同济大学
TONGJI UNIVERSITY

传感器与检测技术课程设计

——GPS

姓　　名：李勇奇

学　　号：1531620

二〇一五年十二月

目录

第1章　引言

GPS(Global Positioning System) 是全球定位系统，是由美国国防部研制建立的一种具有全方位、全天候、全时段、高精度的卫星导航系统，能为全球用户提供低成本、高精度的三维位置、速度和精确定时等导航信息，是卫星通信技术在导航领域的应用典范，它极大地提高了地球社会的信息化水平，有力地推动了数字经济的发展。

本文主要专注于 GPS 模型块的使用，包括 GPS 数据的读取与解析，数据的滤波，以及利用 GPS 数据计算运动的距离，与地图相连接等，最终将上述功能结合在一起利用 Python 语言编写了一个具有图形界面的应用程序。

第2章　GPS 数据解析与应用

由于手头的 GPS 是已经封装好的模块，所以本文主要侧重于 GPS 模块软件方面的应用，包括对 GPS 模块传回的数据的解析、Kalman 滤波以及利用 GPS 计算移动的距离。

2.1　GPS 数据格式

GPS 模块上电之后每隔一段事件都会从串口传回如下固定格式的数据：

```
$GPGGA,235949.042,0000.0000,N,00000.0000,E,0,00,,0.0,M,0.0,M,,0000*45
$GPGLL,0000.0000,N,00000.0000,E,235949.042,V,N*47
$GPGSA,A,1,,,,,,,,,,,,,*1E
$GPGSV,1,1,00*79
$GPRMC,235949.042,V,0000.0000,N,00000.0000,E,,,140209,,,N*7E
$GPVTG,,T,,M,,N,,K,N*2C
```

该数据格式符合 NMEA-0813x 协议，NMEA 0183 是美国国家海洋电子协会（National Marine Electronics Association ）为海用电子设备制定的标准格式。目前业已成了 GPS 导航设备统一的 RTCM（Radio Technical Commission for Maritime services）标准协议。其数据类型主要有以下六种：

- GPGSV：可见卫星信息
- GPGLL：地理定位信息

- GPRMC：推荐最小定位信息
- GPVTG：地面速度信息
- GPGGA：GPS 定位信息
- GPGSA：当前卫星信息

为了获得 GPS 的时间、经度、纬度和地面速率，查阅协议可以发现所要获得的数据可以从 GPRMC 和 GPGGA 提取到，以 GPRMC 数据为例，其数据标准格式如下：

```
\$GPRMC,<1>,<2>,<3>,<4>,<5>,<6>,<7>,<8>,<9>,<10>,<11>,<12>*hh
```

其中：
<1> UTC 时间，hhmmss(时分秒) 格式
<2> 定位状态，A= 有效定位，V= 无效定位
<3> 纬度 ddmm.mmmm(度分) 格式 (前面的 0 也将被传输)
<4> 纬度半球 N(北半球) 或 S(南半球)
<5> 经度 dddmm.mmmm(度分) 格式 (前面的 0 也将被传输)
<6> 经度半球 E(东经) 或 W(西经)
<7> 地面速率 (000.0 999.9 节，前面的 0 也将被传输)
<8> 地面航向 (000.0 359.9 度，以真北为参考基准，前面的 0 也将被传输)
<9> UTC 日期，ddmmyy(日月年) 格式
<10> 磁偏角 (000.0 180.0 度，前面的 0 也将被传输)
<11> 磁偏角方向，E(东) 或 W(西)
<12> 模式指示 (仅 NMEA01833.00 版本输出，A= 自主定位，D= 差分，E= 估算，N= 数据无效)
根据上述信息，要获得时间、经度、纬度和速率信息只要提取相应位的数据以及相应的单位转换。

其他数据类型此处不再详细介绍，参考相关数据手册即可。

2.2 Kalman 滤波

为了更准确的获得经度纬度信息，采用 Kalman 滤波算法对 GPS 返回的经度纬度和地面速率进行滤波。Kalman 滤波算法是一种最优化自回归数据处理算

法，其原理和具体公式推导此处不详细描述，直接给出其时间和状态的更新方程。

- 时间更新方程：

$$\hat{\mathbf{X}}_k^- = \mathbf{A}\hat{\mathbf{X}}_{k-1} + \mathbf{B}\hat{\mathbf{U}}_{k-1} \tag{1}$$

$$\mathbf{P}_k^- = \mathbf{A}\mathbf{P}_{k-1}\mathbf{A}^T + \mathbf{Q} \tag{2}$$

- 状态更新方程：

$$\mathbf{K}_k = \mathbf{P}_k^-\mathbf{H}^T(\mathbf{H}\mathbf{P}_k^- + \mathbf{R})^{-1} \tag{3}$$

