

**本科毕业设计（论文）外文翻译译文**

**学生姓名**： 李永辉

**院 （系）：** 计算机学院

**专业班级**： 计算机科学与技术 计1601

**指导教师**： 康磊

**完成日期**： 20 年 月 日

**要 求**

1、外文翻译是毕业设计（论文）的主要内容之一，必须学生独立完成。

2、外文翻译译文内容应与学生的专业或毕业设计（论文）内容相关，不得少于15000印刷符号。

3.外文翻译译文用A4纸打印。文章标题用3号宋体，章节标题用4号宋体，正文用小4号宋体，20磅行距；页边距上、下、左、右均为2.5cm，左侧装订，装订线0.5cm。按中文翻译在上，外文原文在下的顺序装订。

4、年月日等的填写，用阿拉伯数字书写，要符合《关于出版物上数字用法的试行规定》，如“2018年2月26日”。

5、所有签名必须手写，不得打印。

**文献名称（中文）**

**外文翻译译文：**

基于陀螺仪数据的计步器算法

A Gyroscopic Data based Pedometer Algorithm

作者： Sampath Jayalath

斯里兰卡信息技术学院电气与计算机工程系

斯里兰卡马拉贝新坎迪路

adasdjsampath@gmail.com

起止页码：551-555

出版日期（期刊号）：2013年4月26日至28日

出版单位：第八届计算机科学与教育国际会议（ICCSE 2013）

外文翻译译文：

摘要

计步的准确性是当前计步器中存在的主要问题之一，尤其是在平坦的土地上缓慢行走并执行不同的活动（例如，上下楼梯和在倾斜平面上行走）时。尽管基于加速度计的计步器在较高速度下行走时可提供合理的精度，但在慢速行走和执行不同活动时，其准确性仍不足。本文提出了一种新颖的算法，该算法使用嵌入在移动设备中的单点陀螺仪传感器来检测步距。对在男性和女性志愿者的参与下在不同环境中收集的数据进行的初步分析表明，陀螺仪本身可以提供足够的信息来进行准确的步幅检测。该算法是基于陀螺仪陀螺数据，同时结合了过零和阀值检测技术而开发的。结果证明，基于陀螺仪的步检测算法在执行不同活动和慢速步行时提供较高的准确性。

关键字：计步器算法 陀螺仪数据 单点传感器 成批设备 移动应用程序

1.介绍

现代医学研究强调，计步器不仅可以统计人体行为，而且也在很大程度上统计人类的心理活动[1]，[2]，[3]。低成本的计步器有助于提高步行者的动力[4]，还可以应用于室内导航、活动识别以及各种医疗保健领域。计步器可用于检测人体垂直加速度产生的步数。这在两种机制下起作用。一种是基于机械的加速度计，另一种是基于电子的加速度计。现代计步器通常基于MEMS（微机电系统）加速度计，主要基于1轴，但是也有2轴和3轴加速度计，陀螺仪和磁力计的使用提高了精度并释放了一些使用限制，例如计步器定位。这些系统的精度处于可接受的水平，但是还不够完美，因为存在着各种缺陷[5]，计步器的应用程序现已升级，可以在移动设备中找到它。随着计步器在移动设备上的应用显而易见，现在已经提高了医疗应用的标准。

研究人员提出的某些计步器算法的方法将在背景技术部分进行讨论，包括其特征和缺点。

2.背景

S.E Crouter等人[6]比较了市场上可用的10个计步器的准确性和可靠性。这些计步器基于诸如加速度计，金属对金属和磁簧片接近开关之类的机制。 注意所有测试均以正常的步行速度进行，这很重要。他们的结论是，计步器的准确性在很大程度上取决于内部机制和灵敏度。但是，当他们缓慢行走并进行诸如上下楼梯之类的活动时，他们无法测量计步器的准确性。

Jerome和Albright [7]用市售的计步器进行的比较研究表明，准确度很差，最小平均绝对误差值为13％。他们的结论是，这些计步器均不能用于研究目的或通常用途。

Wasiq Waqar等人[8]为他们的“ 室内定位系统”开发了一种基于加速度计的计步器，该计步器包括一个基于阈值的预设阈值检测方法来识别有效步长，而步周期模式检测方法则要丢弃由于加速度计的瞬时读数而导致无效步长。应该注意的是，由于预设阈值，计步器的结果可能会随不同的步行模式和速度而变化。

Melis Oner等人[9]为他们的“坠落早期检测系统”实现了另一步检测算法。该特定算法的阶跃检测依赖于步行过程中加速度计传感器产生的数据中某个时间段内的峰值检测。他们能够在较高的步行速度下以及将移动式计步器固定并宽松地放在口袋中时获得更高的精度。但是，他们的算法无法在缓慢的步行过程中准确地计算步数。

Mi-hee Leer等人[10]通过一些先进的处理，如FFT，Fuzzy C和统计计算，能够在其便携式加速度传感器模块中实现99％的精度。但是他们最终一直认为，他们的系统不能实时处理数据，无法在活动过程中测量步数，例如上楼梯和下楼梯，并且需要一种有效的设备来进行处理。

A.M. Cavalcante等人[11]开发了一种计步器，用于“移动设备上的室内实时跟踪”研究。该系统结合滑动窗口机制比较加速度样本的平均值以检测步。他们的结论是，适当的采样率，滑动窗口和嵌入在移动设备中的质量传感器是准确检测步的主要要求。

据Garcia等人[12]对基于手机的计步器与市售的计步器精度的比较研究得出的结论是，与市售的计步器相比，手机具有竞争优势。此外，当速度较慢时，两者的准确性均较低；而步行较快时，两者的准确性均较高。

Lim等人[13]提出了一种基于陀螺仪的计步器，但并未对其实际使用进行深入讨论。 Zhong等人[14]还讨论了基于加速度计传感器的计步器，该传感器附在脚上，并在PDA上进行处理，能够达到90％以上的准确度。但是这些计步器都不能方便地将任何传感器与身体相连，因此无法与任何应用程序配合使用。

根据G. Boyce等人[15]对市场上免费提供的基于手机的计步器技术准确性的比较研究得出的结论是，在确定步数时，基于手机的应用程序落后于使用商用计步器。需要进行设置的进一步操作以提高一个活动级别的步数计数的准确性，但是随着活动强度的变化，需要重新校准。

M.Ayabe等人[16]对计步器在上下楼梯时的准确性进行了重要的研究。他们使用了一个弹簧杠杆式和两个压电式电子计步器来测试准确性。他们的结论是，计步器可以在爬楼梯活动期间以80步每分钟的步速或在18厘米或更高的楼梯速度下更快地在可接受的测量误差范围内准确评估步数。但是，应该指出的是，他们已经强调了在较慢的步进速度下精度较差。

以前的大多数研究已经确定，计步器在慢速行走时效果较差，可靠性较差。这种不准确性是由于臀部的较小垂直运动引起的，该运动低于特定的计步器阈值，因此未被记录。计步器的另一个限制是它不包含内部时钟。因此，不可能确定所进行活动的强度或持续时间。

3.实施算法

该算法是基于N. Abhayasinghe等人在“室内定位与导航”[17]理论研究所做的实验结果提出的。该特定算法仅使用由移动设备产生的陀螺仪数据来预测步长。可以描述如下：

A.算法背后的初始概念

步行过程中腿部运动表现出正弦行为。通过监视腿的角速度，可以清楚地识别这种行为。因此，陀螺仪的一个轴根据设备的方向提供有关腿部运动的信息。一项关于“将现代移动设备放在口袋中的方式”的小型研究证明，几乎所有用户都将其设备垂直放置在口袋中。因此，考虑监视陀螺x轴数据。重要的是要注意，对于不同的方向，仅需要更改的变量就是我们从陀螺仪获取数据的轴。

B.过滤原始数据

最初，原始数据是使用适当的过滤器过滤的，该过滤器将保留行走的属性。典型的步行速度可能在2 Hz左右，而跑步可能会翻倍，慢速步行可能是其一半。选择适合慢节奏活动的截止频率是实现更高精度的主要考虑因素。因此，滤波器的选择是具有截止频率在0.9 Hz至3 Hz范围内的简单的六阶Butter有价值的低通滤波器。图1描绘了原始gyro-x值及其滤波版本（2 Hz）。

C.去除不需要的信号成分

陀螺仪数据在对设备的阶跃和瞬时运动进行滤波之后，呈现正弦曲线行为。采样技术用于减少由于设备的瞬时移动而导致的错误步数。根据实验结果，根据行走的强度，平均OAO会发生变化，平均达到1.20秒。因此，一旦检测到步长，就可以将算法设置为消除在平均步长时间之前可以算作有效步长的任何信号。

D.滤波信号关键特征的识别

此算法中步距检测的主要思想取决于检测经过滤波的gyro-x产生的数据中的零交叉。图2说明了单步幅发生的连续零交叉。除了过零以外，还使用自适应峰值阈值来验证步骤。个人可以训练算法以学习可用于实现有效步长的最小信号峰值，尤其是在尽可能缓慢移动和下楼梯时。因此，此峰值阈值用作验证步骤的参数。阈值峰值检测有助于避免设备的瞬时移动和微小移动。此外，可以调整此阈值峰值，以便算法能够检测到将设备放置在松散或紧紧的不同口袋中时的步长（信号强度随设备放置的环境而异）。图2，图3和图4描绘了在行走（平均= 1.3rad / s），上升（平均= 1.6rad / s）和下降（平均= 0.9）时陀螺仪x信号（向上信号）的强度rad / s）个人楼梯。重要的是要注意，这些平均值随行走方式和强度的不同而变化。

图1.陀螺仪X轴读数（2 Hz时的原始值和滤波值）

图2.以中等速度行走时陀螺x信号的强度

图3.上楼梯时陀螺x信号的强度

图4.下降楼梯时陀螺x信号的强度

E.实验测试及结果

苹果设备用于测试计步器算法，提取时提供更高的数据速率和准确性来自嵌入式传感器的数据。 提出的算法是在iPhone 4S和iPod 4G上实现。该算法的测试是在5名男性和5名女性成员，身高、体重随机。 由于此算法旨在独立检测步数年龄，性别和人类身体各个方面不太关心。当应用程序启动时，我们假设初始设备的方向更接近零度。 然后用户被指示执行不同的活动，例如爬山上下楼梯，在倾斜的平面上行走。 样品表1中列出了个人的已计步数。

表1. 针对个人执行不同活动的算法的实验结果

对于参与测试过程的每个人，维护了相似的阅读资料。 综合结果列于表2

表2. 算法的实验结果所有参与者

此外，在多个男性和女性志愿者的参与下，测试了该算法在不同步行速度和步进速度下的准确性。表3显示了在平坦土地上不同步行速度下精度的变化，而表4和表5中列出了不同踏步速度下的精度。

表3. 不同步行速度下算法的准确性

表4. 加速下不同步调算法的准确性

表5. 减速不同步调算法的准确性

该算法的总体准确度在94％以上。 从结果可以明显看出，爬下楼梯的准确性较低。 这主要是由于用户的信号强度较弱。 除此之外，其他错误是由于步行开始和结束时不同个体的信号强度较弱所致。

F.算法的优势

基于陀螺仪数据的阶跃检测算法有很多优点。一个主要优点是能够以慢速检测步行和上、下脚步的能力。这是通过设置适当的滤波过程以及用户设置的自适应峰值阈值来实现的。重要的是要注意，在慢速度的活动强度下，计步器的准确度是研究活动和计步器消费者的主要要求。

