華中科技大學

软件课程设计报告 (个人)

室内行人定位轨迹的校正方法与可视化系统的设计与开发

院　　系　电子信息与通信学院

专　　业　XXXXXXX

班　　级　XXXXXX

姓　　名　凌雪依

学　　号　UXXXXXXXXX

指导教师　XXX

日　　期　2023 年 3 月 18 日

2023 年 3 月 18 日

目　　录

1 项目描述

1.1 项目背景

1.1.1 室内定位

随着联网移动设备的流行，以及交通的快速发展，人们对个人定位的需求越来越高。基于位置的服务也有广泛的应用，并被广泛应用于导航、追踪和导游等方面。的关键技术是导航定位，包括室外到室内的定位。传统的全球定位系统和移动通信等技术方法都在室外环境中有着较为成熟的算法和较高的精度，但是其在室内环境中的表现则不尽人意。近几十年来，室内定位技术也在不断发展进步，但是各种技术都各有优缺，如何保证行人定位的准确度和连续性，是室内定位技术研究的重点。

随着智能设备的普及，基于位置的服务 (LBS) 成为人们日常生活中必不可少的一部分。现代人有 80% 的时间是在室内度过的。如何在室内环境中准确地确定用户的位置，成为近年来的一个热门研究课题。

在室外空间，以 GPS、北斗为代表的卫星定位系统能够满足人们在户外开阔环境中的定位需求; 但在室内环境中，由于建筑结构、室内障碍物对卫星信号的阻挡和反射，卫星定位系统会产生较大的定位误差。

因此，研究者们提出了多种基于无线信号的室内定位方案，利用的信号包括 Wi-Fi、蓝牙、超声波、超宽带、惯性传感器、红外线、声信号等。

1.1.2 行人航位推算

在行人定位领域 [7]，行人航位推算 (PDR[2]) 是一种主流方法。传统的 PDR 算法依赖惯性测量单元 (Inertia Measurement Unit，IMU) 收集行人的加速度、角速度等信息，进而推算行人的运动轨迹，见图 1-1。

$$\begin{aligned} x_t &= x_{t-1} + l_{t-1} \times \cos\theta_{t-1} \\ y_t &= y_{t-1} + l_{t-1} \times \sin\theta_{t-1} \end{aligned} \tag{1-1}$$

PDR 算法主要包括步频检测、步长估计、航向估计三个任务。

1.1.3 基础 PDR 算法

- 步频检测: 波峰测频法

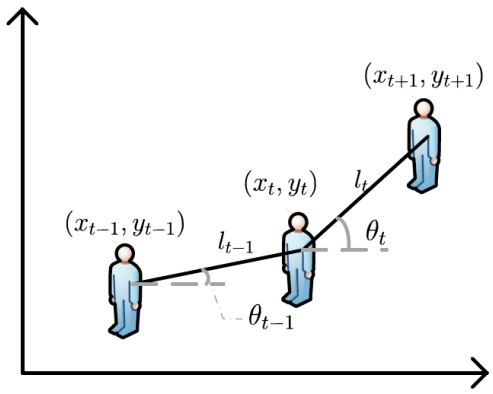


图 1-1　PDR 示意图

被测人员走动且手持终端时，随着人的周期性运动，三轴加速度传感器的波形也会产生周期性的变化，见图 1-2，其中以竖直方向（Z 轴）上的加速度变化最为明显。当一个有效波峰出现，即可认为移动了一步。按此原理即可估计出行人的步频。

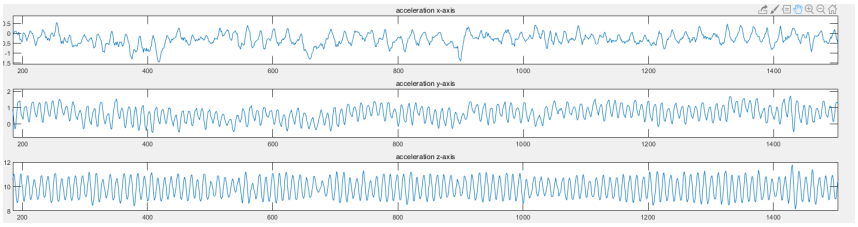


图 1-2　三轴加速度传感器的波形

- 步长估计:Weinberg 算法
  步长估计利用 Weinberg 算法实现，与最大加速度和最小加速度相关。

$$l_n = K \times \left(a_n^{max} - a_n^{min}\right)^{\frac{1}{4}} \tag{1-2}$$

  其中，l_n 是第 n 步的步长；a_n^{max} 和 a_n^{min} 分别表示第 n 步的 Z 轴加速度最大值和最小值；K 为常数。
- 航向估计: 角速度积分
  航向估计通常使用陀螺仪和电子罗盘这两种传感器。我们这里以基于陀螺仪的航向估计算法为例。陀螺仪可以测量人运动时的三轴角速度，对角速度进行积分即可计算出运动航向的变化角度值。如果已知航向的初始值，就可以计算出当前时刻的绝对航向。

1.1.4　定位效果评价指标

评价一个定位算法的定位精度，我们通常有如下几个指标:

1. 平均定位误差：计算所有测试点的定位误差（估计位置与真实位置之间的物理距离），求平均值。

2. 误差累积分布曲线（CDF）：各测试点定位误差的累计分布函数，它的横坐标表示定位误差，对应的纵坐标表示误差小于这个值的测试点的出现的频率。即

$$F_E(x) = P(E \leq x)$$

3. 百分位误差
   百分位误差指在 CDF 图中，纵坐标取一定百分比时，对应横坐标表示的定位误差。我们常用的百分位误差有 50% 误差，75% 误差，90% 误差等。

1.2 系统描述

1.2.1 系统功能

基本要求

- 合理设计网页布局，包括：室内地图区、信息显示区、定位指标计算区等；设计一个菜单作为初始界面。
- 在室内地图区，需要呈现室内平面图、行人真实轨迹和定位轨迹。用户能够通过点击定位轨迹中的点，查看不同位置的传感器信息（三轴加速度、角速度等）和行人航位信息（位移和航向），并在信息显示区显示。
- 在定位指标计算区，需要实时计算并展示当前定位轨迹的平均定位误差、百分位定位误差、CDF 曲线等定位指标。
- 系统需通过数据库实现信息的存储与交互。
- 定位轨迹至少需要实现两种
  - 直接定位点的轨迹
  - 使用一个直接定位点作为初始点，实现 PDR 的基础算法来估计行人轨迹。
- 软件系统可以通过上传 csv 文件（格式见数据文件），实时计算 PDR 算法的定位结果，绘制定位轨迹，显示定位指标。

1.2.2 系统架构

该软件的 系统架构为 B/S 架构，前端负责网页设计，后端负责编写 API 接口，调用设计的算法并利用数据库中的数据并来实现对室内行人定位轨迹的校正，并在网页上实现轨迹的可视化。后端通过 API 接口平台，测试对应的 API 接口; 前端利用 API 接口，测试网页显示内容是否正确，交互是否成功。算法模型通过 CDF 曲线以及平均定位误差来判断优劣; 联调测试: 前后端联调，观察软件是否达到理想的效果，是否满足了所有的用户需求。