$$\hat{\mathbf{X}}_k = \hat{\mathbf{X}}_k^- + \mathbf{K}_k(\mathbf{Z}_k - \mathbf{H}\hat{\mathbf{X}}_k) \tag{4}$$

$$\mathbf{P}_k = (\mathbf{I} - \mathbf{K}_k\mathbf{H})\mathbf{P}_k^- \tag{5}$$

上述时间和状态更新方程是对于多维变量的矩阵形式，由于本文所需要的数据较为简单，控制变量 $\mathbf{U} = 0$，且认为经度纬度和速度并不相关，则将上述 Kalman 更新方程简化为一维形式。

- 时间更新方程：

$$\hat{x}_k^- = \hat{x}_{k-1} \tag{6}$$

$$p_k^- = p_{k-1} + Q \tag{7}$$

$$\tag{8}$$

- 状态更新方程：

$$k_k = p_k^-/(p_k^- + R) \tag{9}$$

$$\hat{x}_k = \hat{x}_k^- + k_k(z_k - \hat{x}_k) \tag{10}$$

$$p_k = (1 - k_k)p_k^- \tag{11}$$

其中 z 表示观测变量，$\hat{x}$ 表示状态变量的后验估计，$\hat{x}^-$ 表示状态变量的先验估计，$\hat{p}$ 表示误差的后验估计，$\hat{p}^-$ 表示误差的先验估计，Q 表示误差方差，R 表示测量方差。简单来说，以经度为例，GPS 返回的经度值就是 z_k，经过 Kalman 滤波，返回滤波后的经度值是 $\hat{x}_k$。

2.3 距离解算

根据 GPS 返回的数据可以计算运动路径中两点的距离，利用返回的时间、经度、纬度和地面速率信息可以想到两种计算距离的方法。较为简单的是利用速率对时间积分，因为 GPS 每个一小段时间 Δt_k 都会返回一个速率 v_k，则距离的可由以下公式计算得到

$$D_{ij} = \sum_{k=i}^{j} \Delta t_k \times v_k \tag{12}$$

另一种计算距离的方法可以通过经度纬度信息得到，计算公式如下：

$$D_{ij} = \sum_{k=i}^{j} \sqrt{(rcos(\frac{\alpha_{k-1} + \alpha_k}{2})(\beta_k - \beta_{k-1}))^2 + (r(\beta_k - \beta_{k-1}))^2} \tag{13}$$

其中 α、β 分别表示纬度和经度，r 表示地球半径。其原理就是利用微积分的思想计算球面上两点的距离。

第 3 章　软件模块设计

软件模块主要完成的功能是实时显示经纬度、运动距离等信息，并在在地图上实时显示位置，主要包括四个模块：

- GPS 模块
- Kalman 滤波模块
- map 模块
- GUI 模块

软件模块整体的构架如图1所示。其中虚线框内为所设计的软件模块，圆形 GPS 表示实际的 GPS，Map servers 表示提供地图服务的服务器，如 Google 的地图服务和百度的地图服务。矩形框内表示四个软件模块，Kalman 模块提供一个 Kalman 滤波器方便 GPS 模块调用；GPS 模块主要从实际的 GPS 中读取数据，然后对数据进行解析，并提供计算运动距离的方法，最后调用 Kalman 滤波器对数据进行滤波；Map 模块主要负责通过互联网和提供地图服务的服务器进行交互，比如可以调用百度地图 api 向其服务器请求指定经纬度的静态图，然后向 GUI 模块提供指定经纬度的静态图；GUI 模块负责整个程序图形界面的显示，以及将各个模块整合在一起。具体各个模块的实现细节见下文。

