卡尔曼滤波算法在运动控制中的应用

AI 航 团队

卡尔曼滤波（Kalman filter）是一种高效率的递归滤波器，卡尔曼滤波在技术领域有许多的应用，常见的有飞机及太空船的导引、导航及控制、信号处理、机器人运动规划及控制等，有时也包括在轨迹最佳化。它能够从一系列的不完全及包含噪声的测量中，估计动态系统的状态。卡尔曼滤波会根据各测量量在不同时间下的值，考虑各时间下的联合分布，再产生对未知变数的估计，因此会比只以单一测量量为基础的估计方式要准。

在轻舟机器人的学习过程中，我们可以在底层驱动板使用简易卡尔曼滤波器对 IMU 的数据进行估计，从而得到稳定输出，我们首先对卡尔曼滤波进行简要介绍，然后对照控制程序进行理解。本文侧重工程应用，想要深入学习卡尔曼滤波相关知识，可从以下几个参考网址入手，结合相关学术论文进行深入理解。

https://zh.wikipedia.org/wiki/%E5%8D%A1%E5%B0%94%E6%9B%BC%E6%BB%A4%E6%B3%A2

https://blog.csdn.net/heyijia0327/article/details/17487467

https://blog.csdn.net/heyijia0327/article/details/17667341

https://blog.csdn.net/victor_zy/article/details/82862904

1. 系统模型介绍

（1）系统离线型状态方程

首先我们需要用方程来描述被观测的系统，因为后面滤波器的方程要用到这里的一些参数。卡尔曼滤波器适用于线性系统，由 k-1 时刻到 k 时刻，系统状态预测方程：

$$x(k) = A \cdot x(k-1) + B \cdot u(k) + w(k)$$

系统状态观测方程

$$z(k) = H \cdot x(k) + y(k)$$

（2）变量说明

x(k) —— k 时刻系统的状态

u(k) —— 控制量

w(k) —— 符合高斯分布的过程噪声，其协方差在下文中为 Q

z(k) —— k 时刻系统的观测值

y(k) —— 符合高斯分布的测量噪声，其协方差在下文中为 R

A——状态转移矩阵

B——输入增益矩阵

H——量测矩阵

A、B、H —— 系统参数，多输入多输出时为矩阵，单输入单输出时就是几个常数

注意在后面滤波器的方程中我们将不会再直接面对两个噪声 w(k)和 y(k)，而是用到他们的协方差 Q 和 R。至此，A、B、H、Q、R 这几个参数都由被观测的系统本身和测量过程中的噪声确定了。

2. 卡尔曼滤波器

卡尔曼估计实际由两个过程组成：预测与校正，在预测阶段，滤波器使用上一状态的估

计，做出对当前状态的预测。在校正阶段，滤波器利用对当前状态的观测值修正在预测阶段获得的预测值，以获得一个更接进真实值的新估计值。

（1）卡尔曼滤波计算公式

预测：

$$x(k|k-1) = A \cdot x(k-1|k-1) + B \cdot u(k)$$

$$P(k|k-1) = A \cdot P(k-1|k-1) \cdot A\text{^}T + Q$$

校正：

$$z(k) = H \cdot x(k) + y(k)$$

$$K(k) = P(k|k-1) \cdot H\text{^}T \cdot (H \cdot P(k|k-1) \cdot H\text{^}T + R)\text{^}(-1)$$

$$x(k|k) = x(k|k-1) + K(k) \cdot (z(k) - H \cdot x(k|k-1))$$

更新协方差估计：

$$P(k|k) = (I - K(k) \cdot H) \cdot P(k|k-1)$$

（2）推导过程简述

工程中用到的卡尔曼滤波是一个迭代的过程，每得到一个新的观测值迭代一次，来回来去地更新两个东西：“系统状态”(x)和“误差协方差”(P)。由于每次迭代只用到上一次迭代的结果和新的测量值，这样滤波对计算资源的占用是很小的。

每一次迭代，两个步骤：预测和修正。预测是根据前一时刻迭代的结果，即 x(k-1|k-1) 和 P(k-1|k-1)，来预测这一时刻的系统状态和误差协方差，得到 x(k|k-1)和 P(k|k-1)：

$$x(k|k-1) = A \cdot x(k-1|k-1) + B \cdot u(k)$$

$$P(k|k-1) = A \cdot P(k-1|k-1) \cdot A^T + Q$$

(这里用到的 A、B、H、Q、R 就是从前面的状态/观测方程中拿来的)

然后计算卡尔曼增益 K(k)，和这一次的实际测量结果 z(k)一起，用于修正系统状态 x(k|k-1)及误差协方差 P(k|k-1)，得到最新的 x(k|k)和 P(k|k)：

$$K(k) = P(k|k-1) \cdot H^T \cdot (H \cdot P(k|k-1) \cdot H^T + R)^{-1}$$

$$x(k|k) = x(k|k-1) + K(k) \cdot (z(k) - H \cdot x(k|k-1))$$

$$P(k|k) = (I - K(k) \cdot H) \cdot P(k|k-1)$$

至此这次迭代就算结束了，**x(k|k)就是我们要的滤波后的值，它和 P(k|k)将会作为 x(k-1|k-1)和 P(k-1|k-1)用在下一时刻的迭代里**。别看现在公式还是一大堆，在我们所谓的“简单场景”下，系统参数一定下来，就能简化很多很多。