除此以外，该算法仅取决于陀螺仪单轴的数据。这提供了较少的计算，并且没有能力将该功能集成到室内导航等研究领域。

在当前可用的高端移动设备中实现了进一步的算法，由于传感器未附着在人体上，因此更便于与任何应用程序一起使用。

G.未来的工作

    在本文中，我们讨论了即使在步行速度较慢的情况下也可用于检测脚步的算法。在该特定算法中，用户可能需要更改阈值，以便与放置设备的环境变化时的准确步数保持一致。因此，设计一个可以在不断变化的环境中学习不同阈值的神经网络可以解决此不便之处。

    该算法的错误百分比是由于它正在检测手机的移动以验证步骤这一事实造成的。因此，我们将致力于改进解决方案，以免出现此类错误计数。

    将Barrow仪表与所提出的算法集成在一起，可以更好地解决由于信号强度较弱而在降压过程中发现的误差。

    将活动识别算法与所提出的算法进一步集成可以提高不同活动期间的准确性。

4.结论

从过去的研究结果可以明显看出，大多数计步器算法在慢速行走（50-70步/分钟）时以及在以不同速度执行不同活动时，表现都非常差。 因此，开发现代计步器的研究者和消费者的主要需求是开发一种可靠的算法，该算法可在任何步行速度下以不同的步行模式执行。 本文提出的算法的关键点包括工程技术，例如噪声消除，过零和阈值检测。 此外，它利用人腿的自然旋转运动来预测步伐，而不是身体的线性运动。 该算法的总体准确性高于94％，其中，慢速行走时的准确性非常高。 同样重要的是要注意，可以通过所提的工程技术进一步提高此准确性。

致谢

作者要感谢澳大利亚科廷大学的lain Murray博士提供了完成该任务所需的所有硬件和软件配件，并感谢所有志愿者（斯里兰卡马拉比信息技术学院的学生）在进行的实验中的参与。

参考文献

[1]AZ. LaCroix, S.G. Leveille, J.A. Hecht, L.c. Grothaus and E.H. Wagner, Does walking decrease the risk of cardiovascular disease hospitalizations and death in older adults?, Journal of the American Geriatrics Society, vol. 44, no.2, pp.113-120, Feb 1996.

[2]E.M. Simonsick, .l.M. Guralnik, S. Volpato, .I. Balfour, and L.P. Fried, Just get out of the door! Importance of walking outside the home for maintaining mobility: Findings from the women's health and aging study, Journal of the American Geriatrics Society, vol.53, no.2, pp.19S- 203, Feb 2005.

[3]Y. Hatano, Use of the pedometer for promoting daily walking exercise, International Council for Health, Physical Education, and Recreation, vol. 29, pp.4-S, 1993.

[4]E. Garcia, Hang Ding, A Sarela and M. Karunanithi, Can a mobile phone be used as a pedometer in an outpatient cardiac rehabilitation program?, in IEEE/ICME International Conference on Complex Medical Engineering (CME) 2010., 2010, pp.250-253.

[5]D.S. Michaud, E. Giovannucci, W.C. Willett, G.A. Colditz, M.l Stampfer and C.S. Fuchs, Physical activity, obesity, height, and the risk of pancreatic cancer, Journal of the American Medical Association, vol. 2S6, no.S, pp. 921-929, Aug 2001.

[6]S.E. Crouter, E. Scott, P.L. Schneider, M. Karabulut and D.R. Bassett, Validity of 10 electronics pedometers for measuring steps, distance and energy cost, Medicine and Science in Sports and Exercise, vol. 35, no.S, pp.1455-1460, 2003.

[7]G. .I. Jerome and C. Albright. (2011, June). Accuracy of Five Talking Pedometers under Controlled Conditions. The Journal of Blindness

Innovation and Research. [On-line]. 1(2). Available: www.nfb­ jbir.org/index.php/JBIR/article/view!l7/3S [Oct. 27, 2011].

[S] W. Waqar, A Vardy and Y. Chen. Motion Modelling using Smart phones for Indoor Mobilephone Positioning, presented at the 20th Newfoundland Electrical and Computer Engineering Conference, Newfoundland, Canada, 2011.

[8]M. Oner, J.A. Pulcifer-Stump, P. Seeling and T. Kaya. Towards the Run and Walk Activity Classification through Step Detection- An Androi Application,in 34th Annual International Conference of the IEEE Engineering in Medicine and Biology, 2012, pp.19S0-19S3.

[9]M. Lee, J. Kim, K. Kim, I. Lee, S.H. .lee and S.K. Yoo. Physical Activity Recognition Using a Single Tri-Axis Accelerometer, in Proceedings of the World Congress on Engineering and Computer Science, vol I, pp.14-17, October 2009.

[10]AM. Cavalcante, E.B. Souza, J.J. Bazzo, N. Bezerra, A Pontesand and RD. Vieira. A Pedometer-Based System for Real-time Indoor Tracking on Mobile Devices, in XXIX Brazilian Symposium on Telecommunications, Curitiba, Brazil, Oct,2011. [Online]. Available: http://mwsl.unb.brlindex.php/pt/publicacoes [Apr. 12,2012].

[11]E. Garcia, H. Ding, A Sarela and M. Karunanithi, Can a mobile phone be used as a pedometer in an outpatient cardiac rehabilitation program?, in IEEE/ICME International Conference on Complex Medical Engineering (CME) 2010., 2010, pp.250-253.

[12]Y.P. Lim, LT. Brown and lC.T. Khoo, An accurate and robust gyroscope-gased pedometer, in 30th Annual International Conference of the IEEE Engineering in Medicine and Biology Society, 200S. EMBS 200S., 200S, pp.45S7-4590.

[13]S. Zhong, L. Wang, AM. Bernardos and M. Song, An accurate and adaptive pedometer integrated in mobile health application, in IET International Conference on Wireless Sensor Network, 2010. IET­ WSN., 2010, pp.7S-S3.

[14]G. Boyce, G. Padmasekara, M. Blum. Accuracy of Mobile Phone Pedometer Technology. Journal of Mobile Technology in Medicine, vol. I, pp.16-22, June 2012.

[15]M. Ayabe, l Aoki, K. Ishii, K. Takayama, H. Tanaka Pedometer accuracy during stair climbing and bench stepping exercises. Journal of Sports Science and Medicine, vol. 7, pp.249-254, June 200S.

[16]N. Abhayasinghe, I. Murray, A Novel Approach for Indoor Localization Using Human Gait Analysis with Gyroscopic Data, in International Conferenc on Indoor Positioning and Indoor Navigation,November 2012, [Online]. Available :www.surveysurveying.unsw.edu.aulipin2012/papers.php [Dec.10,2012].

在嵌入式应用中使用IMU（加速度计和陀螺仪设备）的指南

A Guide To using IMU (Accelerometer and Gyroscope Devices) in Embedded Applications.

作者： STARLINO

起止页码：1-15

出版日期（期刊号）：2009年12月29日

出版单位：http://www.starlino.com/imu\_guide.html

介绍

当前， 该文章已经使用法语翻译为PDF, 感谢Daniel Le Guern！

本指南面向所有对惯性MEMS(微型电机系统)传感器感兴趣的人，尤其是加速度计和陀螺仪以及组合式的IMU设备（惯性测量单元）。

图1 Acc\_Gyro\_6DOF单元

例如IMU单元，图1：在Acc\_Gyro\_6DOF上的处理器顶部的UsbThumb 提供USB/串口连接

我将在本文中介绍一些基本并且很重要的主题：

-加速度计能测量什么

-陀螺仪（又称陀螺计）能测量什么

-如何将你从传感器获得的这些通过数模转换（ADC）后的数据转换为物理量（加速度的单位为g, 陀螺仪单位为deg/s)

-如何将加速度和陀螺仪的数据结合起来使用, 用于获得设备相对于地面的倾斜度的准确信息

在整片文章中, 我将会尽量减少数学运算, 如果你知道什么是 正弦/余弦/正切, 那么无论你使用什么芯片, 如Arduino，Propeller，Basic Stamp，Atmel芯片, 或者什么平台, 如Microchip PIC, 你都应该能够理解并在你的项目中使用这些想法, 没人相信必须要有复杂的数学才能使用IMU单元(复杂的FIR或IIR滤波器, 例如Kalman滤波器, Parks-McClellan滤波器等)。你可以去研究所有的算法，并获得复杂并且出色的结果，我的解释方式只需要基本的数学知。我更相信简单。我认为简单的系统更容易控制和监视，另外很多嵌入式设备没有足够的动力和资源来实现复杂的需要用矩阵计算的算法。

我将以我设计的新IMU单元-Acc\_Gyro 加速度计+陀螺仪IMU为例。我们将在下面的示例中使用此设备的参数。本单元是一个很不错的设备，因为它包含3个设备：

– LIS331AL –模拟3轴2G加速度计

– LPR550AL –双轴（俯仰和横滚），500deg / s陀螺仪

– LY550ALH –单轴（偏航）陀螺仪（最后一个设备是在本教程中未使用，但是如果你继续实现DCM Matrix时，它变得很重要）

第一部分: 加速度计

为了理解这个单元, 我们将从加速度计开始。当我们理解加速度计时候把它想象成一个方盒子里面装了一个球，这通常很有用。你可能想象成其他的东西比如饼干或甜甜圈，但我会想象成一个球：

图2 一直处于失重状态

如果我们将这个盒子放在没有重力场的地方，或者没有其他可能影响球位置的场，那么球只是浮在盒子中间。你可以想象盒子位于远离任何宇宙物体的外太空中，如果很难找到这样的地方，可以想象至少一个宇宙飞船绕着行星运行，一切都处于失重状态。从图3可以看到我们为每个轴分配了一对墙面（我们删除了Y+轴，为了方便我们而已看到框内内容）。想象一下，每堵墙都是压敏的。如果我们突然将盒子向左移动（我们以1g=9.8m/s^2的加速度对其进行加速），则球撞击X-墙壁。然后测量施加到墙壁上的压力，并在X轴上输出值 -1g。

图3 向左9.8m^2/s加速

请注意，加速度计实际上会检测到与加速度矢量方向相反的力。这种力量通常被称为惯性力量或虚拟力量。你应该从中学到的一件事是，加速度计通过施加在其墙面之一上的力间接测量加速度（根据我们的模型，它可能是弹簧或现实生活中的其他加速度计）。该力可能是由加速度引起的，但我们将在下一个实例中看到的那样，它并不是由加速度引起的。

如果我们将模型放在地球上，球将落在Z- 墙壁上，并在墙壁底部施加1g的力，如下图4所示

图4 将模型放在地球上

在这种情况下，盒子不会移动，但我们在Z轴上仍获得-1g的度数。球施加在墙壁上的压力是由重力引起的。从理论上将，它可能是另一种类型的力，比如如果你想象我们的球是金属的，则在盒子旁边可以放一块磁铁可以使球移动，从而使球撞到另一墙面。这样说来这只是为了证明加速度计测量的是力而不是加速度。碰巧，加速度会导致惯性力，该惯性力会被加速度计的力量检测结构捕获。

虽然该模型并不是MEMS传感器的准确结构，但通常在解决与加速度计相关的问题时很有用。实际上，也有类似的传感器，它们内部有金属球，他们被称为倾斜开关，但是他们更为原始，通常只能分辨出设备是否在某个范围内倾斜，而不是倾斜程度。

到目前为止，我们已经分析了单轴加速度计的输出，这就是单轴加速度计的全部功能。三轴加速度计的真正价值在于它们可以检测所有三个轴上的惯性力。让我们回到盒子模型，然后将盒子向右旋转45度。球现在会碰到两个墙壁：Z-和X-，如图5