1.3 模型描述

1.3.1 前后端交互模型

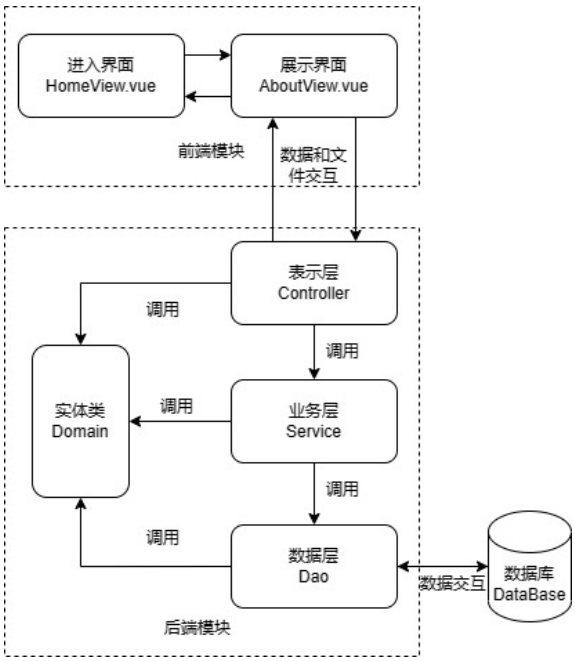


图 1-3 前后端交互模型

1.3.2 算法模型

波峰检测

我们称波宽是一个波峰的周期，最简单的波峰检测是按照固定波宽的大小进行匹配，如果一个点比一个波宽内的点都高，那么就认为他是一个波峰。在本次问题中，我们会发现波峰的周期性并没有和很好，不同组的数据存在一些差别，而且会有伪波峰的存在，既存在波宽不相同的问题，也存在单调性不那么强，因此直接匹配的方法不可取。经过文献查找，查询到一种高效且鲁棒性较强的波峰检测算法，即 AMPD 算法 (自动多尺度峰值查找算法) [3]，这个方法的优势是：（1）算法本身参数较少，对信号具有良好的自适应性，唯一的假设是信号的波峰是周期的或者准周期的（在我们本次的项目中，可以近似认为）；（2）抗噪能力强，对周期性的要求也不是很高。原理上来说：首先计算一个寻找局部最大值图，即对于不同的周期，来匹配寻找极大值，并且记录不同的周期对应极大值个数，对于个数最多的，我们可以认为是周期最明显的主要部分，在这里，即代表人移动时造成的波峰。我们找到局部最大值最多的作为窗口长度（window length）在窗口长度下，匹配出来的局部最大值认为是波峰。因为原论文里面有随机数，且计算较复杂。另找到一篇改进的方法，于是在此基础上进行修改 [4] 开始效果不错，但是在测试集上效果欠佳，因为波峰不明显，出现了平顶波峰，会影响窗口长度 = 3 的局部极大值图。于是对代码进行了修改，增加了一个参数，为最大值长度，如果识别的最

大值对应的窗口长度过大，会自动降低到阈值，此阈值是根据采样周期近似为 100ms，在平持状态设置为 3，在摇摆状态设置为 5，之后波峰检测效果很好，基本不会漏步。

步长估计

步长估计的方法主要有两种，第一种是对加速度求积分算出行进距离，第二种是利用加速度本身的峰峰值来计算。这里需要用到是否静止的信息来代表人的速度。但是在本问题中，有一组只有两个 stay = 1 是静止的点，很难利用速度信息。因此我们采取别的第二种方法进行处理。我们采取的是 Weinberg 算法计算方法见公式 (1-2)。K 通过近似估算，取了 0.45。在波峰检测的时候，我们对波谷和波峰都进行了检测。我们根据波峰和波谷之间的峰峰值，即可计算对应步长，我们考虑实际情况，在转弯的情况下，步长有时候会有较大浮动，因此我们对每个峰峰值单独计算步长，而不是求平均值计算一个整体的步长，平均的方法更加理想化，其实可以通过指数记忆的方式进行平均，这样既减少了步长的高频误差，同时也能考虑实际每一步本身的特性，本质还是一个混合滤波。

航向估计

航向估计是 PDR 算法中也是各种遥感、飞行器航迹等最难的一部分，对此的研究也较为深入，包括很多种方法。有欧拉角法、方向余弦法、三角函数法、Rodrigues 参数法、四元数法、等效旋转矢量法、投影向量法等各种方法。本次课程设计主要研究并实现了 Rodrigues 参数法、分解向量法、四元数等方法。

Rodrigues 参数法（旋转矩阵） 此法是本人在研究时独立发现出来的，最后查询资料时发现已经有此方法。Rodrigues 数法是法国数学家 Rodrigues 在 1840 年提出的，该方法所描述的姿态是唯一的，并且简洁、直观的优点，其微分方程结构简单，但 Rodrigues 参数法存在奇异值的缺陷，会造成一些不稳定的情况。在本次作业中会出现在失真的现象。

因为三轴角速度是可以进行矢量合成的，合成后在新的旋转轴上进行旋转。我们记录物体坐标系到地理坐标系的转换矩阵，开始时两者相同，但是每次旋转后就有所区别，因为每个时间间隔都有一次角速度，我们这里就认为每个时隙之间进行了一次旋转，因为我们要计算的航向是相对于地理坐标系的航向，而对于我们来说，我们知道航向在物体坐标系的方向向量（比如平持始终是 $(-1,0,0)$，所以我们只要知道两个坐标系之间的变换矩阵，即可得到在地理坐标系的方向向量，从而得到航向角。而变换矩阵的含义是向量 $\vec{v}$ 在坐标系 A 的坐标是 $\vec{v}_1$，在坐标系 B 的坐标是 $\vec{v}_2$，那么 $\vec{v}_2 = \mathbf{C}_{\mathbf{A}}^{\mathbf{B}} \times \vec{v}_1$，其中 $\mathbf{C}_{\mathbf{A}}^{\mathbf{B}}$ 是由坐标系 A 到坐标系 B 的变换矩阵。因为变换矩阵是一个正交矩阵，因此由坐标系 B 到坐标系 A 的变换矩阵 $\mathbf{C}_{\mathbf{B}}^{\mathbf{A}} = \mathbf{C}_{\mathbf{A}}^{\mathbf{B}\top}$ 。

为了得到从起始到第 i 个时刻对应的坐标系旋转矩阵，因为旋转是可以累计的，所以计算出每个时隙的旋转矩阵，累乘得到的就是从起始到第 i 个时刻的旋转矩阵。这个方法的数学推导有一些复杂，但可以查到有三维空间绕任意轴旋转的公式。注意到，固定坐标系，向量绕旋转轴旋转和固定向量，坐标系绕旋转轴旋转是完全一样的，唯一的区别就是角度要取相反数。因此

参考向量绕坐标轴旋转 [1]，通过对公式进行修改，角度倒置，可以计算得到从初始时刻到各时刻的旋转矩阵和逆矩阵。（旋转矩阵是一个正交矩阵，它的转置也是它的逆）并且根据对于前三组来说，航向始终是物体坐标系的 $(-1,0,0)$。记录前 n 个变换矩阵的逆连乘，对应 $\mathbf{C_B^E}$ 是从物体坐标系到地理坐标系的变换矩阵。然后再把物体坐标系的航向乘 $\mathbf{C_B^E}$ 得到物体坐标系的航向。对于后三组来说，认为开始手表的位置给出，对应的方向向量相对手表 $(-1,0,0)$。即手臂自然垂下。那么物体坐标系下的航向也不确定。那么每一个 timeslot 需要计算一次变换矩阵。利用上一次 timeslot 的航向和变换矩阵求出下一次的航向。那么地理坐标系的航向只要把物体坐标系的航向乘 $\mathbf{C_B^E}$ 即可得到方向，此时得到的是物体坐标系下的手表的航向，因为手表可能有垂直于地面的速度（因为人的摆臂）。所以要投影到 XOY 平面上，所以还要对向量进行投影并且归一化得到真正的方向向量。