先看状态方程和观测方程。假设我们是在测温度、加速度或者记录蓝牙 RSSI，此时控制量是没有的，即：

$$B \cdot u(k) \equiv 0$$

另外，参数 A 和 H 也简单地取 1。现在滤波器的预测方程简化为：

① $x(k|k-1) = x(k-1|k-1)$

② $P(k|k-1) = P(k-1|k-1) + Q$

同时修正方程变成：

$$③\ K(k) = P(k|k-1) / (P(k|k-1) + R)$$

$$④\ x(k|k) = x(k|k-1) + K(k) \cdot (z(k) - x(k|k-1))$$

$$⑤\ P(k|k) = (1 - K(k)) \cdot P(k|k-1)$$

①对②、③无影响，于是④可以写成：

$$x(k|k) = x(k-1|k-1) + K(k) \cdot (z(k) - x(k-1|k-1))$$

在程序中，该式写起来更简单：

x = x + K * (新观测值 - x);

观察②，对这一时刻的预测值不就是上一时刻的修正值+Q 嘛，不妨把它合并到上一次迭代中，即⑤改写成：

$$P(k+1|k) = (1 - K(k)) \cdot P(k|k-1) + Q$$

这一时刻的 $P(k+1|k)$，会作为下一时刻的 $P(k|k-1)$，刚好是③需要的。于是整个滤波过程只用这三个式子来迭代即可：

$K(k) = P(k|k-1) / (P(k|k-1) + R)$
$x(k|k) = x(k-1|k-1) + K(k) \cdot (z(k) - x(k-1|k-1))$
$P(k+1|k) = (1 - K(k)) \cdot P(k|k-1) + Q$

3. 卡尔曼滤波算法的实现

```
#include "filter.h"
//变量声明
float K1 =0.02;
float angle, angle_dot;     // 角度，角速度
float Q_angle=0.001;        // 陀螺仪噪声的协方差
float Q_gyro=0.003;         // 陀螺仪漂移噪声的协方差
float R_angle=0.5;          // 加速度计测量噪声的协方差
float dt=0.005;             // 积分时间，dt 为滤波器采样时间（秒）
char   C_0 = 1;             // H 矩阵的一个数
float Q_bias, Angle_err;    // Q_bias 为陀螺仪漂移
float PCt_0, PCt_1, E;      //中间变量
float K_0, K_1, t_0, t_1;   //K 是卡尔曼增益，t 是中间变量
float Pdot[4] ={0,0,0,0};   //计算 P 矩阵的中间变量
float PP[2][2] = { { 1, 0 },{ 0, 1 } }; //公式中 P 矩阵，X 的协方差
```

函数功能：简易卡尔曼滤波
入口参数：加速度、角速度
返回　值：无
X(k|k-1)=A X(k-1|k-1)+B U(k) ………(1)先验估计
P(k|k-1)=A P(k-1|k-1) A'+Q ………(2)协方差矩阵的预测
Kg(k)= P(k|k-1) H' / (HP(k|k-1) H' + R) ………(3)计算卡尔曼增益
X(k|k)= X(k|k-1)+Kg(k) (Z(k) - H X(k|k-1)) ………(4)通过卡尔曼增益进行修正
P(k|k)=（I-Kg(k) H）P(k|k-1) ………(5)更新协方差阵
前面预测的结果就是先验，测量出的结果就是后验

```
void Kalman_Filter(float Accel,float Gyro)     //Gyro 陀螺仪的量测值，Accel 加速度计的角度计算值
{
	angle+=(Gyro - Q_bias) * dt;                //先验估计  对应第一个公式

	Pdot[0]=Q_angle - PP[0][1] - PP[1][0]; // Pk-先验估计误差协方差的微分
	Pdot[1]=-PP[1][1];
	Pdot[2]=-PP[1][1];
	Pdot[3]=Q_gyro;
	PP[0][0] += Pdot[0] * dt;    // Pk-先验估计误差协方差微分的积分
	PP[0][1] += Pdot[1] * dt;    // =先验估计误差协方差
	PP[1][0] += Pdot[2] * dt;
	PP[1][1] += Pdot[3] * dt;
	//以上 8 行对应第二个方程

	//计算卡尔曼增益 K_0,K_1
	Angle_err = Accel - angle;      //zk-先验估计

	PCt_0 = C_0 * PP[0][0];        //矩阵乘法的中间变量
	PCt_1 = C_0 * PP[1][0];        //C_0=1

	E = R_angle + C_0 * PCt_0;   //分母

	K_0 = PCt_0 / E;                //卡尔曼增益，两个，一个是 Angle 的，一个是 Q_bias 的
	K_1 = PCt_1 / E;
	//以上 6 行对应公式 3

	//对 PK 进行更新
	t_0 = PCt_0;                    //矩阵计算中间变量，相当于 a
	t_1 = C_0 * PP[0][1];           //矩阵计算中间变量，相当于 b

	PP[0][0] -= K_0 * t_0;          //后验估计误差协方差
	PP[0][1] -= K_0 * t_1;
	PP[1][0] -= K_1 * t_0;
	PP[1][1] -= K_1 * t_1;
	//以上 6 行对应公式 5

	//通过卡尔曼增益修正当前角度、陀螺仪零点、当前角速度
	angle += K_0 * Angle_err;         //后验估计
	Q_bias     += K_1 * Angle_err;   //后验估计
	angle_dot    = Gyro - Q_bias;      //输出值(后验估计)的微分=角速度
	//对应第 4 个方程
}
```