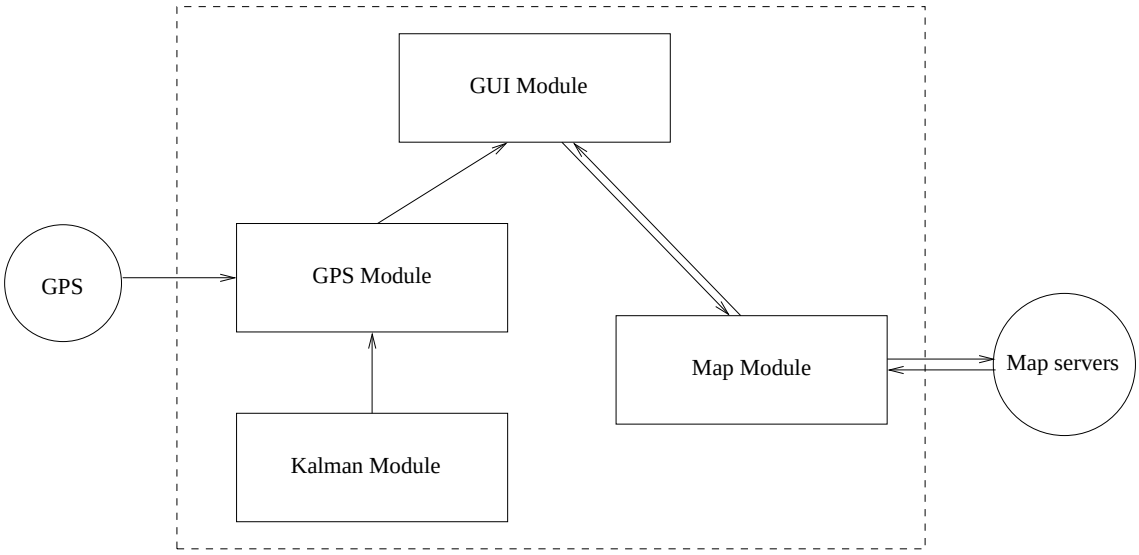

图 1: 软件模块整体构架

3.1 开发环境

软件的开发是在 Linux 下使用 Python 语言编写的，具体信息如表1所示。选择 Python 作为开发语言一方面是因为时间比较紧，而 Python 作为一门相当

表 1: 软件开发环境

操作系统	Ubuntu 14.04.3 LTS x86_64
程序语言	Python 2.7.6
编辑器	Emacs 24.5
版本控制	git 2.6.4

高级的语言相比 C 有较高的开发效率；另一方面，Python 是一门面向对象语言，而且第三方库相当丰富，写 GUI 程序较为方便。Python 的高效的开发效率的后果就是程序运行速率慢，但是对于本次实验并没有影响。

3.2 具体实现

对于 GPS 数据解析模块，其逻辑较为简单，其基本内容包括从 GPS 中读取原始数据，然后从原始数据中解析所需要的数据，然后解算运动的距离，再用 Kalman 滤波对数据滤波一下，然后提供给 GUI 模块。由于只要获取时间、经度、纬度和地面速率四个量，所以只要处理 GPS 返回数据中的 $GPRMC 信息即可，然后提取相应的时间、经度、纬度和地面速率，然后计算运动距离，最后

再用 Kalman 对数据滤波。其程序流程图如图2所示。调用该 GPS 数据解析函

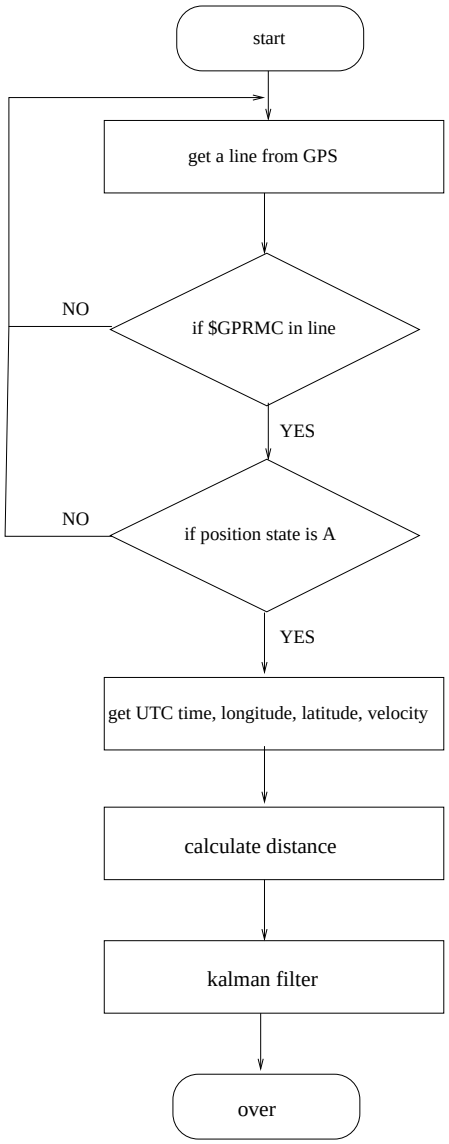

图 2: GPS 数据解析流程图

数一次就返回一次有效定位的时间、经度、纬度、地面速率值和运动距离。

对于 Kalman 滤波模块，为了增加该模块的复用和可扩展性，将其写成了一个类，其核心部分就是第二章中五个更新公式，其更新的算法如算法1所示。

对于 map 模块，其内部逻辑是首先从 GUI 模块出获得经度纬度信息，然后根据经度纬度信息，以及在地图服务端设置的 ak 和 sk 信息计算 sn 值，然后向地图服务端发送指定经度纬度的静态图，等待地图服务器传回来的图片，最后将图片返回给 GUI 模块。