分解向量法 分解向量法是根据方法本身的一个称呼，方法的来源是 [6]，并没有搜索到这个方法的学名。方法的原理是，因为我们只需要航向角，所以等效只需要地理坐标系的 z 轴角速度。我们首先定义竖直方向的轴叫做 z 轴。在忽略人的加速度的影响的时候，即我们认为加速度只有重力加速度，那么三轴加速度显示就是重力加速度在三轴上的投影。

理论上，我们并不需要复杂的姿势估计，只需要一个物理坐标系下的真实 z 轴角速度就可以。在波峰处，因为 $g >> a_{\square}$，可以近似认为和加速度就是 g。所以此时的三轴加速度就是地理坐标系下 $(0,0,-g)$ 在物体坐标系下的投影。所以对于三轴加速度 $(a1,a2,a3)$，有 $a1 = vx * -g, a2 = vy * -g, a3 = vz * -g$ 其中 (vx,vy,vz) 是物体坐标系的三个向量。所以 $-g = a1 * vx + a2 * vy + a3 * vz$。所以对于对于 (gx,gy,gz) 也有 $gz = g1 * vx + g2 * vy + g3 * vz$，因此我们可以通过计算得到地理坐标系下的 z 轴角速度，从而得到航向。但是这个方法有一个比较强的前提，就是加速度近似为 g，移动幅度越大，结果越差。

四元数法 四元数是英国数学家 W.R.Haminlton 在 1843 年引入的。但直到 20 世纪 60 年代末期都没有得到实际应用，随着空间技术的发展，数学家发现，四元数可以用在表示三维空间的旋转上，之后四元数才得到了广泛的应用。求解四元数微分方程要解四个微分方程。比解欧拉微分方程多一个方程，但计算量小、精度高、可避免奇异性，该方法是目前研究的重点之一。目前的主流观点表明四元数法具有最佳的性能。

四元数推到得到姿态角的过程十分繁琐，这里提供一部分推导过程参考自 [5]。

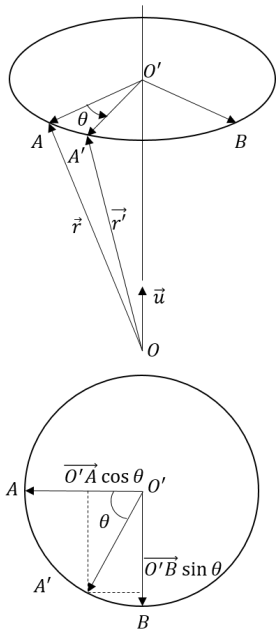


图 1-4　旋转参考图

如图 1-4所示, 设参考坐标系 R, 坐标轴为 x_R, y_R, z_R, 坐标轴所对应的单位向量记为 $\vec{i}_R, \vec{j}_R, \vec{k}_R$。有一刚体相对 R 系做定点转动, 记定点为 O。建立与刚体固连的动坐标系 b, 坐标轴为 x_b, y_b, z_b, 坐标轴所对应的单位向量记为 $\vec{i}_b, \vec{j}_b, \vec{k}_b$。初始时刻 R 系与 b 系重合。取刚体上一点 A，记其相对于定点的位置矢量为 $\vec{OA} = \vec{r}$。经过时间 t 后运动到 A' 的位置上，此时的位置矢量为 $\vec{OA'} = \vec{r'}$。根据欧拉定理，仅考虑 0 时刻到 t 时刻的角位置时，从 A 到 A' 的转动可以等效为绕瞬轴 $\vec{u}$ 转动 θ 角。为了计算方便，我们认为 $\vec{u}$ 为单位向量。这样，位置矢量做圆锥运动，A 和 A' 在同一个圆上，圆心为 O'，并且 $\vec{OO'}$ 与瞬轴 $\vec{u}$ 同向。

我们在圆 O' 上取一点 B 使得 $\angle AO'B = 90°$，根据右图，可以写出如下的几何关系：

$$\begin{cases} \vec{OO'} = (\vec{r} \cdot \vec{u})\vec{u} \\ \vec{O'A} = \vec{r} - \vec{OO'} = \vec{r} - (\vec{r} \cdot \vec{u})\vec{u} \\ \vec{O'B} = \vec{u} \times \vec{r} \\ \vec{O'A'} = \vec{O'A}\cos\theta + \vec{O'B}\sin\theta = \cos\theta\vec{r} - \cos\theta(\vec{r} \cdot \vec{u})\vec{u} + \sin\theta(\vec{u} \times \vec{r}) \end{cases}$$

之后经过代数化简，可以得到其坐标变换矩阵 C_B^E 为：

$$\mathbf{C_B^E} = I + 2\sin\frac{\theta}{2}\cos\frac{\theta}{2}U + 2\sin^2\frac{\theta}{2}U \cdot U \tag{1-3}$$

令 $q_0=\cos\dfrac{\theta}{2},q_1=l\sin\dfrac{\theta}{2},q_2=m\sin\dfrac{\theta}{2},q_3=n\sin\dfrac{\theta}{2}$，则

$$\begin{aligned}\vec{Q} &= q_0+q_1\vec{i}_R+q_2\vec{j}_R+q_3\vec{k}_R=\cos\frac{\theta}{2}+(l\vec{i}_R+m\vec{j}_R+n\vec{k}_R)\sin\frac{\theta}{2}\\ &= \cos\frac{\theta}{2}+\vec{u}^R\sin\frac{\theta}{2}\end{aligned}\tag{1-4}$$

将 $\mathbf{C}_{\mathbf{B}}^{\mathbf{E}}$ 写成矩阵形式有

$$\mathbf{C}_{\mathbf{B}}^{\mathbf{E}}=\begin{bmatrix}1-2(q_2^2+q_3^2) & 2(q_1q_2-q_0q_3) & 2(q_0q_2+q_1q_3)\\ 2(q_0q_3+q_1q_2) & 1-2(q_1^2+q_3^2) & 2(q_2q_3-q_0q_1)\\ 2(q_1q_3-q_0q_2) & 2(q_0q_1+q_2q_3) & 1-2(q_1^2+q_2^2)\end{bmatrix}\tag{1-5}$$

所以我们可以根据四元数求得坐标变换矩阵，因为欧拉角也对应了一个坐标变换矩阵。所以对应的可以求出航向角。

关于四元数的更新，因为四元数本身的角度表示，可以得到微分表达式。

$$\frac{d\vec{Q}}{dt}=\frac{1}{2}\vec{Q}\otimes\vec{\omega_{nb}}^{b}\tag{1-6}$$

$$\vec{Q}_{k+2}=e^{\frac{1}{2}\Delta\Theta}\vec{Q}_k=\left[I+\frac{\frac{1}{2}\Delta\Theta}{1!}+\frac{\left(\frac{2}{2}\Delta\Theta\right)^2}{2!}+\cdots\right]\vec{Q}_k\tag{1-7}$$

