# 使用KalmanFilter对手机IMU运动传感器进行朝向估计测试&问题报告

## Changelog

|  |  |  |  |
| --- | --- | --- | --- |
| 版本号 | 变更人 | 变更说明 | 变更时间 |
| V1.0 | 张琛 | 初稿 | 2015/04/06 |

## 编写目的

**实现**一个基于Extended Kalman Filter (EKF) 的传感器融合模型，并进行**代码正确性测试**，以及模型滤波结果与Android结果**对比测试**。

## 问题描述

**已知**手机IMU单元输出的9DOF运动传感器数据：

1. 三轴加速度计数值序列：

2. 三轴磁通传感器数值序列：

3. 三轴陀螺仪数值序列：

**输出**应用KalmanFilter进行IMU单元的设备姿态估计（单位四元数Q表示）

**基本模型**：

(1)

(2)

其中，为k时刻的状态向量，为已知（或测量）的系统输入，是过程激励噪声；是观测变量（或系统输出），是观测噪声。本文目的在于估计设备朝向，使用四元数表示。传感器输出的3D观测值有：，作为观测值时分别表示为，各自采样频率分别为。

为了表述清晰，我们将三组**输入**观测值序列**概念上分为模型输入与输出**（输入还是输出取决于所设计的滤波器模型结构）。目前我们使用较**简单模型**：

(3)

即，**将作为控制输入，而作为观测值输出**。目前状态量仅包含四元数，如果要进一步估计相机移动轨迹，还可以将位移与速度作为状态量加入，此时可以做状态量，也可以作为观测值输出。

## 实现步骤&结果

EKF滤波器分为两个更新步骤[2]：时间更新（预测），测量更新（校正），如图1所示：

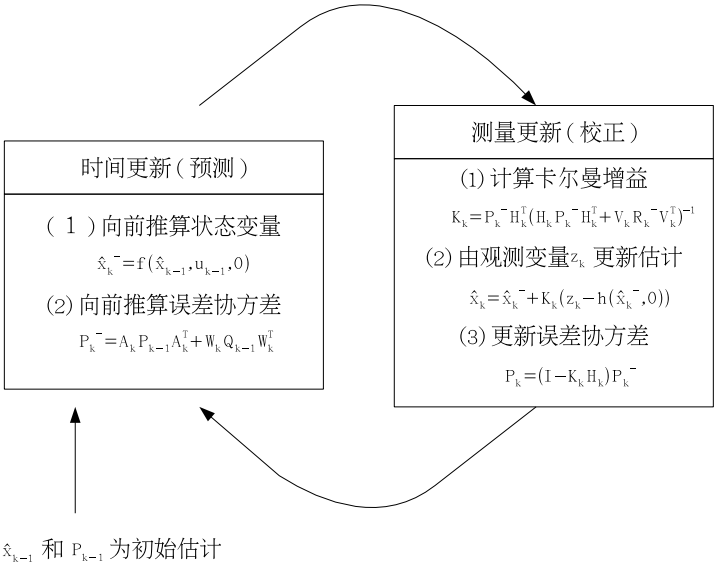


图 1 EKF工作原理图

我们按照文献[1]进行模型实现如下。

### 时间更新

#### 公式描述

由公式（3）得知，，四元数作为状态量，为4\*1列向量。基于角速度做控制输入（）的离散时间更新推导如图2：

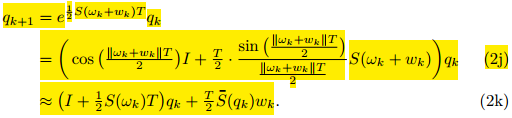


图 2时间更新线性化近似过程

其中， 分别如文献[1]-(2d), (2e) 所示，如图3：

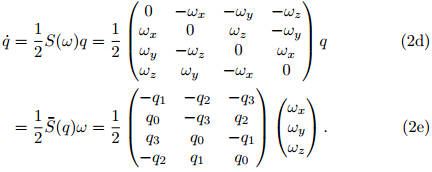


图 3 计算方法

#### 代码实现

参考图2，时间更新函数实现如下：

|  |
| --- |
| function [x, P] = tu\_qw(x, P, omega, T, Rw)  A = 0.5\*Somega(omega);  G = 0.5\*T\*Sq(x);  x = (eye(4) + A\*T)\*x + G\*Rw;  P = A\*P\*A';  x = mu\_normalizeQ(x);  end |

**x，P**分别为状态（4\*1）与误差协方差矩阵（4\*4），**omega**为一帧三轴角速度数据（3\*1），**T**为时间间隔（单位：s），**Rw**为过程噪声协方差矩阵。其中，Somega, Sq 函数分别为的实现，具体代码见附件Somega.m，Sq.m文件。

函数tu\_qw的调用初始化为：

x = [1; 0; 0 ;0];

P = eye(4);