图5 向右旋转45度

0.71的值不是任意的，它们实际是1/2开方的近似值。当我们介绍加速度的下一个型号，你将更加清楚这个值。

在前面的模型中，我们固定了重力并旋转了想象出来的盒子。在最后的两个示例中，我们分析了两个不同盒子位置的输出，而矢量力保持恒定。虽然这对于理解加速度计与外力相互作用很有用，但如果将坐标系固定到加速度计的轴上并想象矢量力围绕我们旋转，则在实施计算时候更有用。

图6 盒子模型转移到坐标轴上

如上图6所示，我保留了轴的颜色，以便你可以从前面的模型过度到新的模型。试想一下，新的模型中的每个轴都分别垂直于之前模型中盒子的各个面。向量R是加速度计正在测量的向量力（可以是上述示例中的重力或惯性力，也可以是两者的组合）。Rx，Ry，Rz是R向量在X，Y，Z轴上的投影。注意有以下关系

（式1）

这基本相当于3D中的勾股定理。

还记得，我早些时候告诉你的 1/2开方约等于0.71不是随机的。如果将他们插入上述公式，回忆一下当我们的引力为1g后，我们可以验证：

将R = 1， 简单的带入（式1）中

经过漫长的理论介绍，我们越来越接近现实中的加速度计。Rx，Ry，Rz值实际上与你的真实加速度计输出值线型相关，你可以将其用于执行各种计算。

在这之前，让我们先谈谈加速度计将这些信息传递给我们的方式。大多数加速度计分为两类：数字和模拟。数字加速度计将使用I2C，SPI或UART等串行协议为你提供信息，而模拟加速度计将输出预定义范围内的电压点平，你必须使用ADC(模拟到数字转换器)模块将其转换位数字值。我不会详细介绍ADC的工作原理，一部分原因是ADC主题太广泛，另一部分原因是ADC平台间存在差异。一些微控制器将具有内置的ADC模块，最终都会得到一个在一定范围内的值。例如一个10位ADC模块将输出一个范围0～1023的值，请注意1023 = 210 - 1。12位的ADC模块将输出值在0~4095范围内，请注意，4095 = 212 - 1。

让我们继续来看一个简单的例子，家色号我们的10位ADC模块为我们提供三个加速度通道（三轴）的一下值：

AdcRx = 586

AdcRy = 630

AdcRz = 561

每个ADC模块都有一个参考的电压，在我们的示例中假设它是3.3V。要将10bit ADC值转化位电压，我们使用以下公式：

这里有个简短的注释：对于8位的ADC，最后的分频器将为255 = 28 - 1；对于12位的ADC，最后的分频器将为4096 = 212 - 1。

将此公式应用于所有3个通道，我们得到： (我们将所有的结果四舍五入到小数点后两位）

每个加速度计都偶一个0V电压电平，你可以在规格中找到它，这是对应于0g的电压。为了获得有正负的电压值，我们需要计算从该电平开始的偏移。假设我们的0g电压电平位VzeroG = 1.65。我们从0g电压计算出电压偏移，如下所示：

现在，我们的加速度计度数以伏特为单位，但仍未以g（9.8m/s^2）为单位，为了进行最终转换，我们应用了加速度计灵敏度，通常以mV/g表示。假设我们的灵敏度=478.5mV/g=0.4785。灵敏度值可以在加速度计规格中找到。为了获得以g表示的最终力值，我们使用以下公式：

我们当然可以将所有步骤组合到一个公式中，但是我自信研究了所有步骤，以明确说明如何从ADC读数转换位g表示矢量力的分量。

(式2)

现在，我们具有定义惯性矢量力的三个方向全部分量，如果设备不受重力以外的其他作用力，则可以假定这是重力矢量的方向。如果计算设备相对于地面有倾斜度，则可以计算该矢量和Z轴之间的角度。如果你还对每个轴的倾斜方向感兴趣，可以将此结果分为两个分量：X和Y轴上的倾斜度，可以计算为重力矢量与X/Y轴之间的角度。现在，我们已经计算了Rx，Ry，Rz的值，因此计算这些角度比你想象的要简单得多。让我们回到最后的加速度计模型，并做了一些附加的表示法：

图7 三维加速度计模型

我们感兴趣的角度是X ，Y，Z轴矢量力与R之间的角度。我们将这些角度定义位Axr，Ayr，Azr。你可以从R和Rx形成的直角三角形中注意到:

类似的:

我们可以从(式1)推到

现在，我们可以使用arccos() 函数（逆 cos() 函数）找到角度：

为了证明这些公式，我们花费了很长时间来解释，加速度计模型。根据你的应用程序，你可以希望使用我们得出的任何一个中间公式。我们还将很快介绍陀螺仪模型，并且将理解如何将加速度计和陀螺仪数据结合起来，以提供更准确的倾斜度估算。

但是在我们这样做之前，让我们做一些更有用的表示法：

这个三元组通常被成为Direction Cosine，它基本上表示与R向量具有相同方向的单位向量（长度为1的向量）。你可以轻松地验证：

这是一个很好的属性，因为它使我们免于监视R向量的摸（长度）。通常情况下，如果我们只对惯性矢量的方向感兴趣，为了简化其他计算，将其模量归一化是有意义的。

第二部分: 陀螺仪

我们不会像对加速度计那样引入任何等效的陀螺仪盒子模型，而是直接跳到第二个加速度模型，并展示陀螺仪根据该模型进行的测量。

图8 三轴陀螺仪模型

每个陀螺仪通道测量围绕一根轴的旋转。例如，两周陀螺仪将测量围绕（或者说“关于”）X轴和Y轴的旋转。为了用数字表示这种旋转，让我们做一些记号。首先让我们定义：

Rxz - 是惯性矢量力R在XZ平面上的投影

Ryz - 是惯性矢量力R在YZ平面上的投影

从Rxz和Rz形成的直角三角形，使用毕达哥拉斯定理，我们得到：

Rxz2 = Rx2 + Rz2，类似地：

Ryz2 = Ry2 + Rz2

还应该注意：

R2 = Rxz2 + Ry2，这可以从(式1)和以上公式得出，也可以从R和Ryz形成的直角三角形得出

R2 = Ryz2 + Rx2

在本文中我们将不适用这些公式，但是注意模型中所有值之间的关系会很有用。

相反，我们将定义Z轴与Rzy，Ryz向量之间的角度，如下所示，

Axz - 是Rxz（R在XZ平面上的投影）与Z轴之间的夹角

Ayz - 是Ryz（R在YZ平面上的投影）与Z轴之间的夹角

现在我们越来越接近陀螺仪的测量范围。陀螺仪 测量上面定义的是角度的变化率。换句话说，它将输出与这些角度的变化率线性相关的值。为了解释这一点，我们假设我们已经测量了在时间t0处绕Y轴的旋转角度（即Axz角），并将其定义位Axz0，接下来我们在稍后的时间t1处测量了该角度，他就是Axz1。变化率的计算方法如下：

如果我们用Axz表示角度，用秒表示时间，则该值将用度/秒表示。这是陀螺仪测量的。

实际上，陀螺仪（除非它是特殊的数字陀螺仪）很少会为您提供以度/秒为单位的值。与加速度计相同，你将获得一个ADC值，你需要使用类似于我们为加速度计定义的(式2)将其转换位度/秒。让我们为陀螺仪介绍ADC到度/秒的转换公式（假设我们使用的是10位的ADC模块，对于8为ADC将1023替换为255，对于12位ADC将替换为4095）。

(式3)

AdcGyroXZ，AdcGyroYZ - 是我们从adc模块获得的，它们代表分别测量R矢量在XZ上在YZ上面的投影旋转的通道，这相当于说分别绕Y轴和X轴旋转。

Vref - 是ADC参考电压，在下面的示例中，我们将使用3.3V的电压

VzeroRate - 是零速率电压，换句话说，陀螺仪在不旋转的情况下输出的电压，例如Acc\_Gyro板为1.23V（你可以在规格说明书中找到这个值，但不要相信规格，大多数陀螺仪在焊接后会遭受轻微的偏移，因此请使用电压表测量每个轴输出的VzeroRate，通常， 一旦陀螺仪焊接后，此值将不会随着时间变化，如果他发生变化，请编写校准程序，当测量设备启动时，必须指示用户在启动时将设备保持在静止以进行陀螺仪校准）。

灵敏度 - 陀螺仪的灵敏度，通常以mV /（deg / s）表示，通常写为mV / deg / s，它基本上能够告诉你，如果旋转速度提高一度，陀螺仪的输出将增加多少mV / s。Acc\_Gyro板的灵敏度例如为2mV / deg / s或0.002V / deg / s

我们来举个例子，假设我们的ADC模块返回以下值：

使用以上公式，并使用Acc\_Gyro板的specs的参数，我们将获得：

换句话说，设备绕Y轴旋转（或者可以说它在XZ平面中旋转）的速度为306度/秒，并且绕X轴旋转（或者可以说它的YZ平面中旋转）的速度为-94度/秒，请注意，负号表示设备沿与常规正方向相反的方向旋转。按照惯例，一个旋转方向为正。较好的陀螺仪规格表会告诉你哪个方向是正方向，否则你必须通过实验测试该设备并注意哪个旋转方向会导致输出的引脚上的电压增加，从而找到它的正方向。最好使用示波器完成这个操作，因为一旦停止旋转，电压就会降至零速率水平。如果你使用万用表，必须保持恒定的旋转速度至少几秒钟，并记下旋转期间的电压，然后将其与零速率电压进行比较。如果它大于零速率电压，则表示旋转方向为正。

第三部分: 将他们放在一起, 结合加速度和陀螺仪的数据

如果你正在阅读本文，则可能已经购买或打算购买IMU设备，或者可能单独打算虫单独的加速度计和陀螺仪设备构建一个。

注意：为实际实施和测试此算法，请阅读一下文章：

http://starlino.com/imu\_kalman\_arduino.html

使用结合了加速度计和陀螺仪的组合IMU设备的第一步是对其坐标系。最简单的方法是选择加速度计的坐标系作为参考坐标系。大多数加速度计数据表都会显示相对于物理芯片或者设备图像的X，Y，Z轴方向。例如，这是X，Y，Z轴的方向，如Acc\_Gyro板规格中所示：

图9 Acc\_Gyro板各轴正方向

下一步是：

- 确定上面讨论的RateAxz和RateAyz值相对应的陀螺仪输出。

- 确定由于陀螺仪相对于加速度计的物理位置是否需要相反的这些输出

不要假设陀螺仪的输出标记为X或Y，即使该输出是IMU单元的一部分，它也不会与加速度计坐标系中任何轴相对应。最好的方法是对其进行测试。

这是一个示例，用于确定陀螺仪的哪个输出对应于上面讨论的TateAxz值。

- 从设备水平放置开始。加速度计的X和Y输出都将输出零重力电压（例如，对于Acc\_Gyro板，此电压为1.65V）

- 接下来开始围绕Y轴旋转设备，另一种说法是，你在XZ平面中旋转设备，因此X和Z轴加速度计的输出发生变化，而Y轴输出保持不变。

- 当以恒定的速度旋转设备时，请注意陀螺仪输出会发生变化，而其他陀螺仪输出应该保持恒定

- 在绕Y轴（XZ平面旋转）旋转期间改变的陀螺仪输出将提供给AdcGyroXZ作为输入值，我们可以从中计算RateAxz

- 最后一步是确保旋转方向符合我们的模型，在某些情况下，由于陀螺仪相对于加速度计的物理位置，你可能需要反转RateAxz值

- 再次执行上述测试，围绕Y轴旋转设备，这一次监视加速度计的X轴输出，（在我们的模型中位AdcRx）。如果AdcRx增大（水平位置开始的第90度旋转），则AdcGyroXZ应该减小。这是由于以下事实：我们在监视重力矢量，并且当一个设备沿着一个方向旋转时，矢量将沿相反的方向旋转（相对于我们正在使用的设备协调系统）。因此，否则你需要反转RateAx，可以通过引入一个标志因数达到这个公式3，如下所示：