令 $\Delta\vec{\theta}=\begin{bmatrix}\omega_x\Delta t\omega_y\Delta t\omega_z\Delta t\end{bmatrix}^T$，其模长为 $\left\|\Delta\vec{\theta}\right\|=(\omega_x\Delta t)^3+(\omega_y\Delta t)^2+(\omega_z\Delta t)^2$。忽略式 $(1-7)$ 高阶项就得到了各阶近似方程（这里只保留了一阶和四阶）：

$$\begin{aligned}\text{一阶近似}\quad :\vec{Q}_{k+2} &= \left[I+\frac{\Delta\Theta}{2}\right]\vec{Q}_k\\ \text{四阶近似}\quad :\vec{Q}_{k+2} &= \left[I\left(1-\frac{\left\|\Delta\vec{\theta}\right\|}{8}+\frac{\left\|\Delta\vec{\theta}\right\|^2}{384}\right)+\left(\frac{2}{2}-\frac{\left\|\Delta\vec{\theta}\right\|}{48}\right)\frac{\Delta\Theta}{2}\right]\vec{Q}_k\end{aligned}$$

令 $\Delta\vec{\theta}=\begin{bmatrix}\omega_x\Delta t & \omega_y\Delta t & \omega_z\Delta t\end{bmatrix}^T$，其模长为 $\left\|\Delta\vec{\theta}\right\|=(\omega_x\Delta t)^3+(\omega_y\Delta t)^2+(\omega_z\Delta t)^2$。忽略式 $(1-7)$ 高阶项就得到了各阶近似方程：因为四阶滤波取四阶近似方程得到结果更精准，所以采用四阶方法进行更新。如此更新四元数同时，旋转矩阵也可以通过三轴矩阵相乘得到，这里我们取姿态矩阵的旋转顺序为 $Z\to Y\to X$ 即可逐次计算航向角，

$$\mathbf{C}_{\mathbf{E}}^{\mathbf{B}}=\begin{bmatrix}\cos\gamma\cos\psi & \cos\gamma\sin\psi & -\sin\gamma\\ \sin\theta\sin\gamma\cos\psi-\cos\theta\sin\psi & \sin\theta\sin\gamma\sin\psi+\cos\theta\cos\psi & \sin\theta\cos\gamma\\ \cos\theta\sin\gamma\cos\psi+\sin\theta\sin\psi & \cos\theta\sin\gamma\sin\psi-\sin\theta\cos\psi & \cos\theta\cos\gamma\end{bmatrix}\tag{1-8}$$

这两个矩阵应该是相等，因此想要求出航向角，只要求 $\arctan(\frac{\mathbf{C}_{\mathbf{E}}^{\mathbf{B}}[0][1]}{\mathbf{C}_{\mathbf{E}}^{\mathbf{B}}[0][0]})$ 即可。从以上几种算法

的介绍中，每个算法求解欧拉角都有其自身的小缺点，但通过综合对比，在四轴上用来解算欧拉角的更好的算法是四元数法，虽然其也有缺点，但其优越性在于计算量小、精度高、可避免奇异性，因此被大多数人所采用。所以下面我们将接收通过如何通过四元数来计算我们所需要的姿态角。

互补滤波 互补滤波的原理是通过对同一个量计算两次，对两次取一个加权平均，如此可以避免某一种偏斜过大。在这里即高频的陀螺数据具有较高的带宽，动态特性良好，但由于零偏和累积误差的存在其低频数据的可信度较低。加速度传感器，低频区域良好，但是高频噪声大。而互补滤波器可以取长补短，提高传输性能。在这里就是考虑一个简单的一阶系统：

$$\dot{\theta} = \omega$$

我们有两种手段对系统状态进行测量，其一直接测量系统状态本身，即 $y_\theta = \theta + \varepsilon_\theta$，其中 y_θ 是测量值，ε_θ 是测量噪声。其二是间接的测量 ω 然后通过积分得到对系统状态的估计，即 $y_\omega = \omega + \varepsilon_\omega + b_1$，其中 y_ω 是测量值，ε_ω 是测量噪声，然后我们想要求出 θ，就要对进行 y_ω 进行积分。b_1 是测量值中的一个固定的偏移量，它将在积分过程中得到累积。互补滤波就是通过设计滤波器 $F_\theta(s) + F_\omega(s) = 1$ 来估计系统状态：

$$\hat{\theta}(s) = F_\theta(s) y_\theta + F_\omega(s) \frac{y_\omega(s)}{s}$$

分母上的 s，是因为在 z 变换中，积分会产生一个分母。其中 $F_\theta(s)$ 是一个低通滤波器，F_ω 是一个高通滤波器，两者的传递函数求和为 1 可以保证它们的穿越频率 (cross over frequency) 相同，并在频域上形成互补的关系。

使用互补滤波的方法，我们就可以实际上把四元数法和分解向量法结合在一起，用两个向量叉乘的方式得到误差向量进行修正。滤波器的系数可以在 matlab 通过滤波器频谱特性确定，过滤高频噪声，人走路的频率处于 1.5Hz 左右。最后确定系数。这就是互补滤波的原理。

2 软件设计

2.1 模块层次

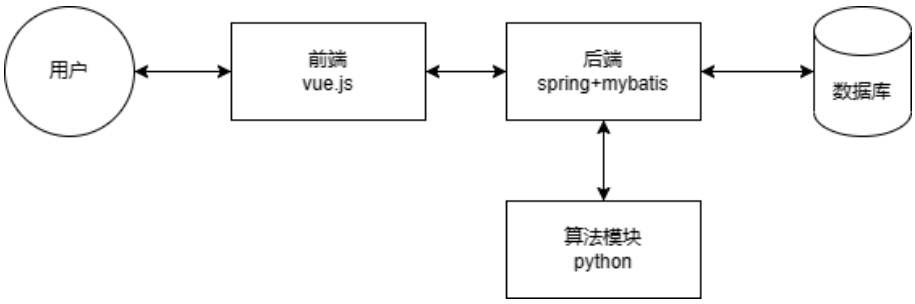


图 2-1 模块层次

2.2 技术选型

在为各模块选择实现方法时，主要考虑到学习难度、相关资源是否丰富和技术是否成熟等。前端和后端选用业界开发最常用的 vue+spring 组合，这套轻量级框架学习资源较多，能够大大简化开发难度而且技术成熟，相关支持较多，因此是比较适合的选择。算法部分采用 python 语言，因为它有很多数学计算库可以方便调用。

2.3 技术实现

2.3.1 算法

算法设计

算法部分实现

峰值检测 峰值检测，采用的是 AMPD 方法。其中输入参数为加速度和窗口，窗口是如前文算法设计中提到的为了一些波峰不明显的情况进行限制，用 for 循环计算局部极大值图，找到主周期，并根据修正后的周期得到峰值对应的序号。同时要根据模式不同，分别检测加速度和角速度来检测步频，因为摇摆状态加速度波动较大，而角速度因为摇摆具有较为明显的周期性。实验结果较好，其中对 batch27 的波峰检测效果如图 2-2所示，对于后三组，我们使用角速度进行峰值检测如图 2-3, 可以看到，峰值检测效果较好，而且加窗口是非常有必要的，否侧会因为平顶波峰漏掉点。