Rw = -[-0.0084 -0.0176 0.0120]';

Rw 值为静止状态下测得的三轴角速度**平均偏移量**。

#### 结果测试 & 结论

对最终输出变量，两组四元数meas.orient, xhat.x进行处理，转换为第i帧相对第0帧的变化的欧拉角（弧度制），代码如下（见analyse.m文件）：

|  |
| --- |
| %by zhangxaochen: filterTemplate 滤波器跑完之后， 分析对比 OWN, GOOGLE 各自得到的朝向误差：  % 1. google：  qs = meas.orient'; %quaternions, n\*4  % 第k帧相对第0帧的累积旋转 q  tdfqm = diffq(repmat(qs(1, :), size(qs, 1)-1, 1), qs(2:end, :));  % 2. own：  qs = xhat.x';  tdfqx = diffq(repmat(qs(1, :), size(qs, 1)-1, 1), qs(2:end, :));  % total euler from meas.orient:  tdfem = q2euler(tdfqm')';  % total euler from xhat.x:  tdfex = q2euler(tdfqx')';  % 绘制两组（2\*3）Δ欧拉角曲线：  plot([tdfem, tdfex])  legend('google.x', 'google.y', 'google.z', 'own.x', 'own.y', 'own.z') |

图 4 BGR与CMY色曲线分别表示Google与我们的旋转结果，对应绘制命令 plot([tdfem, tdfex])，即，第i帧相对第0帧的变化的欧拉角（**采样动作**：静止约20s）

图 5 RGB表示对于“第i帧相对i-1帧欧拉角变化量”，google计算结果与我们的结果的差值，（**采样动作**：静止10s，绕X轴缓慢旋转+90°，静止10s，绕X轴缓慢旋转-90°）

##### 结论

1. 在 Rw 输入手动调整至合适值时， 我们的**时间更新**结果可以做到相对准确；

2. Rw **受温度变化影响较大**，预先测定好的Rw值在之后的使用过程中也有可能偏差很大（假设仅看时间更新）

### 测量更新（加速度部分）

#### 公式描述

对于加速度，由先验估计预测测量变量的表达式如下图（文献[1]公式(5)）：



图 6 加速度的测量变量预测方法

为从世界坐标系到机身坐标系的旋转矩阵，为世界坐标系重力加速度经验值，个人测得值为：g0 = [0, 0, 9.6680]。假定手机一直处于静止或缓慢转动过程中，没有施加外力，即为测量噪声。

#### 代码实现

参考图1 中EKF测量更新工作原理，代码实现如下（详见mu\_g.m文件）。其中，yacc为实际第i帧加速度测量值，Ra为**静止状态**下加速度误差协方差矩阵，为**经验值**。

|  |
| --- |
| function [ x, P ] = mu\_g( x, P, yacc, Ra, g0 )  %zhangxaochen: ²Î¿¼ http://pastebin.com/C00tedaN  xt = [x(1), -x(2), -x(3), -x(4)]; %ËÄÔªÊýÄæ±ä»»£¬Ïàµ±ÓÚÈ¡ Q.T    [Q1, Q2, Q3, Q4] = dQqdq(xt);  if size(g0,1)==1  g0=g0';  end  H = [Q1\*g0, Q2\*g0, Q3\*g0, Q4\*g0];  y\_l = Qq(xt)\*g0;  y = yacc - y\_l;    K = P\*H'/(H\*P\*H' + Ra); %²»ÓÃ inv, eye  P = P - K\*H\*P;    x = x + K\*y;  x = mu\_normalizeQ(x); |

#### 结果测试 & 结论

使用加速度进行测量更新，理论上应达到的效果是，计算得到的朝向四元数能够“识别”竖直向上方向。但是实际测试时往往结果错误（以google朝向计算结果为groundtruth）：



(a) (b)

(c) (d)

图 7 加入加速度“校正”四元数计算结果

由图7(a)~(d)看到，传感器开启时，不管手机处于什么姿态下，google的计算结果总能迅速收敛到（大致）正确结果，而我们自己的测量更新结果则往往失败。

##### 结论

1. 通过对代码运行时跟踪，发现误差协方差矩阵P, 卡尔曼增益矩阵 K总是收敛为零矩阵，导致加速度对预测值的“校正”失效。目前尚未找到问题原因。

2. 因为我们的目的在于，使用实验室的IMU单元进行朝向估计，目前打算：①测试杜宇之前的api估计结果；②参考android内部sensor fusion的代码（c++实现），将其移植到PC上。

### 测量更新（磁通部分）

存在问题与3.2类似，过程略。

## 参考文献

[1] Gustafsson F. TSRT14 Sensor fusion; Linköpings universitet[J]. 2014.

[2] Welch G, Bishop G. An Introduction to the Kalman Filter. 2006[J]. University of North Carolina: Chapel Hill, North Carolina, US.