和现象的几种处理方式**

PID算法中引入积分项对四旋翼进行控制的主要目的在于消除系统的静态误差（相当于一个常数的偏置量），然而积分项的引入会导致系统的稳定性下降且容易产生超调，其中最严重的问题就是积分饱和。由于在PID控制器的执行过程中，积分项不断地累加，当积分项进入饱和区后，相对于PD控制，其需要更多的时间用于退出饱和区，这会导致系统的超调甚至震荡，并降低系统的动态响应特性。

如果执行机构已经到极限位置，仍然不能消除静态误差时，由于积分作用，尽管PID差分方程式所得的运算结果继续增大或减小，但执行机构已无相应的动作，这就叫积分饱和。

下面是集中处理的方法：

（1）积分分离法

该算法为三种算法中最简单易用的一种，它的基本思路为：当输入量与设定值偏差较大时，忽略积分项的作用，即采用PD控制，以避免系统由于积分饱和而产生不稳定的现象；当输入量与设定值的偏差接近于0时，引入积分控制，即采用PID控制，以此起到消除静态误差的作用，提高控制精度。

（2） 遇限削弱积分法

当控制器进入积分饱和区后，不再进行积分项的累加，而只进行削弱积分的运算。因此，在计算U(k)时，首先判断U(k-1)是否超出限制范围，若超出范围则只累加负偏差，反之，则累加正偏差，采用该算法来避免控制量长期停留在饱和区。

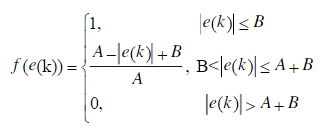
（3）变速积分法

变速积分法是一种通过改变积分速率来使得积分项的累加更为合理的算法，即偏差越大积分速度越慢，偏差越小积分速度越快。

设积分项前的系数为f(e(k))，则积分项表达式为:



其中有



参考资料：<http://wenku.baidu.com/view/3940f1a2910ef12d2bf9e71a.html?from=search>

**5.  PID算法中微分项的几种处理方式**

结合不同的PID算法，角度环的微分项存在几种不同的处理方式，主要有以下几种：

(1)  直接使用陀螺仪测量得到的角速度作为微分项。由于角度的微分即角速度，因此直接使用陀螺仪测得的角速度值最为直观简便，且因为没有引入期望值进行微分。因此，在期望值进行切换时不会对系统产生较大的干扰。然而，陀螺仪的零点飘移却是一个不可忽视的问题。随着时间，系统运动状态的改变，陀螺仪的零点可能随之改变，因而从理论上讲，应当对陀螺仪的零点进行动态修正，以保证微分项的准确性。

(2)  使用本次角度偏差与上一时刻的角度偏差的差值作为微分项。由于对期望值进行了微分（短时间内做差），因此在期望值进行切换时，存在较大扰动，若使用该方法，可对微分项进行一阶惯性滤波，使微分项较为平滑。

(3)  使用微分先行PID算法，将角度期望进行分离，直接针对当前时刻姿态解算得到的姿态角进行微分，与上一种方法相比，在期望值进行切换时，不会对系统产生较大的扰动。与第一种方法相比，由于在姿态解算时，通过加速度计与陀螺仪进行融合得到较为准确的姿态角，因此，将其用于微分项在理论上较为合理。

然而，由于第三种方法所使用的姿态解算得到的姿态角并非时刻接近真实值，在动态过程中，该现象尤为明显。当真实值进行切换时，估计值从上一时刻逼近这一时刻的真实值需要一定的过渡时间，因此，可以认为第三种方法得到的微分项是滞后的，微分项的作用体现在对未来的预测，而使用一个“过去”的控制量对“未来”进行预测控制，这显然是不合理的。在实验中则体现为系统抵抗外力的反作用力较小。因此，使用第三种方法的前提为，具备响应极快的姿态解算算法且控制频率较高。