对于 GUI 模块，其实现的主要功能是根据当前经度纬度等信息在图形界面

算法 1: 一维 Kalman 滤波数据更新算法

Input: 观察值 z_k

Output: 滤波后的观察值 $\hat{z}_k$

初始化参数 Q、R;

for *k=1; k<N;* **do**

// 时间更新;

$\hat{x}_k^- = \hat{x}_{k-1}$;

$p_k^- = p_{k-1} + Q$;

// 状态更新;

$k_k = p_k^-/(p_k^- + R)$;

$\hat{x}_k = \hat{x}_k^- + k_k(z_k - \hat{x}_k)$;

$p_k = (1 - k_k)p_k^-$;

// 输出;

$\hat{z}_k = \hat{x}_k$;

实时显示以及向 map 模块发送请求获得当前经度纬度的静态图。该模块较为复杂，包含 GUI 相关部分，并且涉及到相关模块之间的通信。GUI 部分采用 PyQt 框架，PyQt 是一个创建 GUI 应用程序的工具包，它是 Python 编程语言和 Qt 库的成功融合，Qt 是比较有名的跨平台 GUI 库，因此采用 PyQt 框架一方面可以高效的开发出 GUI 程序，另一方面所编写的程序也是跨平台的，只要计算机安装好 Python 和 Qt，所编写的程序就可以在其上运行。

第 4 章　实验结果

所编写软件的运行画面如图3所示。图中第一个下拉组合框可选择 GPS 的端口号，第二个下拉组合框选择波特率，按钮“open gps”根据选择的端口号和波特率打开对应端口的 GPS 设备，成功打开 GPS 后按钮标签将显示为“close gps”，另外一个“calcaulate distance”用来开始/停止计算距离。文本框显示相关信息表示当前 GPS 是否定位。五个标签动态显示当前的 UTC 时间、速度、经度、纬度和距离信息。地图是根据当前经度纬度信息向地图服务器请求的指定经度纬度的静态图，程序中每 1 秒向服务器请求一次，也就是说该地图每一秒刷新一次。