当InvertAxz 是 1 或 -1

通过围绕X轴旋转设备，可以对RateAyz进行相同的测试，你可以哪个确定陀螺仪输出对应于RateAyz，以及是否需要反转。获得InvertAyz的值后，应使用以下公式计算RateAyz：

如果你要在Acc\_Gyro板上进行这些测试，则会得到以下结果：

- RateAxz的输出引脚为GX4，InvertAxz = 1

- RateAyz的输出引脚为GY4，InvertAyz = 1

从这一点出发，我们将认为你已经设置了IMU，这样你就可以为可以为Axr，Ayr，Azr（如第一部分中定义的加速度计）和RateAxz，RateAyz（如第二部分定义的陀螺仪）计算正确的值。接下来我们将分析这些值之间的关系，这对于更准确的获得设备相对于地平面的倾斜度估计很有用。

到目前为止，你可能会问自己，如果加速度计模型已经给我们提供了Axr，Ayr，Azr的倾斜角度，为什么我们要混入陀螺仪数据呢？答案很简单：加速度传感器数据不能总是100%可信。有多种原因，请记住，加速度计会测量惯性力，这种力可能是由重力引起的（理想情况下仅由重力引起），但也可能是由设备的加速度（运动引起的）。结果是，即使加速度计处于相对稳定的状态，总体上它仍对震动和机械噪声非常敏感。这就是大多数IMU系统使用陀螺仪消除任何加速度计误差的主要原因。但是，这是怎么做的呢？陀螺仪是否没有噪音？

陀螺仪并非没有噪音，但是由于他测量旋转时对线型机械运动不太敏感，因此加速度计会遭受噪声的影响，但是陀螺仪还存在其他类型的问题，例如飘漂移（当旋转停止时不会回到零速率值）。尽管如此，通过对来自加速度计和陀螺仪的数据求平均，与仅使用加速度计数据相比，我们可以获得相对更好的设备倾斜度估计值。

在接下来的步骤中，我将介绍一种算法，该算法的灵感来自卡尔曼滤波器中使用的一些思想，但是，在嵌入式设备上实现它要简单得多，也要容易得多。在此之前，让我们先看一下我们希望算法计算的内容。那么，这是重力矢量的方向R = [Rx，Ry，Rz]，从中我们可以得出其他值，例如Axr，Ayr，Azr或cosX，cosY，cosZ，这将使我们对设备相对于地平面的倾斜度有所了解，我们将在第1部分中讨论这些值之间的关系。有人可能会说–我们是否已经从在第1部分的(式2)中获得了Rx，Ry，Rz这些值？是的，但是请记住，这些值仅来自加速度计数据，因此，如果要直接在应用程序中使用它们，则可能会收到比应用程序所能承受的更多的噪声。为避免更多的混乱，让我们重新定义加速度计的测量值，如下所示：

Racc –是由加速度计测量的惯性力矢量，由以下分量（在X，Y，Z轴上的投影）组成：

到目前为止，我们有一组测量值，可以完全从加速度计ADC值获得。我们将这组数据称为“向量”，并使用以下表示法。

因为可以从加速度计数据中获得Racc的这些分量，所以我们可以将其视为算法的输入。

请注意，由于Racc可以测量重力，因此，如果假设此向量的模定义如下或等于1g，那将是正确的。

但是，请确保按如下方式更新此向量是有意义的：

Racc（标准化过的）=

这将确保标准化Racc向量的模始终为1。

接下来，我们将介绍一个新的向量，我们将其称为

这将是我们算法的输出，这些是基于陀螺仪数据和过去估计数据的校正值。

我们的算法将执行以下操作：

– 加速度计告诉我们：“您现在处于Racc位置”

– 我们说“谢谢，但让我检查”，然后使用陀螺仪数据以及过去的Rest数据和我们输出一个新的估计向量Rest。

–我们认为Rest是关于设备当前位置的“最佳选择”。

让我们看看如何使它工作。

我们将通过信任加速度计并分配以下内容来开始序列：

顺便说一句，请记住Rest和Racc是向量，因此上述方程式只是编写3套方程式的一种简单方法，并且避免重复：

接下来，我们将在T秒的相等时间间隔进行常规测量，并获得新的测量值，这些测量值将定义为，，等。我们还将在每个时间间隔下Rest（1），Rest（2），Rest（3）等发布新的估算值。

假设我们在步骤n。我们要使用两组已知的值：

– 我们先前的估计，

– 我们当前的加速度计测量

在计算Rest（n）之前，让我们介绍一个新的测量值，该值可以从我们的陀螺仪和先前的估算中获得。

我们将其称为Rgyro，它也是一个由3个成分组成的向量：

我们将一次计算此向量的一个分量。我们将从RxGyro开始。

图10 陀螺仪模型

让我们从观察陀螺仪模型中的以下关系开始，从Rz和Rxz形成的直角三角形可以得出：

Atan2可能是你从未使用过的函数，它与atan相似，不同之处在于它返回（-PI，PI）范围内的值，而不是atan返回的（-PI / 2，PI / 2）范围，并且需要2个参数，而不是一个。它使我们能够将Rx，Rz的两个值转换为360度（-PI至PI）的整个角度。你可以在此处阅读有关atan2的更多信息。

因此，知道和，我们可以找到：

请记住，陀螺仪测量的是Axz角的变化率。因此我们可以如下估计新角度Axz（n）：

请记住，可以从我们的陀螺仪ADC读数中获得RateAxz。更精确的公式可以使用如下计算的平均转速：

我们可以找到相同的方式：

好了，现在我们有了Axz（n）和Ayz（n）。我们从哪里去扣除RxGyro / RyGyro？从等式 1我们可以写出矢量Rgyro的长度，如下所示：

另外，由于我们对Racc向量进行了归一化，因此我们可以假定它的长度为1，并且在旋转后没有变化，因此编写它是相对安全的：

让我们为以下计算采用一个临时的较短的符号：

使用以上关系，我们可以编写：

让我们将分数的分子和分母除以

请注意，

因此：

现在将根号中的分数的分子和分母乘以z2

注意

所以最后：

回到我们的符号，我们得到：

我们发现

现在，终于可以找到：

其中当RzGyro> = 0时Sign，而当时

一种简单的估算方法是

实际上，当RzEst（n-1）接近0时要小心。在这种情况下，您可以完全跳过陀螺仪相位并分配：Rgyro = Rest（n-1）。Rz用作计算Axz和Ayz角度的参考，当它接近0时，值可能会溢出并触发不良结果。您将在大浮点数域中，其中tan（）/ atan（）函数实现可能缺乏精度。

因此，让我们回顾一下到目前为止，我们处于算法的第n步，并且已经计算出以下值：

Racc - 来自我们的加速度计

Rgyro - 的当前读数来自Rest（n-1）和陀螺仪的当前读数

我们使用哪些值来计算更新的估计值Rest（n）？您可能猜到了我们会同时使用两者。我们将使用加权平均值，以便：

我们可以通过将分数的分子和分母都除以w1来简化此公式。

在替换之后，我们得到：

在上面的公式中，wGyro告诉我们与陀螺仪相比，我们对陀螺仪的信任程度。可以通过实验选择该值，通常在5到20之间的值会触发良好的结果。

该算法与卡尔曼滤波器的主要区别在于该权重相对固定，而在卡尔曼滤波器中，权重会根据测得的加速度计读数的噪声进行永久性更新。卡尔曼滤波器专注于为你提供“最佳”的理论结果，而该算法可以为你的实际应用提供“足够好”的结果。您可以实现一种算法，该算法可以根据你测量的某些噪声因素来调整wGyro，但是固定值对于大多数应用程序来说效果很好。

我们离获取更新的估计值仅一步之遥：

现在让我们再次规范化此向量：

我们准备再次重复循环。

注意：为实际实施和测试此算法，请阅读以下文章：

http://starlino.com/imu\_kalman\_arduino.html

有关加速度计和陀螺仪IMU Fusion的其他资源：

http://www.mikroquad.com/pub/Research/ComplementaryFilter/filter.pdf

http://stackoverflow.com/questions/1586658/combine-gyroscope-and-accelerometer-data

http://www.dimensionengineering.com/accelerometers.htm

**文献名称（外文）**

**作者：**

**起止页码：**

**出版日期（期刊号）：**

**出版单位：**

**外文原文：**

A Gyroscopic Data based Pedometer Algorithm

Sampath Jayalath

Department of Electrical and Computer Engineering Sri Lanka Institute of Information Technology

New Kandy Rd, Malabe, Sri Lanka adasdjsampath@gmail.com

Abstract-Accuracy of step counting is one of the main problems that exist in current Pedometers, especially when walking slowly on flat lands and performing different activities, such as climbing up and down stairs and walking on inclined planes. Although accelerometer based pedometers provide a reasonable accuracy when walking at higher speeds, the accuracy of them are not sufficient at slow walking speeds and performing different activities. This paper proposes a novel algorithm to detect steps using single-point gyroscopic sensors embedded in mobile devices. Preliminary analysis of data collected in different environments with the involvement of male and female volunteers indicated that gyroscope alone provides sufficient information necessary for accurate step detection. Algorithm was developed based on the gyroscopic data in conjunction with zero crossing and threshold detection techniques. The results proved that gyroscope based step detection algorithm provide a high accuracy when performing different activities and at slow paced walking.

Keywords-Pedometer algorithms; gyroscopic data; single­ point sensors; o-the-shelfdevices; mobile applications;

I.INTRODUCTION

Modern medical researches highlight that pedometers support not only to physical body but mental activities of human beings to a greater extent[I],[2],[3]. Low cost pedometers help to improve the motivation of the walker [4], indoor navigation, activity recognition and for various applications in the field of health care. Pedometers can be used to detect steps from vertical acceleration of the human body. This works under two systems of mechanism. One is of mechanical based and other being of the electrical based accelerometers. Modem pedometers are generally based on MEMS (micro-electromechanical systems) accelerometer, mostly I-axis, but the use of 2-axis and 3-axis accelerometers, gyroscopes and magnetometers improves precision and releases some utilization constraints e.g. positioning of the pedometer. The accuracy of these systems is at an acceptable level but not perfect due to various drawbacks [5]. Applications of pedometer are now upgraded and can be found in mobile devices. [t is obvious with the application of pedometers to mobile devices, has now improved the standards of healthcare applications.

The approaches of some pedometer algorithms proposed by researchers are discussed in the background section including their features and drawbacks.

978-1-4673-4463-0/13/$31.00 ©2013 IEEE 551

Nimsiri Abhayasinghe

Department of Electrica[ and Computer Engineering Sri Lanka Institute of Information Technology

New Kandy Rd, Malabe, Sri Lanka nimsiri.a@sliit.lk

II.BACKGROUND

S.E Crouter et al. [6] have compared the accuracy and reliability of 10 pedometers available in the market. These pedometers were based on mechanisms like, accelerometer, metal-on-metal and magnetic reed proximity switch. [t is important to notice that all the testing was done at normal walking speeds. Their conclusion was that accuracy of pedometers was highly subjective upon the internal mechanism and sensitivity. But they have failed to measure the accuracy of pedometers when walking slowly and performing different activities like ascending and descending stairs.

Comparative study with commercially available pedometers done by Jerome and Albright [7] has shown accuracies are poor with a minimum average absolute error value of 13%. Their conclusion was that none of the pedometers can be used for research purpose or general usage.