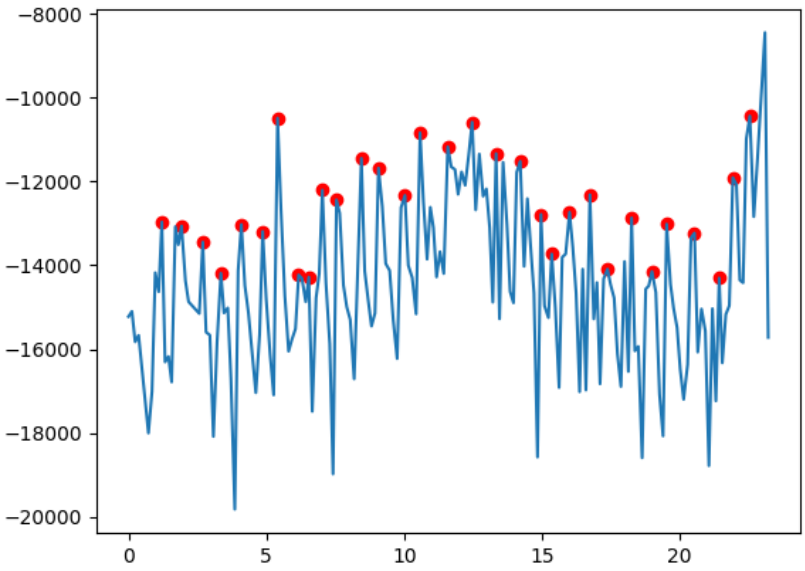


图 2-2　对 batch27(平持状态) 的加速度峰值检测

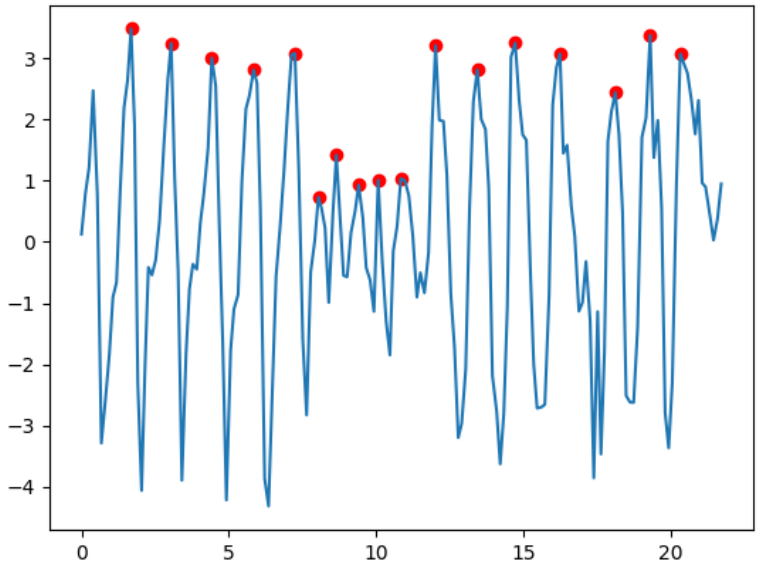


图 2-3　对 batch30(摇摆状态) 的角速度峰值检测

PDR 类的初始化　因为步长计算非常简单，因此没有单写一个函数，而是直接添加到了航向推算的中间。类的初始化进行了数据读入和预处理，使用 csv.reader() 函数读取 csv 文件，使用一个 header 变量判断是否为 csv 第一行，在第一行根据“x”，“y”，“accx”等实际含义求得对应的 index（列数）。使用行的遍历，读取对应的数据，之后加入到数组中，类的初始化中加入了 accx = [] 等一些列表。并且在全部读入后，使用 numpy.array 函数使其变为 numpy 数

组。在读入的过程中，根据摇摆和平持两种方式，这两种方式对应的物体坐标系不同，我们在读入时进行处理，这样在之后的四元数计算时就不必再分开处理了。同时我们在类的初始化中对数据进行了预处理，对角速度除去了分辨率，同时考虑了陀螺仪 0 偏的影响并去除。将 position.csv 的数据读入到 x_position 和 y_position。

航向估计 航向估计这里既包括了四元数方法，也把分解向量法的结果也记录了，因为分解向量法代码非常简单。首先初始化四元数方向，之后根据式子计算姿态矩阵，进行互补滤波，计算分解向量法对应的航向，求梯度矩阵，更新四元数 Q，计算方位角，并存入类的列表。这里对方位进行展示如图 2-4，图 2-5所示, 四元数法因为航向在 180° 的时候会出现相位倒置的情况。

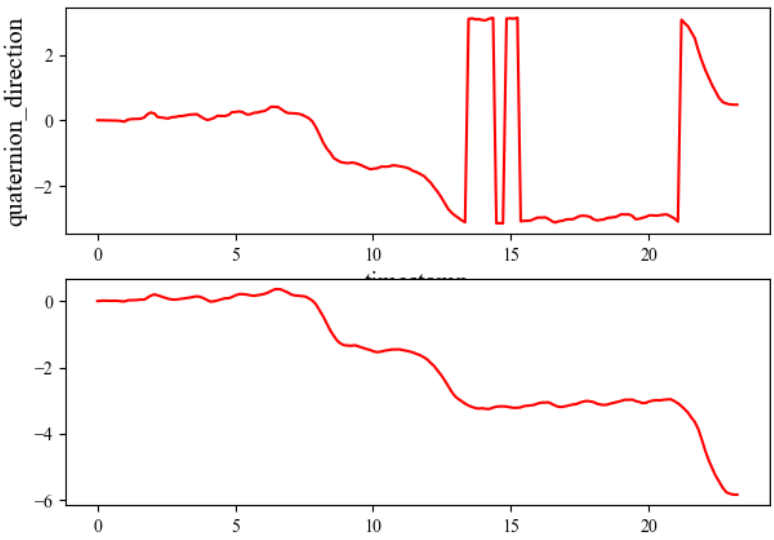


图 2-4 对 batch27(平持状态) 的航向估计 (上面是四元数法, 下面是分解向量法)

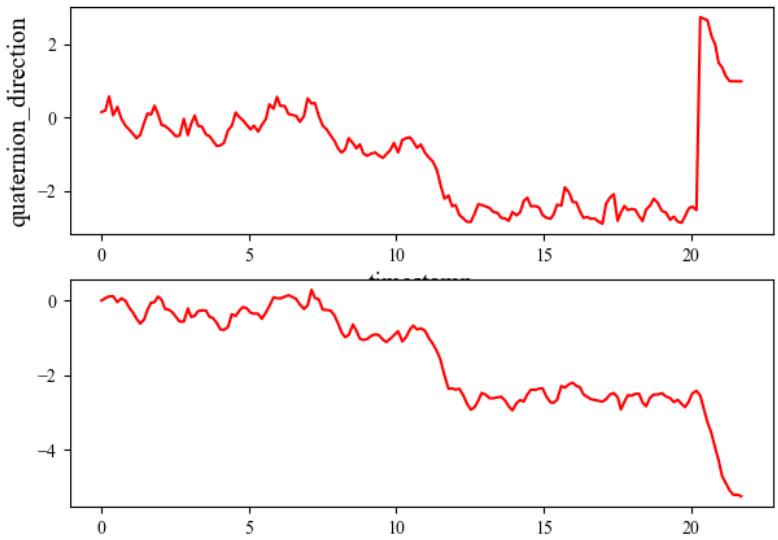


图 2-5　对 batch30(摇摆状态) 的航向估计 (上面是四元数法, 下面是分解向量法)

步长估计　vis2 函数主要是通过调用峰值检测得到峰值和谷值，并且使其相匹配，防止峰值谷值数目对应不上，之后使用公式 (1-2) 计算步长。得到步频和步长后就可以根据 (1-1) 之前算出的方位直接进行 PDR 的轨迹了，将步点的轨迹存到 x2_position 和 y2_position。

向量旋转　旋转矩阵函数，作用是把在物体坐标系下的加速度和角速度变到地理坐标系。本来这样是希望进行波峰检测判断摇摆状态的步频的，但效果一般，所以后来对摇摆状态使用的是角速度检测步频。