为了展现一下 Kalman 滤波的效果，滤波前和滤波后纬度值如图4所示，该

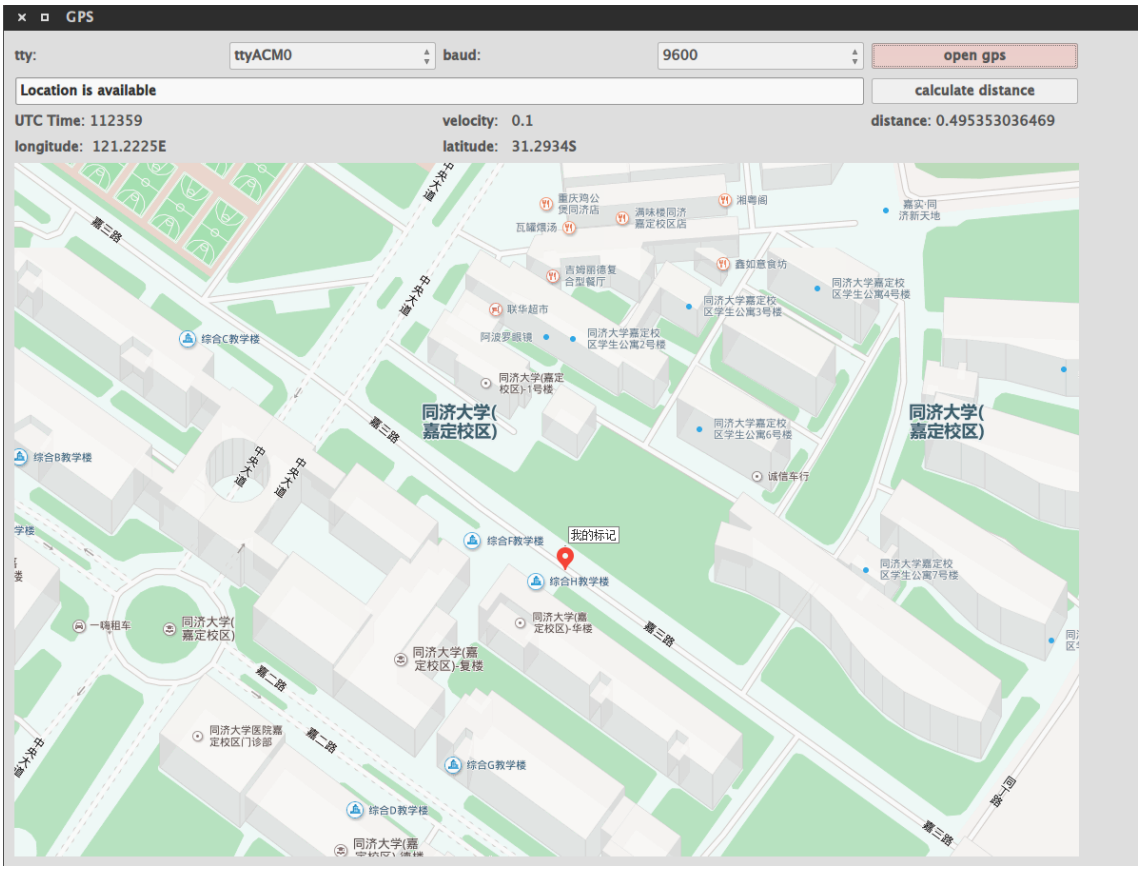

图 3: 软件运行界面

纬度信息是从华楼走到十七号楼 GPS 记录的纬度信息。从图4可以看出 Kalman 滤波后的纬度比滤波前相对平滑些，由此可见 Kalman 滤波有一定的作用。

第 5 章　结论

本文在 Linux 平台上利用 Python 语言编写了一个基于 GPS 的应用程序，程序完成了对 GPS 数据的读取、解析、滤波和应用，能够实时显示当前位置、速度以及运动距离等信息，并在地图上显示当前位置。通过实际运行程序，展现了其可顺利运行，并实现相关功能。此外，对比 Kalman 滤波前后的数据，展现了 Kalman 滤波良好的效果。

致谢

在此论文完成之际，衷心感谢苏永清老师的认真教导与帮助。

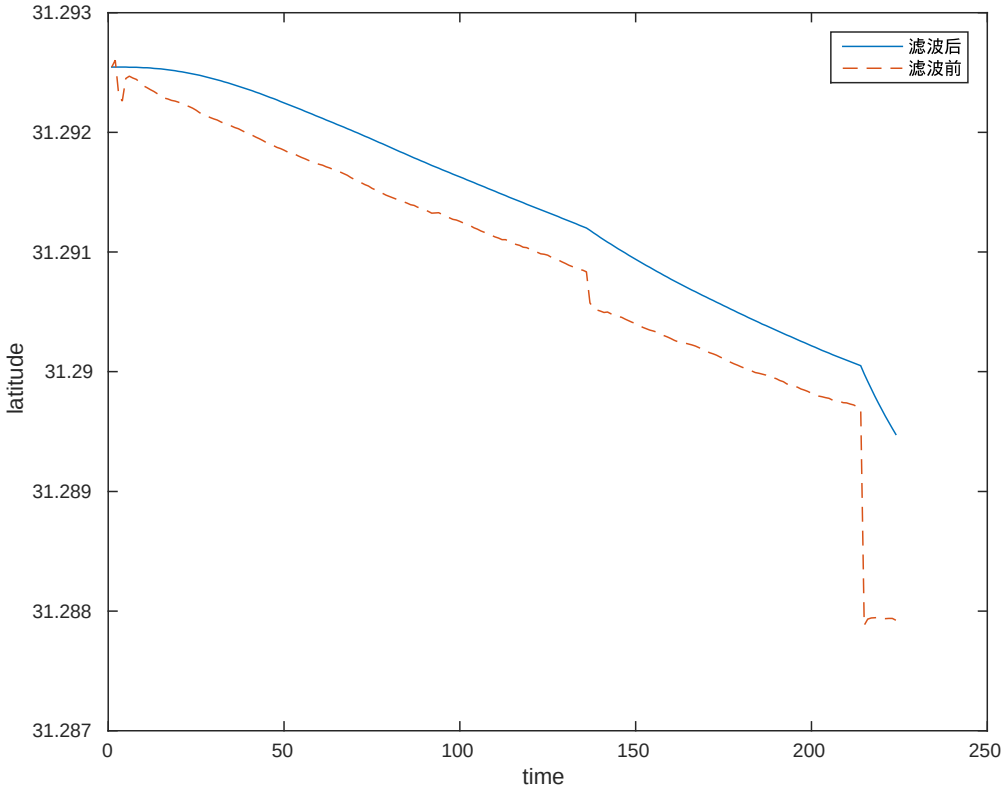

图 4: Kalman 滤波前后纬度值