Wasiq Waqar et al. [8] have developed a pedometer based on accelerometer for their [ndoor Positioning System which consists of a preset threshold based peak detection method to identify a valid step and step cycle pattern detection method to discard invalid steps due to instantaneous readings of the accelerometer. [t should be noted that the results of the pedometer may change with different individual walking patterns and speeds due to preset threshold.

Melis Oner et al. [9] have implemented another step detecting algorithm for their Early detection of the falling event system. Step detection of this particular algorithm relies on the detecting peaks within a period in the data produced by the accelerometer sensor during walking. They were able to achieve higher accuracies during higher speeds of walking and with the mobile based pedometer placed fixed and loose in the pocket. However, their algorithms failed to count steps accurately during slow paced walking.

Mi-hee Lee et al. [10] were able to achieve 99% accuracy in their portable acceleration sensor module with some advanced processing like FFT, Fuzzy C and statistical calcu[ations. But they have agreed finally that their system doesn't process data in real time, inability to measure steps during activities [ike ascending and descending stairs walking and need for an efficient device to cary out processing.

A.M. Cavalcante et al. [[ 1] have developed a pedometer to be used with their research on Real-time indoor tracking on

SuC2.3

mobile devices. System compares mean values of acceleration samples in conjunction with a sliding window mechanism to detect steps. Their conclusion was that a proper sampling rate, sliding window and quality sensors embedded in mobile devices are a major requirement to detect steps accurately.

According to Garcia et al. [12] comparative study on the accuracies of mobile phone based pedometers with the commercially available pedometers concluded that mobile phones provide a competitive performance against the commercially available pedometers. Further both indicate less accuracy when at slower speed and high accuracy while in faster walking.

Lim et al. [13] have proposed a pedometer based on gyroscope but not discussed deeply on its practical usage. Zhong et al. [14] has also discussed pedometer based on accelerometer sensor attached to foot with processing done on a PDA and were able to achieve accuracies more than 90%. But none of these pedometers are inconvenient to be used with any application as sensors are attached to the body.

According to G. Boyce et al. [IS] comparative study on the accuracies of mobile phone based pedometer technologies which are freely available in the market, concluded that mobile phone based applications lag behind the use of a commercially available pedometer when determining step count. Further manipulation of settings is required to improve the accuracy of step counting for one activity level, but recalibration is required as intensity of activity changes.

M. Ayabe et al. [16] have conducted an important research on accuracies of pedometers during ascending and descending stairs. They have used one spring-levered and two piezo­ electric pedometers to test the accuracy. Their conclusion was that pedometers can assess the number of step accurately within an acceptable range of measurement error during the stair climbing activities at a stepping rate of 80 step·min-I or faster with 18 cm or higher stairs. However, it should be noted that they have highlighted the poor accuracy at slower stepping rates.

Most of the previous researches have identified that pedometers are less valid and reliable during slow walking speeds. This inaccuracy results from smaller vertical movements of the hip which are below the specific pedometer threshold value and are therefore not recorded. Another limitation is that pedometers do not contain an internal clock; therefore it is not possible to determine the intensity or duration of activity performed.

III.IMPLEMENTED ALGORITHM

The development of this algorithm is based on the experimental results of the research conducted by N. Abhayasinghe et al. on Indoor Positioning and Indoor Navigation [17]. This particular algorithm only uses gyroscopic data produced by the mobile device to predict steps. It can be described as follows;

A.

Initial concept behind the algorithm

Leg movement during walking shows a sinusoidal behavior. This behavior can be clearly identified by monitoring the angular velocity of the leg. Therefore one axis of the gyroscope provides the information about the movement of the leg depending on the orientation of the device. A small research on ways of placing the modern mobile devices on a pocket proved that almost all the users placed their devices vertically in the pocket. Therefore monitoring the gyroscopic x axis data is considered. It is important to notice for different orientations only variable that needed to change is the axis that we are obtaining data from the gyroscope.

B.Filtering raw data

Initially raw data are filtered using a proper filter that would preserve the properties of walking. A typical walking pace may be around 2 Hz, while running may be double and slow walking may be half of that. A choice of cutoff frequency that accommodates slow-paced activities is a major concern in achieving better accuracy. Therefore choice of the filter was a simple sixth order Butter worth low-pass filter having a cutoff frequency in the range of 0.9 Hz to 3 Hz. Fig. I depicts the raw gyro-x value and filtered version (2 Hz) of it.

C.Removal of unwanted signal components

Gyroscopic data takes a sinusoidal behavior after filtering for both steps and instantaneous movement of the device. A sample out technique is used to reduce wrong step counting due to instantaneous movement of the device. According to the experimental results a step occurs on an average of OAO to l.20 seconds depending on the intensity of walking. Therefore once a step is detected, the algorithm can be set to eliminate any signal that can be counted as a valid step before the average step time.

D. key features the filtered signal

The main idea for the step detection in this algorithm relies on detecting zero crosses in the data produced by the filtered gyro-x. Fig. 2 illustrates the consecutive zero crosses that occurs during a single stride. In addition to zero crossing an adaptive peak threshold is used to validate a step. An individual can train the algorithm to learn the minimum signal peak that can be used to realize a valid step, especially when moving slowly as possible and descending stairs. So this peak threshold is used as a parameter to validate a step. Threshold peak detection helps to avoid instantaneous and small movements of the device. Further this threshold peak can be adjusted so that the algorithm is capable of detecting steps when the device is placed in different pockets which are loose or tight (Signal strength differs with the environment where the device is placed). Fig. 2, Fig 3 and Fig 4 depict the strength of the gyro-x signal (Upward-peak signal) when walking (Average = 1.3rad/s), ascending (Average = 1.6rad/s) and descending (Average = 0.9rad/s) stairs for an individual. It is important to note that these averages vary with different patterns and intensities of walking.

Figure 1. Gyroscopic X Axis reading (Raw and Filtered Value at 2 Hz)

Figure 4. Strength of the gyro-x signal when descending stairs

Gyro Sensor X-axis Reading(Walking at Normal Speed)

E.Experimental test and result

Apple devices are used to test the Pedometer algorithm, which provide a higher data rate and accuracy when extracting data from the embedded sensor. The proposed algorithm was implemented on iPhone 4S and iPod 4G.

Testing of the algorithm was done with the involvement of

5 male and 5 female members with random heights and weights. Since this algorithm aimed to detect steps irrespective of age, sex and various physical aspect of human being, we were not much concerned of varieties of the people involved sex etc. When the application starts we assume that the initial orientation of the device is closer to zero degrees. Then users were instructed to perform different activities, such as climbing

up and down stairs and walking on inclined planes. Sample of

Figure 2. Strength of the gyro-x signal when walking at a medium pace

counted steps for an individual is tabulated in Table I.

TABLE 1. EXPERIMENTAL RESULTS OF THE PROPOSED ALGORITHM FOR AN INDTVUAL PERFORMING DIFFERENT ACTIVITIES

Gyro Sensor X-axis Reading(Ascending stairs)

individual who participate the testing process. Combined results were tabulated in Table II.

Figure 3. Strength of the gyro-x signal when ascending stairs

TABLE II. EXPERIMENTAL RESULTS OF THE PROPOSED ALGORITHM FOR ALL PARTICIPANTS

Overall accuracy of the algorithm was above 94%. It is evident from the results that climbing down stairs registered a low percentage of accuracy. This is mainly due to the weak signal strength by the user. Apart from that other errors were due to weak signal strength of different individuals at the start

Further this algorithm was tested for accuracy at different walking speeds and stepping speeds with the involvement of multiple male and female volunteers. Table III shows the variation of the accuracies at different walking speeds on a flat land while accuracies at different stepping speeds are tabulated in Table IV and Table V.

TABLE III. ACCUARACY OF THE ALGORITHM AT DIFFERENT WALKING SPEEDS

down. This is achieved by setting a proper filtering process along with an adaptive peak threshold set by the user. It is important to notice that accuracies of pedometers at slow speeds of activity intensities were a major requirement for both research activities and consumers of pedometers.

In addition to those, the algorithm depends only on the data of a single axis of the gyroscope. This provides lesser computations and ability to integrate this function in research areas like Indoor Navigation.

Further algorithm is implemented in currently available high end mobile devices, which is more convenient to be used with any application as sensors are not attached to the body.

G. Future Work

TABLE IV.

ACCUARACY OF THE ALGORITHM AT DIFFERENT STEPPING UP SPEEDS

Actual Calculated Accuracy

step counting as the environment that places the device changes. So designing a neural network that learns different thresholds with the changing environments can solve this inconvenience.

The error percentage of the algorithm is due to the fact that it is detecting the movement of the phone to validate a step. So we will be working on an improved solution to avoid such false

counts.

TABLE V.

ACCUARACY OF THE ALGORITHM AT DIFFERENT STEPPING DOWN SPEEDS

Actual Calculated Accuracy

IV.

CONCLUSIONS

It is evident from the results of past researches that most of the pedometer algorithms perform very poorly at slow speeds of walking (50-70 steps/min) and when performing different

activities at varying speeds. Therefore the need of developing a reliable algorithm which performs at any walking speed and with different walking patterns was a major requirement among modern researchers and consumers of pedometers. Key points of the algorithm presented in this paper consist of engineering techniques like noise removal, zero crossing and threshold detection. Further it makes use of the natural rotational motion of the leg of the human being to predict steps rather than the

linear motion of the body. Overall accuracy of the algorithm was above 94%, in which accuracies at slow speeds of walking was remarkably high. It is also important to notice that this accuracy can be further improved with proposed engineering techniques.

ACKNOWLEDGMENT

The authors would like to thank Dr.lain Murray from Curtin University, Australia for providing all hardware and software accessories required to accomplish this task and all the volunteer participants (students of Sri Lanka Institute of Information Technology, Malabe, Sri Lanka) for their participation in the experiments conducted.

REFERENCES

[1]AZ. LaCroix, S.G. Leveille, J.A. Hecht, L.c. Grothaus and E.H. Wagner, Does walking decrease the risk of cardiovascular disease hospitalizations and death in older adults?, Journal of the American Geriatrics Society, vol. 44, no.2, pp.113-120, Feb 1996.

[2]E.M. Simonsick, .l.M. Guralnik, S. Volpato, .I. Balfour, and L.P. Fried, Just get out of the door! Importance of walking outside the home for maintaining mobility: Findings from the women's health and aging study, Journal of the American Geriatrics Society, vol.53, no.2, pp.19S- 203, Feb 2005.

[3]Y. Hatano, Use of the pedometer for promoting daily walking exercise, International Council for Health, Physical Education, and Recreation, vol. 29, pp.4-S, 1993.

[4]E. Garcia, Hang Ding, A Sarela and M. Karunanithi, Can a mobile phone be used as a pedometer in an outpatient cardiac rehabilitation program?, in IEEE/ICME International Conference on Complex Medical Engineering (CME) 2010., 2010, pp.250-253.

[5]D.S. Michaud, E. Giovannucci, W.C. Willett, G.A. Colditz, M.l Stampfer and C.S. Fuchs, Physical activity, obesity, height, and the risk of pancreatic cancer, Journal of the American Medical Association, vol. 2S6, no.S, pp. 921-929, Aug 2001.

[6]S.E. Crouter, E. Scott, P.L. Schneider, M. Karabulut and D.R. Bassett, Validity of 10 electronics pedometers for measuring steps, distance and energy cost, Medicine and Science in Sports and Exercise, vol. 35, no.S, pp.1455-1460, 2003.