误差计算　峰值检测得到了一些步数，有可能偏多，有可能偏少，对于偏多的情况，我们根据对称以及匹配的原则，把多余的部分删去，因为实际上 PDR 预测的时间段是正确的，但是这里的数据有些是明显冗余的，所以对于多了的步点，要进行剔除，当然实际遇不到这个问题。因为给的数据是两步合到一起算一步的，所以我们会进行插值，使得真实值与预测值的序号可以对应。然后计算误差对于少了的部分，通过 error 函数，合理的进行计算得到误差。并且根据最近来寻找合适的 position 中的出发点。并且可以得到 CDF 曲线。如图 2-6，图 2-7所示。

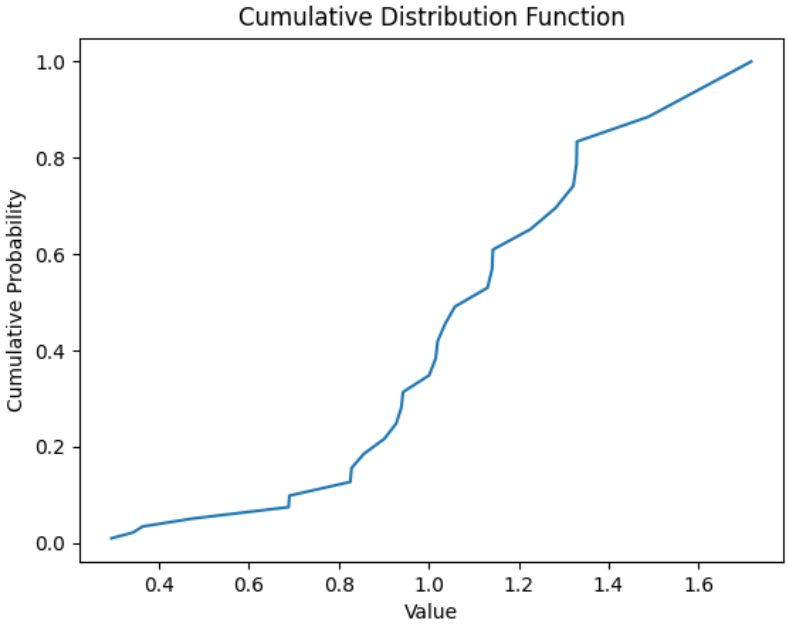


图 2-6　batch_27 的 CDF 曲线

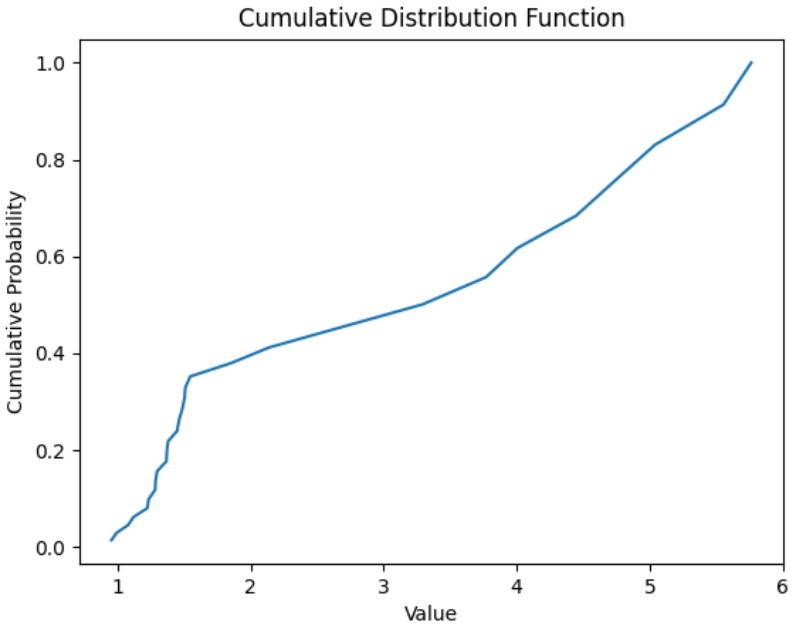
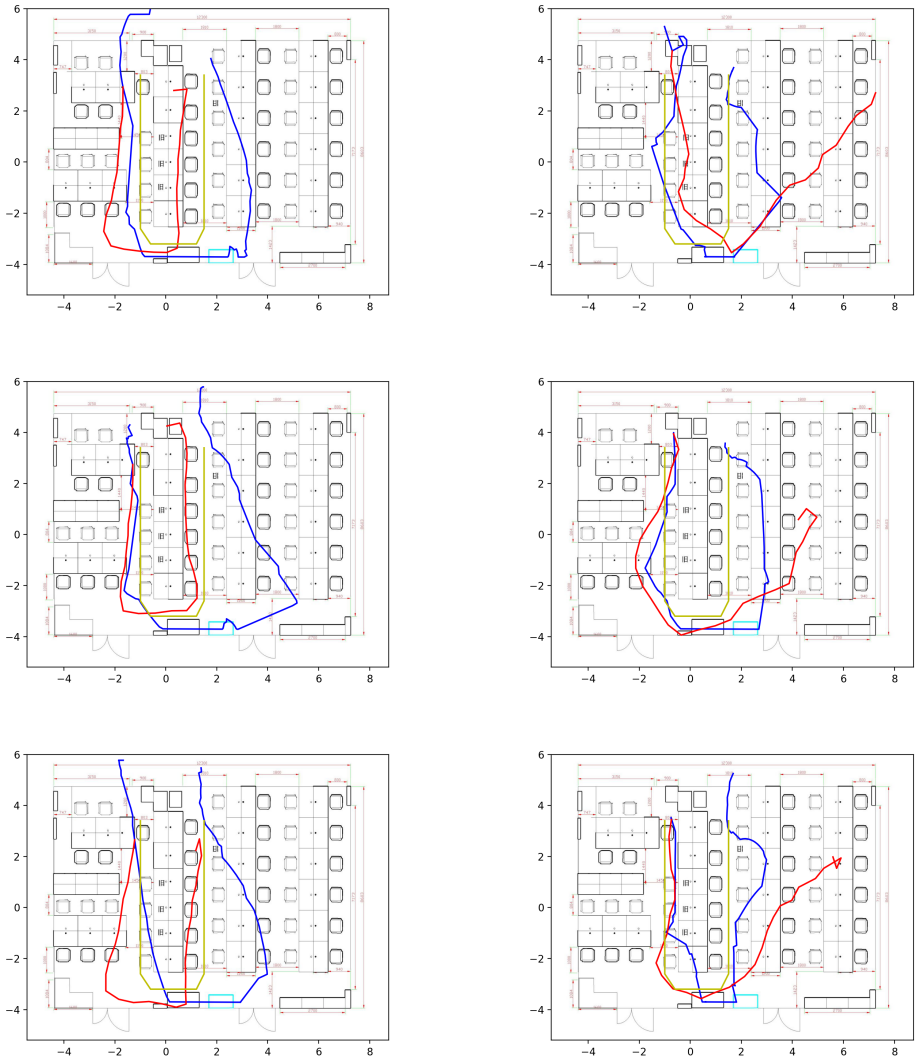


图 2-7　batch_30 的 CDF 曲线

数据保存　error() 函数通过计算预测轨迹和真实轨迹之间的误差来计算平均误差以及各误差分位点。并且因为后端数据库读取文件使用 json 格式，因此将误差保存成 json 文件。save() 函数保存预测的轨迹 x2_position，y2_position 和对应点的一些加速度信息，以及基站定位的坐标，都存到 json 文件中，方便后续数据库读入。