[7]G. .I. Jerome and C. Albright. (2011, June). Accuracy of Five Talking Pedometers under Controlled Conditions. The Journal of Blindness

Innovation and Research. [On-line]. 1(2). Available: www.nfb­ jbir.org/index.php/JBIR/article/view!l7/3S [Oct. 27, 2011].

[S] W. Waqar, A Vardy and Y. Chen. Motion Modelling using Smart phones for Indoor Mobilephone Positioning, presented at the 20th Newfoundland Electrical and Computer Engineering Conference, Newfoundland, Canada, 2011.

[9]M. Oner, J.A. Pulcifer-Stump, P. Seeling and T. Kaya. Towards the Run and Walk Activity Classification through Step Detection- An Androi Application,in 34th Annual International Conference of the IEEE Engineering in Medicine and Biology, 2012, pp.19S0-19S3.

[10]M. Lee, J. Kim, K. Kim, I. Lee, S.H. .lee and S.K. Yoo. Physical Activity Recognition Using a Single Tri-Axis Accelerometer, in Proceedings of the World Congress on Engineering and Computer Science, vol I, pp.14-17, October 2009.

[11]AM. Cavalcante, E.B. Souza, J.J. Bazzo, N. Bezerra, A Pontesand and RD. Vieira. A Pedometer-Based System for Real-time Indoor Tracking on Mobile Devices, in XXIX Brazilian Symposium on Telecommunications, Curitiba, Brazil, Oct,2011. [Online]. Available:

http://mwsl.unb.brlindex.php/pt/publicacoes [Apr. 12,2012].

[12]E. Garcia, H. Ding, A Sarela and M. Karunanithi, Can a mobile phone be used as a pedometer in an outpatient cardiac rehabilitation program?, in IEEE/ICME International Conference on Complex Medical Engineering (CME) 2010., 2010, pp.250-253.

[13]Y.P. Lim, LT. Brown and lC.T. Khoo, An accurate and robust gyroscope-gased pedometer, in 30th Annual International Conference of the IEEE Engineering in Medicine and Biology Society, 200S. EMBS 200S., 200S, pp.45S7-4590.

[14]S. Zhong, L. Wang, AM. Bernardos and M. Song, An accurate and adaptive pedometer integrated in mobile health application, in IET International Conference on Wireless Sensor Network, 2010. IET­ WSN., 2010, pp.7S-S3.

[15]G. Boyce, G. Padmasekara, M. Blum. Accuracy of Mobile Phone Pedometer Technology. Journal of Mobile Technology in Medicine, vol. I, pp.16-22, June 2012.

[16]M. Ayabe, l Aoki, K. Ishii, K. Takayama, H. Tanaka Pedometer accuracy during stair climbing and bench stepping exercises. Journal of Sports Science and Medicine, vol. 7, pp.249-254, June 200S.

[17]N. Abhayasinghe, I. Murray, A Novel Approach for Indoor Localization Using Human Gait Analysis with Gyroscopic Data, in International Conferenc on Indoor Positioning and Indoor Navigation,November 2012, [Online]. Available :

www.surveysurveying.unsw.edu.aulipin2012/papers.php [Dec.10,2012].

A Guide To using IMU (Accelerometer and Gyroscope Devices) in Embedded Applications.

POSTED ON DECEMBER 29, 2009 BY STARLINO

Introduction

There’s now a FRENCH translation of this article in PDF. Thanks to Daniel Le Guern!

This guide is intended to everyone interested in inertial MEMS (Micro-Electro-Mechanical Systems) sensors, in particular Accelerometers and Gyroscopes as well as combination IMU devices (Inertial Measurement Unit).

Example IMU unit:  Acc\_Gyro\_6DOF on top of MCU processing unit UsbThumb providing USB/Serial connectivity

I'll try try to cover few basic but important topics in this article:

– what does an accelerometer measure

– what does a gyroscope (aka gyro) measure

– how to convert analog-to-digital (ADC) readings that you get from these sensor to physical units (those would be g for accelerometer, deg/s for gyroscope)

– how to combine accelerometer and gyroscope readings in order to obtain accurate information about the inclination of your device relative to the ground plane

Throughout the article I will try to keep the math to the minimum. If you know what Sine/Cosine/Tangent are then you should be able to understand and use these ideas in your project no matter what platform you're using Arduino, Propeller, Basic Stamp, Atmel chips, Microchip PIC, etc. There are people out there who believe that you need complex math in order to make use of an IMU unit (complex FIR or IIR filters such as Kalman filters, Parks-McClellan filters, etc). You can research all those and achieve wonderful but complex results. My way of explaining things require just basic math. I am a great believer in simplicity. I think a system that is simple is easier to control and monitor, besides many embedded devices do not have the power and resources to implement complex algorithms requiring matrix calculations.

I'll use as an example a new IMU unit that I designed – the Acc\_Gyro Accelerometer + Gyro IMU. We'll use parameters of this device in our examples below. This unit is a good device to start with because it consists of 3 devices:

– LIS331AL (datasheet) – analog 3-axis 2G accelerometer

– LPR550AL (datasheet) – a dual-axis (Pitch and Roll), 500deg/second gyroscope

– LY550ALH (datasheet) – a single axis (Yaw) gyroscope (this last device is not used in this tutorial but it becomes relevant when you move on to DCM Matrix implementation)

Together they represent a 6-Degrees of Freedom Inertial Measurement Unit. Now that's a fancy name! Nevertheless, behind the fancy name is a very useful combination device that we'll cover and explain in detail below.

Part 1. Accelerometer

To understand this unit we'll start with the accelerometer. When thinking about accelerometers it is often useful to image a box in shape of a cube with a ball inside it. You may imagine something else like a cookie or a donut , but I'll imagine a ball:

If we take this box in a place with no gravitation fields or for that matter with no other fields that might affect the ball's position – the ball will simply float in the middle of the box. You can imagine the box is in outer-space far-far away from any cosmic bodies, or if such a place is hard to find imagine at least a space craft orbiting around the planet where everything is in weightless state . From the picture above you can see that we assign to each axis a pair of walls (we removed the wall Y+ so we can look inside the box). Imagine that each wall is pressure sensitive. If we move suddenly the box to the left (we accelerate it with acceleration 1g = 9.8m/s^2), the ball will hit the wall X-. We then measure the pressure force that the ball applies to the wall and output a value of -1g on the X axis.

Please note that the accelerometer will actually detect a force that is directed in the opposite direction from the acceleration vector. This force is often called Inertial Force or Fictitious Force . One thing you should learn from this is that an accelerometer measures acceleration indirectly through a force that is applied to one of it's walls (according to our model, it might be a spring or something else in real life accelerometers). This force can be caused by the acceleration , but as we'll see in the next example it is not always caused by acceleration.

If we take our model and put it on Earth the ball will fall on the Z- wall and will apply a force of 1g on the bottom wall, as shown in the picture below:

In this case the box isn't moving but we still get a reading of -1g on the Z axis. The pressure that the ball has applied on the wall was caused by a gravitation force. In theory it could be a different type of force – for example, if you imagine that our ball is metallic, placing a magnet next to the box could move the ball so it hits another wall. This was said just to prove that in essence accelerometer measures force not acceleration. It just happens that acceleration causes an inertial force that is captured by the force detection mechanism of the accelerometer.

While this model is not exactly how a MEMS sensor is constructed it is often useful in solving accelerometer related problems. There are actually similar sensors that have metallic balls inside, they are called tilt switches, however they are more primitive and usually they can only tell if the device is inclined within some range or not, not the extent of inclination.

So far we have analyzed the accelerometer output on a single axis and this is all you'll get with a single axis accelerometers. The real value of triaxial accelerometers comes from the fact that they can detect inertial forces on all three axes. Let's go back to our box model, and let's rotate the box 45 degrees to the right. The ball will touch 2 walls now: Z- and X- as shown in the picture below:

The values of 0.71 are not arbitrary, they are actually an approximation for SQRT(1/2). This will become more clear as we introduce our next model for the accelerometer.

In the previous model we have fixed the gravitation force and rotated our imaginary box. In last 2 examples we have analyzed the output in 2 different box positions, while the force vector remained constant. While this was useful in understanding how the accelerometer interacts with outside forces, it is more practical to perform calculations if we fix the coordinate system to the axes of the accelerometer and imagine that the force vector rotates around us.

Please have a look at the model above, I preserved the colors of the axes so you can make a mental transition from the previous model to the new one. Just imagine that each axis in the new model is perpendicular to the respective faces of the box in the previous model. The vector R is the force vector that the accelerometer is measuring (it could be either the gravitation force or the inertial force from the examples above or a combination of both). Rx, Ry, Rz are projection of the R vector on the X,Y,Z axes. Please notice the following relation:

R^2 = Rx^2 + Ry^2 + Rz^2     (Eq. 1)

which is basically the equivalent of the Pythagorean theorem in 3D.

Remember that a little bit earlier I told you that the values of SQRT(1/2) ~ 0.71 are not random. If you plug them in the formula above, after recalling that our gravitation force was 1 g we can verify that:

1^2 = (-SQRT(1/2) )^2 + 0 ^2 + (-SQRT(1/2))^2

simply by substituting R=1, Rx = -SQRT(1/2), Ry = 0 , Rz = -SQRT(1/2) in Eq.1

After a long preamble of theory we're getting closer to real life accelerometers. The values Rx, Ry, Rz are actually linearly related to the values that your real-life accelerometer will output and that you can use for performing various calculations.

Before we get there let's talk a little about the way accelerometers will deliver this information to us. Most accelerometers will fall in two categories: digital and analog. Digital accelerometers will give you information using a serial protocol like I2C , SPI or USART, while analog accelerometers will output a voltage level within a predefined range that you have to convert to a digital value using an ADC (analog to digital converter) module. I will not go into much detail about how ADC works, partly because it is such an extensive topic and partly because it is different from one platform to another. Some microcontroller will have a built-in ADC modules some of them will need external components in order to perform the ADC conversions. No matter what type of ADC module you use you'll end up with a value in a certain range. For example a 10-bit ADC module will output a value in the range of 0..1023, note that 1023 = 2^10 -1. A 12-bit ADC module will output a value in the range of 0..4095, note that 4095 = 2^12-1.

Let's move on by considering a simple example, suppose our 10bit ADC module gave us the following values for the three accelerometer channels (axes):

AdcRx = 586

AdcRy = 630

AdcRz = 561

Each ADC module will have a reference voltage, let's assume in our example it is 3.3V. To convert a 10bit adc value to voltage we use the following formula:

VoltsRx = AdcRx \* Vref / 1023

A quick note here: that for 8bit ADC the last divider would be 255 = 2 ^ 8 -1 , and for 12bit ADC last divider would be 4095 = 2^12 -1.

Applying this formula to all 3 channels we get:

VoltsRx = 586 \* 3.3V / 1023 =~ 1.89V (we round all results to 2 decimal points)

VoltsRy = 630 \* 3.3V / 1023 =~ 2.03V

VoltsRz = 561 \* 3.3V / 1023 =~ 1.81V

Each accelerometer has a zero-g voltage level, you can find it in specs, this is the voltage that corresponds to 0g. To get a signed voltage value we need to calculate the shift from this level. Let's say our 0g voltage level is VzeroG = 1.65V. We calculate the voltage shifts from zero-g voltage as follows::

DeltaVoltsRx = 1.89V – 1.65V = 0.24V

DeltaVoltsRy = 2.03V – 1.65V = 0.38V

DeltaVoltsRz = 1.81V – 1.65V = 0.16V

We now have our accelerometer readings in Volts , it's still not in g (9.8 m/s^2), to do the final conversion we apply the accelerometer sensitivity, usually expressed in mV/g. Lets say our Sensitivity = 478.5mV/g = 0.4785V/g. Sensitivity values can be found in accelerometer specifications. To get the final force values expressed in g we use the following formula:

Rx = DeltaVoltsRx / Sensitivity

Rx = 0.24V / 0.4785V/g =~ 0.5g

Ry = 0.38V / 0.4785V/g =~ 0.79g

Rz = 0.16V / 0.4785V/g =~ 0.33g

We could of course combine all steps in one formula, but I went through all the steps to make it clear how you go from ADC readings to a force vector component expressed in g.

Rx = (AdcRx \* Vref / 1023 – VzeroG) / Sensitivity (Eq.2)

Ry = (AdcRy \* Vref / 1023 – VzeroG) / Sensitivity

Rz = (AdcRz \* Vref / 1023 – VzeroG) / Sensitivity

We now have all 3 components that define our inertial force vector, if the device is not subject to other forces other than gravitation, we can assume this is the direction of our gravitation force vector. If you want to calculate inclination of device relative to the ground you can calculate the angle between this vector and Z axis. If you are also interested in per-axis direction of inclination you can split this result into 2 components: inclination on the X and Y axis that can be calculated as the angle between gravitation vector and X / Y axes. Calculating these angles is more simple than you might think, now that we have calculated the values for Rx,Ry and Rz. Let's go back to our last accelerometer model and do some additional notations:

The angles that we are interested in are the angles between X,Y,Z axes and the force vector R. We'll define these angles as Axr, Ayr, Azr. You can notice from the right-angle triangle formed by R and Rx that:

cos(Axr) = Rx / R , and similarly :

cos(Ayr) = Ry / R

cos(Azr) = Rz / R

We can deduct from Eq.1 that R = SQRT( Rx^2 + Ry^2 + Rz^2).

We can find now our angles by using arccos() function (the inverse cos() function ):

Axr = arccos(Rx/R)

Ayr = arccos(Ry/R)

Azr = arccos(Rz/R)

We've gone a long way to explain the accelerometer model, just to come up to these formulas. Depending on your applications you might want to use any intermediate formulas that we have derived. We'll also introduce the gyroscope model soon, and we'll see how accelerometer and gyroscope data can be combined to provide even more accurate inclination estimations.

But before we do that let's do some more useful notations:

cosX = cos(Axr) = Rx / R

cosY = cos(Ayr) = Ry / R

cosZ = cos(Azr) = Rz / R

This triplet is often called Direction Cosine , and it basically represents the unit vector (vector with length 1) that has same direction as our R vector. You can easily verify that:

SQRT(cosX^2 + cosY^2 + cosZ^2) = 1

This is a nice property since it absolve us from monitoring the modulus(length) of R vector. Often times if we're just interested in direction of our inertial vector, it makes sense to normalize it's modulus in order to simplify other calculations.

Part 2. Gyroscope

We're not going to introduce any equivalent box model for the gyroscope like we did for accelerometer, instead we're going to jump straight to the second accelerometer model and we'll show what does the gyroscope measure according to this model.

Each gyroscope channel measures the rotation around one of the axes. For instance a 2-axes gyroscope will measure the rotation around (or some may say about) the X and Y axes. To express this rotation in numbers let's do some notations. First let's define:

Rxz – is the projection of the inertial force vector R on the XZ plane

Ryz – is the projection of the inertial force vector R on the YZ plane

From the right-angle triangle formed by Rxz and Rz, using Pythagorean theorem we get:

Rxz^2 = Rx^2 + Rz^2 , and similarly:

Ryz^2 = Ry^2 + Rz^2

also note that:

R^2 = Rxz^2 + Ry^2 , this can be derived from Eq.1 and above equations, or it can be derived from right-angle triangle formed by R and Ryz

R^2 = Ryz^2 + Rx^2

We're not going to use these formulas in this article but it is useful to note the relation between all the values in our model.

Instead we're going to define the angle between the Z axis and Rxz, Ryz vectors as follows:

Axz – is the angle between the Rxz (projection of R on XZ plane) and Z axis

Ayz – is the angle between the Ryz (projection of R on YZ plane) and Z axis

Now we're getting closer to what the gyroscope measures. Gyroscope measures the rate of changes of the angles defined above. In other words it will output a value that is linearly related to the rate of change of these angles. To explain this let's assume that we have measured the rotation angle around axis Y (that would be Axz angle) at time t0, and we define it as Axz0, next we measured this angle at a later time t1 and it was Axz1. The rate of change will be calculated as follows:

RateAxz = (Axz1 – Axz0) / (t1 – t0).

If we express Axz in degrees, and time in seconds , then this value will be expressed in deg/s . This is what a gyroscope measures.

In practice a gyroscope(unless it is a special digital gyroscope) will rarely give you a value expressed in deg/s. Same as for accelerometer you'll get an ADC value that you'll need to convert to deg/s using a formula similar to Eq. 2 that we have defined for accelerometer. Let's introduce the ADC to deg/s conversion formula for gyroscope (we assume we're using a 10bit ADC module , for 8bit ADC replace 1023 with 255, for 12bit ADC replace 1023 with 4095).

RateAxz = (AdcGyroXZ \* Vref / 1023 – VzeroRate) / Sensitivity Eq.3

RateAyz = (AdcGyroYZ \* Vref / 1023 – VzeroRate) / Sensitivity

AdcGyroXZ, AdcGyroYZ – are obtained from our adc module and they represent the channels that measure the rotation of projection of R vector in XZ respectively in YZ planes, which is the equivalent to saying rotation was done around Y and X axes respectively.

Vref – is the ADC reference voltage we'll use 3.3V in the example below

VzeroRate – is the zero-rate voltage, in other words the voltage that the gyroscope outputs when it is not subject to any rotation, for the Acc\_Gyro board it is for example 1.23V (you can find this values in the specs – but don't trust the specs most gyros will suffer slight offset after being soldered so measure VzeroRate for each axis output using a voltmeter, usually this value will not change over time once the gyro was soldered, if it variates – write a calibration routine to measure it before device start-up, user must be instructed to keep device in still position  upon start-up for gyros to calibrate).

Sensitivity – is the sensitivity of your gyroscope it is expressed in mV / (deg / s) often written as mV/deg/s , it basically tells you how many mV will the gyroscope output increase , if you increase the rotation speed by one deg/s. The sensitivity of Acc\_Gyro board is for example 2mV/deg/s or 0.002V/deg/s

Let's take an example, suppose our ADC module returned following values:

AdcGyroXZ = 571

AdcGyroXZ = 323

Using the above formula, and using the specs parameters of Acc\_Gyro board we'll get:

RateAxz = (571 \* 3.3V / 1023 – 1.23V) / ( 0.002V/deg/s) =~ 306 deg/s

RateAyz = (323 \* 3.3V / 1023 – 1.23V) / ( 0.002V/deg/s) =~ -94 deg/s

In other words the device rotates around the Y axis (or we can say it rotates in XZ plane) with a speed of 306 deg/s and around the X axis (or we can say it rotates in YZ plane) with a speed of -94 deg/s. Please note that the negative sign means that the device rotates in the opposite direction from the conventional positive direction. By convention one direction of rotation is positive. A good gyroscope specification sheet will show you which direction is positive, otherwise you'll have to find it by experimenting with the device and noting which direction of rotation results in increasing voltage on the output pin. This is best done using an oscilloscope since as soon as you stop the rotation the voltage will drop back to the zero-rate level. If you're using a multimeter you'd have to maintain a constant rotation rate for at least few seconds and note the voltage during this rotation, then compare it with the zero-rate voltage. If it is greater than the zero-rate voltage it means that direction of rotation is positive.

Part 3. Putting it all together. Combining accelerometer and gyroscope data.

If you're reading this article you probably acquired or are planning to acquire a IMU device, or probably you're planning to build one from separate accelerometer and gyroscope devices.

NOTE: FOR PRACTICAL IMPLEMENTATION AND TESTING  OF THIS ALGORITHM PLEASE READ THIS ARTICLE:

http://starlino.com/imu\_kalman\_arduino.html

The first step in using a combination IMU device that combines an accelerometer and a gyroscope is to align their coordinate systems. The easiest way to do it is to choose the coordinate system of accelerometer as your reference coordinate system. Most accelerometer data sheets will display the direction of X,Y,Z axes relative to the image of the physical chip or device. For example here are the directions of X,Y,Z axes as shown in specifications for the Acc\_Gyro board:

Next steps are:

– identify the gyroscope outputs that correspond to RateAxz , RateAyz values discussed above.

– determine if these outputs need to be inverted due to physical position of gyroscope relative to the accelerometer

Do not assume that if a gyroscope has an output marked X or Y, it will correspond to any axis in the accelerometer coordinate system, even if this output is part of an IMU unit. The best way is to test it.

Here is a sample sequence to determine which output of gyroscope corresponds to RateAxz value discussed above.

– start from placing the device in horizontal position. Both X and Y outputs of accelerometer would output the zero-g voltage (for example for Acc\_Gyro board this is 1.65V)

– next start rotating the device around the Y axis, another way to say it is that you rotate the device in XZ plane, so that X and Z accelerometer outputs change and Y output remains constant.

– while rotating the device at a constant speed note which gyroscope output changes, the other gyroscope outputs should remain constant

– the gyroscope output that changed during the rotation around Y axis (rotation in XZ plane) will provide the input value for AdcGyroXZ, from which we calculate RateAxz

– the final step is to ensure the rotation direction corresponds to our model, in some cases you may have to invert the RateAxz value due to physical position of gyroscope relative to the accelerometer

– perform again the above test, rotating the device around the Y axis, this time monitor the X output of accelerometer (AdcRx in our model). If AdcRx grows (the first 90 degrees of rotation from horizontal position), then AdcGyroXZ should decrease. This is due to the fact that we are monitoring the gravitation vector and when device rotates in one direction the vector will rotate in oposite direction (relative to the device coordonate system, which we are using). So, otherwise you need to invert RateAxz , you can achieve this by introducing a sign factor in Eq.3, as follows:

RateAxz = InvertAxz \* (AdcGyroXZ \* Vref / 1023 – VzeroRate) / Sensitivity , where InvertAxz is 1 or -1

same test can be done for RateAyz , by rotating the device around the X axis, and you can identify which gyroscope output corresponds to RateAyz, and if it needs to be inverted. Once you have the value for InvertAyz, you should use the following formula to calculate RateAyz:

RateAyz = InvertAyz \* (AdcGyroYZ \* Vref / 1023 – VzeroRate) / Sensitivity

If you would do these tests on Acc\_Gyro board you would get following results:

– the output pin for RateAxz is GX4 and InvertAxz = 1

– the output pin for RateAyz is GY4 and InvertAyz = 1

From this point on we'll consider that you have setup your IMU in such a way that you can calculate correct values for Axr, Ayr, Azr (as defined Part 1. Accelerometer) and RateAxz, RateAyz (as defined in Part 2. Gyroscope). Next we'll analyze the relations between these values that turn out useful in obtaining more accurate estimation of the inclination of the device relative to the ground plane.

You might be asking yourself by this point, if accelerometer model already gave us inclination angles of Axr,Ayr,Azr why would we want to bother with the gyroscope data ? The answer is simple: accelerometer data can't always be trusted 100%. There are several reason, remember that accelerometer measures inertial force, such a force can be caused by gravitation (and ideally only by gravitation), but it might also be caused by acceleration (movement) of the device. As a result even if accelerometer is in a relatively stable state, it is still very sensitive to vibration and mechanical noise in general. This is the main reason why most IMU systems use a gyroscope to smooth out any accelerometer errors. But how is this done ? And is the gyroscope free from noise ?