(a) 平持状态三组轨迹对比　　(b) 摇摆状态三组轨迹对比

图 2-8　六组数据的轨迹对比

python 模拟 PDR 因为 Rodrigues 参数法对于后三种情况效果不好，因此我们选择时只考虑了四元数法和分解向量法两种情况，通过表格分析，在前三组情况，分解向量法的误差较小，后三组情况四元数法效果较好，因此我们对两种不同的模式采取了不同的方法。实际上也可以考虑混合模式。由上面的各个函数，得出后，首先在 python 画图进行测试以确保结果正确，最终可以得到 PDR 预测的轨迹。如图 2-8所示，可以看到效果基本不错。其中黄色代表真实的轨迹，蓝色代表基站定位轨迹，红色是 PDR 预测轨迹测试后符合预期，算法部分就完成了。

表 2-1 不同方法误差对比表

batch	分解向量法平均误差	50% 误差	75% 误差	四元数法平均误差	50% 误差	75% 误差	混合法平均误差	50% 误差	75% 误差
27	1.038	1.059	1.287	0.997	1.015	1.226	0.977	0.939	1.158
28	1.292	1.083	1.977	1.443	1.032	2.260	2.789	1.018	3.351
29	0.783	0.671	0.902	0.836	0.864	0.946	2.793	1.009	4.303
30	2.418	1.625	3.41	2.178	1.354	3.129	2.302	1.483	3.291
31	2.336	0.963	4.322	1.449	0.886	2.516	1.918	0.767	3.45
32	2.000	0.583	3.641	1.389	0.533	2.039	1.785	0.602	2.776

2.3.2 后端设计

后端分为表示层、业务逻辑层和数据访问层。对应 Spring 框架中的 Controller 类，Service 类，Mapper 类。下面分别介绍这三个层次。

表示层

表示层其实就是用户能够看到的界面显示层，但是它的职责并不仅仅是显示界面那么简单，而是需要完成三件事情：

- 从界面中取得数据跟后台服务器交互
- 跟后台交互后进行数据绑定
- 将绑定的数据呈现在页面中

一般来说，这一层的设计会采用 MVC 的模式，M 称为模型也就是实体类，用于数据的封装和数据的传输；V 为视图也就是页面组件，用于数据的展示；C 为控制也就是流程事件，用于流程的控制。

简而言之，表示层是与前端打交道的，需要处理前端发来的数据请求。因此我们为每一个请求定义一个 Controller 类。这些 Controller 类在接收到前端发来的请求时可以处理该请求然后调用服务层提供的接口提供数据。

具体 API 如下:

- recPositon
    1. url:/impt/position

2. 请求方式：POST
3. 请求体：position.csv
4. 返回结果：boolean flag

- recRunning

1. url:/impt/running
2. 请求方式：POST
3. 请求体：running.csv
4. 返回结果：boolean flag

- recTruth

1. url:/impt/ground
2. 请求方式：POST
3. 请求体：truth.csv
4. 返回结果：boolean flag

- compute

1. url:/cmpt/predict/batch/isHold
2. 请求方式：GET
3. 请求体：batch、isHold
4. 返回结果：{”x”: double,
   ”y”: double,
   ”xtrue”: double,
   ”ytrue”: double,
   ”direction”: double,
   ”accx”:double,
   ”accy”: double,
   ”accz”: double,
   ”gyrx”:double,
   ”gyry”: double,
   ”gyrz”: double,
   ”length”:double,
   ”error”: double,
   ”sampleBatch”: double}

服务层

这一层的功能主要是实现一些具体问题的操作，因为它是表示层和持久层之间沟通的桥梁，主要负责数据的传递和处理。

在日常的代码开发中一般对应着逻辑 Service 层，对于一些复杂的逻辑判断和涉及到数据库的数据验证都需要在这一层做出处理，同时根据传入的值返回用户想得到的值，或者处理相关的操作。

服务层承担处理数据的任务，向上与表示层交互，向下与持久层交互。所以这部分的代码一方面要为表示层提供接口，另一方面要调用持久层提供的接口。

接口定义如下：

```
public interface IRunningService extends IService<Running>
public interface IPredictService extends IService<Predict>
public interface IPositionService extends IService<Position>
```

图 2-9　业务层

持久层

也称为数据访问层，顾名思义，这一层其实就是跟数据库直接打交道的层面，通过连接数据库，根据传入的值对数据库进行增删改查。

在持久层，我们使用 Mybatis 框架简化开发，只需要定义相应的数据层接口 Dao，就可以直接调用 Mybatis 提供的通用数据库操作接口。

数据库结构：

根据本项目的功能需求，我们使用了三个数据库 predict、running 和 position。分别存储预测数据，手表记录和基站定位。各数据库字段如下 对应每一张表都定义了一个数据类，接口定义

Table: position

Columns:

id	int(11)
address	text
x	double
y	double
z	double
stay	int(11)
timestamp	bigint(20)
bs_address	text
sample_time	text
sample_batch	int(11)

Table: predict

Columns:

x	double
y	double
x_true	double
y_true	double
direction	double
accx	double
accy	double
accz	double
gyrx	double
gyry	double
gyrz	double
length	double
error	double
sample_batch	double

Table: running

Columns:

id	int(11)
address	text
power	text
heart_rate	text
step	int(11)
temperature	text
systolic_pressure	text
diastolic_pressure	text
accx	int(11)
accy	int(11)
accz	int(11)
gyroscopex	int(11)
gyroscopey	int(11)
gyroscopez	int(11)
stay	int(11)
sos	text
elock_open_illegal_warn	text
timestamp	bigint(20)
bs_address	text
sample_time	text
sample_batch	int(11)

图 2-10　数据库

如下：

```
public interface PositionDao extends BaseMapper<Position>
public interface PredictDao extends BaseMapper<Predict>
public interface RunningDao extends BaseMapper<Running>
```

图 2-11　持久层

2.3.3 前端设计

前端设计，根据前述系统需求分析，以 Vue 作为前端框架，Vue-cli 脚手架构建基本前端项目，并规划路由，主要划分为主菜单（HomeView.vue）和信息显示 (AboutView.vue) 两个页面。其中主菜单功能包括介绍团队信息、进入软件系统等功能，实现较为简单，不做赘述。信息显示页面包括室内地图区、信息显示区、定位指标计算区等三个子地图区。因为我们小组对所有的 6 组数据都进行了处理，因此我们的页面可以显示 6 种不同的情况。下面从 AboutView 这一 vue 页面设计具体介绍前端设计思路：

页面路由设计

考虑到对室内行人系统的三大地图区设计任务要求不同，□ 室内地图区需要呈现室内平面图、行人真实轨迹和定位轨迹；□ 定位指标计算区，需要实时计算并展示当前定位轨迹的平均定位误差、百分位定位误差、CDF 曲线等定位指标；□ 信息显示区需要通过点击定位轨迹中的点，查看不同位置的传感器信息。因此在该 vue 页面中需要分模块进行相应功能的实现。随后设计导入文件接口，将数据文件传递给后端；设计 1-6 六个按钮，绑定触发事件，实现后端传输六组数据的切换。

室内地图区设计

室内地图区需要应用后端传递的数据，在地图上显示真实的行人轨迹、定位轨迹以及预测的行人轨迹。首先由于地图的像素坐标与在网页中显示的地图像素坐标不一致，因此需要将后端传递的位置数据进行相应的坐标变化。坐标变换后，使用 canvas 可视化工具库，在地图上渲染相应信息，成功将所需要的三条轨迹图画了出来。