The gyroscope is not free from noise however because it measures rotation it is less sensitive to linear mechanical movements, the type of noise that accelerometer suffers from, however gyroscopes have other types of problems like for example drift (not coming back to zero-rate value when rotation stops). Nevertheless by averaging data that comes from accelerometer and gyroscope we can obtain a relatively better estimate of current device inclination than we would obtain by using the accelerometer data alone.

In the next steps I will introduce an algorithm that was inspired by some ideas used in Kalman filter, however it is by far more simple and easier to implement on embedded devices. Before that let's see first what we want our algorithm to calculate. Well , it is the direction of gravitation force vector R = [Rx,Ry,Rz] from which we can derive other values like Axr,Ayr,Azr or cosX,cosY,cosZ that will give us an idea about the inclination of our device relative to the ground plane, we discuss the relation between these values in Part 1. One might say – don't we already have these values Rx, Ry , Rz from Eq.2 in Part 1 ? Well yes, but remember that these values are derived from accelerometer data only, so if you would be to use them directly in your application you might get more noise than your application can tolerate. To avoid further confusion let's re-define the accelerometer measurements as follows:

Racc – is the inertial force vector as measured by accelerometer, that consists of following components (projections on X,Y,Z axes):

RxAcc = (AdcRx \* Vref / 1023 – VzeroG) / Sensitivity

RyAcc = (AdcRy \* Vref / 1023 – VzeroG) / Sensitivity

RzAcc = (AdcRz \* Vref / 1023 – VzeroG) / Sensitivity

So far we have a set of measured values that we can obtain purely from accelerometer ADC values. We'll call this set of data a vector and we'll use the following notation.

Racc = [RxAcc,RyAcc,RzAcc]

Because these components of Racc can be obtained from accelerometer data , we can consider it an input to our algorithm.

Please note that because Racc measures the gravitation force you'll be correct if you assume that the length of this vector defined as follows is equal or close to 1g.

|Racc| = SQRT(RxAcc^2 +RyAcc^2 + RzAcc^2),

However to be sure it makes sense to update this vector as follows:

Racc(normalized) = [RxAcc/|Racc| , RyAcc/|Racc| , RzAcc/|Racc|].

This will ensure the length of your normalized Racc vector is always 1.

Next we'll introduce a new vector and we'll call it

Rest = [RxEst,RyEst,RzEst]

This will be the output of our algorithm , these are corrected values based on gyroscope data and based on past estimated data.

Here is what our algorithm will do:

– accelerometer tells us: You are now at position Racc

– we say Thank you, but let me check,

– then correct this information with gyroscope data as well as with past Rest data and we output a new estimated vector Rest.

– we consider Rest to be our best bet as to the current position of the device.

Let's see how we can make it work.

We'll start our sequence by trusting our accelerometer and assigning:

Rest(0) = Racc(0)

By the way remember Rest and Racc are vectors , so the above equation is just a simple way to write 3 sets of equations, and avoid repetition:

RxEst(0) = RxAcc(0)

RyEst(0) = RyAcc(0)

RzEst(0) = RzAcc(0)

Next we'll do regular measurements at equal time intervals of T seconds, and we'll obtain new measurements that we'll define as Racc(1), Racc(2) , Racc(3) and so on. We'll also issue new estimates at each time intervals Rest(1), Rest(2), Rest(3) and so on.

Suppose we're at step n. We have two known sets of values that we'd like to use:

Rest(n-1) – our previous estimate, with Rest(0) = Racc(0)

Racc(n) – our current accelerometer measurement

Before we can calculate Rest(n) , let's introduce a new measured value, that we can obtain from our gyroscope and a previous estimate.

We'll call it Rgyro , and it is also a vector consisting of 3 components:

Rgyro = [RxGyro,RyGyro,RzGyro]

We'll calculate this vector one component at a time. We'll start with RxGyro.

Let's start by observing the following relation in our gyroscope model, from the right-angle triangle formed by Rz and Rxz we can derive that:

tan(Axz) = Rx/Rz => Axz = atan2(Rx,Rz)

Atan2 might be a function you never used before, it is similar to atan, except it returns values in range of (-PI,PI) as opposed to (-PI/2,PI/2) as returned by atan, and it takes 2 arguments instead of one. It allows us to convert the two values of Rx,Rz to angles in the full range of 360 degrees (-PI to PI). You can read more about atan2 here.

So knowing RxEst(n-1) , and RzEst(n-1) we can find:

Axz(n-1) = atan2( RxEst(n-1) , RzEst(n-1) ).

Remember that gyroscope measures the rate of change of the Axz angle. So we can estimate the new angle Axz(n) as follows:

Axz(n) = Axz(n-1) + RateAxz(n) \* T

Remember that RateAxz can be obtained from our gyroscope ADC readings. A more precise formula can use an average rotation rate calculated as follows:

RateAxzAvg = ( RateAxz(n) + RateAxz(n-1) ) / 2

Axz(n) = Axz(n-1) + RateAxzAvg \* T

The same way we can find:

Ayz(n) = Ayz(n-1) + RateAyz(n) \* T

Ok so now we have Axz(n) and Ayz(n). Where do we go from here to deduct RxGyro/RyGyro ? From Eq. 1 we can write the length of vector Rgyro as follows:

|Rgyro| = SQRT(RxGyro^2 + RyGyro^2 + RzGyro^2)

Also because we normalized our Racc vector, we may assume that it's length is 1 and it hasn't changed after the rotation, so it is relatively safe to write:

|Rgyro| = 1

Let's adopt a temporary shorter notation for the calculations below:

x =RxGyro , y=RyGyro, z=RzGyro

Using the relations above we can write:

x = x / 1 = x / SQRT(x^2+y^2+z^2)

Let's divide numerator and denominator of fraction by SQRT(x^2 + z^2)

x = ( x / SQRT(x^2 + z^2) ) / SQRT( (x^2 + y^2 + z^2) / (x^2 + z^2) )

Note that x / SQRT(x^2 + z^2) = sin(Axz), so:

x = sin(Axz) / SQRT (1 + y^2 / (x^2 + z^2) )

Now multiply numerator and denominator of fraction inside SQRT by z^2

x = sin(Axz) / SQRT (1 + y^2  \* z ^2 / (z^2 \* (x^2 + z^2)) )

Note that z / SQRT(x^2 + z^2) = cos(Axz) and y / z = tan(Ayz), so finally:

x = sin(Axz) / SQRT (1 + cos(Axz)^2 \* tan(Ayz)^2 )

Going back to our notation we get:

RxGyro = sin(Axz(n)) / SQRT (1 + cos(Axz(n))^2 \* tan(Ayz(n))^2 )

same way we find that

RyGyro = sin(Ayz(n)) / SQRT (1 + cos(Ayz(n))^2 \* tan(Axz(n))^2 )

Side Note: it is possible to further simplify this formula. By dividing both parts of the fraction by sin(Axz(n)) you get:

RxGyro =  1  / SQRT (1/ sin(Axz(n))^2  + cos(Axz(n))^2 / sin(Axz(n))^2  \* tan(Ayz(n))^2 )

RxGyro =  1  / SQRT (1/ sin(Axz(n))^2  + cot(Axz(n))^2  \* sin(Ayz(n))^2  / cos(Ayz(n))^2 )

now add and substract     cos(Axz(n))^2/sin(Axz(n))^2   = cot(Axz(n))^2

RxGyro =  1  / SQRT (1/ sin(Axz(n))^2  –  cos(Axz(n))^2/sin(Axz(n))^2   + cot(Axz(n))^2  \* sin(Ayz(n))^2  / cos(Ayz(n))^2  + cot(Axz(n))^2 )

and by grouping  terms 1&2 and then 3&4  we get

RxGyro =  1  / SQRT (1  +   cot(Axz(n))^2 \* sec(Ayz(n))^2 ),      where  cot(x) = 1 / tan(x)  and  sec(x) = 1 / cos(x)

This formula uses only 2 trigonometric functions and can be computationally less expensive. If you have Mathematica program you can verify it

by evaluating    FullSimplify [Sin[A]^2/ ( 1 + Cos[A]^2  \* Tan[B]^2)]

Now, finally we can find:

RzGyro  =  Sign(RzGyro)\*SQRT(1 – RxGyro^2 – RyGyro^2).

Where Sign(RzGyro) = 1 when RzGyro>=0 , and Sign(RzGyro) = -1 when RzGyro<0.

One simple way to estimate this is to take:

Sign(RzGyro) = Sign(RzEst(n-1))

In practice be careful when RzEst(n-1) is close to 0. You may skip the gyro phase altogether in this case and assign:  Rgyro = Rest(n-1). Rz is used as a reference for calculating Axz and Ayz angles and when it's close to 0, values may overflow and trigger bad results. You'll be in domain of large floating point numbers where tan() / atan() function implementations may lack precision.

So let's recap what we have so far, we are at step n of our algorithm and we have calculated the following values:

Racc – current readings from our accelerometer

Rgyro – obtained from Rest(n-1) and current gyroscope readings

Which values do we use to calculate the updated estimate Rest(n) ? You probably guessed that we'll use both. We'll use a weighted average, so that:

Rest(n) = (Racc \* w1 + Rgyro \* w2 ) / (w1 + w2)

We can simplify this formula by dividing both numerator and denominator of the fraction by w1.

Rest(n) = (Racc \* w1/w1 + Rgyro \* w2/w1 ) / (w1/w1 + w2/w1)

and after substituting w2/w1 = wGyro we get:

Rest(n) = (Racc + Rgyro \* wGyro ) / (1 + wGyro)

In the above formula wGyro tells us how much we trust our gyro compared to our accelerometer. This value can be chosen experimentally usually values between 5..20 will trigger good results.

The main difference of this algorithm from Kalman filter is that this weight is relatively fixed , whereas in Kalman filter the weights are permanently updated based on the measured noise of the accelerometer readings. Kalman filter is focused at giving you the best theoretical results, whereas this algorithm can give you results good enough for your practical application. You can implement an algorithm that adjusts wGyro depending on some noise factors that you measure, but fixed values will work well for most applications.

We are one step away from getting our updated estimated values:

RxEst(n) = (RxAcc + RxGyro \* wGyro ) / (1 + wGyro)

RyEst(n) = (RyAcc + RyGyro \* wGyro ) / (1 + wGyro)

RzEst(n) = (RzAcc + RzGyro \* wGyro ) / (1 + wGyro)

Now let's  normalize this vector again:

R = SQRT(RxEst(n) ^2 + RyEst(n)^2 +  RzEst(n)^2 )

RxEst(n) = RxEst(n)/R

RyEst(n) = RyEst(n)/R

RzEst(n) = RzEst(n)/R

And we're ready to repeat our loop again.

NOTE: FOR PRACTICAL IMPLEMENTATION AND TESTING  OF THIS ALGORITHM PLEASE READ THIS ARTICLE:

http://starlino.com/imu\_kalman\_arduino.html

Other Resources on Accelerometer and Gyroscope IMU Fusion:

http://www.mikroquad.com/pub/Research/ComplementaryFilter/filter.pdf

http://stackoverflow.com/questions/1586658/combine-gyroscope-and-accelerometer-data

http://www.dimensionengineering.com/accelerometers.htm

|  |
| --- |
| 指导教师意见：  翻译两篇与计步器设计相关的英文文献，译文能够较准确地表达原文中的语义，术语翻译正确，且语句较通顺，翻译工作量饱满，上传附件中的译文文档符合学校毕业设计的要求。  指导教师签字：  2020年3月18日 |
| 系(教研室)意见：  同意  主任签字：徐英卓  2020年6月6日 |

注：此表单独作为一页。