指标计算区设计

通过对后端传递误差数据进行相应的数据处理，然后利用 echart 进行可视化设计，将 CDF 曲线较好地在该地图区显示出来。

2.3.3.1 信息显示区设计

信息展示区左上方为室内地图区，正上方为指标计算区。再经过坐标变换后，通过 mousedown 方法，获取鼠标点击时的像素值，通过 v-if 和点击事件绑定数据判断与后端传递过来的数据文件中的数据点是否足够接近，若足够接近，则可认为鼠标点击的即为该数据点，即将该数据点的位置、加速度、角速度等信息显示在该地图区，不占用空间且提高了用户体验。

3 总结与建议

我们今年因为疫情的影响，软件课设推迟到了开学验收，因此我们的项目主要是寒假进行完成的。在这次的软件课程设计中，我们小组三人针对这一软件项目的实现进行了分工，其中我主要负责算法部分的设计以及算法和后端的衔接读入，在着手算法的设计之前自己从来没有接触定位方面的知识。首先我经过自己复杂的思考，独立发现了 Rodrigues 参数法。这个过程是困难的，首先第一个问题，角速度能不能合成，对矢量加旋转并不了解的我，经过思考并且向物理专业的同学请教后得到了肯定的答案。于是这一问题得到解决，那么旋转的时候旋转矩阵如何得到呢，独立推导时是困难的，幸而找到一些资料做参考，最终得到了旋转矩阵的公式。然而开始在我写好后，出来的结果确是原地打转，我百思不得其解，只能单步调试，发现旋转的角度太大了，开始不知道为什么要加一个系数，后来发现原来要加一个角速度的分辨率，于是前三组得到了近似的效果。前三组成了，那后三组呢？用这一方法在后三组上蹉跎几日，我并没有什么进展，于是我去请教了一名对遥感定位的专业人士，他向我解释了一些方法并不完备，需要使用四元数才有较好的效果。同时助教学长也给出了一个启发（分解向量法），于是我又重新开始编写算法，从 main 到 main3，一遍一遍的调参，遇到各种 bug，也得到了更加令人记忆深刻的领悟，最终实现了这三种的轨迹实现。但是只得到远远不够，我希望得到更少的误差，更准确的结果。于是又考虑到了零偏这一问题，通过简单计算，增加零偏后，效果果然好了很多。同时一些参数也是调了又调，三种方法也经过了对比。之后在计算误差时，因为步数不同，需要进行删除，同时误差的计算也下了一番功夫。采用了插值的方法使得步数匹配，从而计算误差。在这个过程中我从一个从未接触过陀螺仪定位的小白逐渐了解到这方面很多的算法知识以及实现原理，使我更加明白了从实践中来、在实践中学习的含义。在软件开发的过程中，我感受到的不仅是个人能力的提高，在与小组队友之间的讨论与学习之中，更是意识到了团队协作的重要性，我们一起进行接口的一一调试时。很快就将项目完成。我们了解到彼此之间相互学习的重要性，最后，我们对自己的结果也比较满意，也希望今后能够通过团队合作进一步提高自己的能力！

对于这门软件课程设计的一些建议，个人认为包括，希望每个人都能前端、后端、算法都会是合理的，但是一般情况下，一个项目的分工是不能出现两个人同时负责某一个部分，这样不利于项目的进展，作为学生，我更倾向于每人负责一部分，因为如果我没有什么基础的话，我需要同时学习前端与后端还是有些困难。第二个是希望测试数据多给一些，在本次实验中就是在测试的六组数据中效果很好，但是在验收的时候就出现了一些 bug，因为我错把局部现象当作了一般情况。结果当场调试代码出现了一些问题。总而言之，软件课程设计这门课程任务的存在确实是能够让我们的能力得到提高的，同时也对一个软件的完整框架有所了解。在最后也是希望软件课设这门课程也能越来越好！

参考文献

[1] chongbin li. 三维空间绕任意轴旋转矩阵的推导. 2020. https://zhuanlan.zhihu.com/p/56587491, Last accessed on 2023-3-4.

[2] Wensong Li, Bang Wang, Laurence T Yang, and Mu Zhou. Rmaptafa: Radio map construction based on trajectory adjustment and fingerprint amendment. *IEEE Access*, 7:14488–14500, 2019.

[3] Felix Scholkmann, Jens Boss, and Martin Wolf. An efficient algorithm for automatic peak detection in noisy periodic and quasi-periodic signals. *Algorithms*, 5(4):588–603, 2012.

[4] 大锅豁阔落. 推荐一个非常实用的峰值查找算法（peak detection）. 2022. https://zhuanlan.zhihu.com/p/549588865, Last accessed on 2023-3-4.

[5] 无处不在的小土. 定点转动与四元数. 2021. https://gaoyichao.com/Xiaotu/?book=math_physics_for_robotics&title=%E5%AE%9A%E7%82%B9%E8%BD%AC%E5%8A%A8%E4%B8%8E%E5%9B%9B%E5%85%83%E6%95%B0, Last accessed on 2023-3-4.

[6] 黄承恺. 多传感器信息融合算法在室内定位中的应用. PhD thesis, 北京邮电大学, 2015.

[7] 齐保振. 基于运动传感的个人导航系统及算法研究. PhD thesis, 杭州: 浙江大学, 2013.

附录

A 前端代码结构

- **前端代码结构：**

```
├── README.md 项目说明以及运行方法
├── node_modules 通过 node 下载的模块，以下仅为部分示例
│   ├── build.js                生产环境构建代码
│   ├── check-versions.js       检查 node&npm 等版本
│   ├── utils.js 构建配置公用工具
│   ├── vue-loader.conf.js      vue 加载器
│   ├── webpack.base.conf.js    webpack 基础环境配置
│   ├── webpack.dev.conf.js     webpack 开发环境配置
│   └── webpack.prod.conf.js    webpack 生产环境配置
├── vue.config.js     项目开发环境配置相关代码
├── index.html 入口页面
├── package-lock.json 项目基本信息
├── package.json 项目基本信息
├── src     源码目录
│   ├── App.vue 页面入口文件
│   ├── assets  资源目录
│   │   ├── logo.png    软件 logo 文件
│   │   └──img 文件夹 存储页面展示用图
│   ├── components  vue 公共组件
│   │       └── HelloWorld.vue     默认渲染组件
│   ├── main.js 程序入口文件，加载各种公共组件
│   ├── router      前端路由
│   │   └── index.js          路由配置文件
│   └── views                 vue 视图文件，被路由引用
│           ├── HomeView.vue    主页面
│           └── AboutView.vue   信息显示页面
└── utils       接口文件
    ├── api.js 封装接口
    └── request.js 配置前端接口，实现数据传输功能
```

B 后端代码结构

src
- main.java.com.example.rjks
  - controller 表现层
    - ComputeController.java 用来接收前端的计算命令，生成预测结果并返回
    - ImptController.java 用来接收前端导入的文件并保存
    - PositionController.java 为前端返回 position 数据
    - PredictController.java 为前端返回预测数据
  - dao 数据层
    - PositionDao.java 提供 Position 表的数据库交互操作
    - PredictDao.java 提供 Predict 表的数据库交互操作
    - RunningDao.java 提供 Running 表的数据库交互操作
  - domain 实体类
    - Position.java Position 实体类，属性与 Position 表字段一一对应
    - Predict.java Predict 实体类，属性与 Predict 表字段一一对应
    - Running.java Running 实体类，属性与 Running 表字段一一对应
  - service 服务层
  - tool 工具类
    - Function.java 调用 python 脚本的功能类
    - ParseCsv.java 解析 csv 文件的功能类
- test.java.com.example.rjks. 测试文件路径
  - RjksApplicationTests.java. 测试类