

第二次大作业 P2: PVT 解算

聂浩 (2013011280)

2016 年 6 月 19 日

1 单系统定位：自己实现一种定位解算算法，分别利用 GPS、北斗的数据文件计算用户的位置与速度。通过查阅地图，看看用户的位置在哪里？

a 单 GPS

中国境内大部分地图采用的为 GCJ02 火星坐标系，而百度地图使用的为 BD09 坐标系，而所提供的 GPS 数据为基于 WGS84 的 ECEF 坐标系。

本次大作业中首先完成的是对于以上坐标系的相互转换¹以及 ECEF 与 LLH 坐标的转换²，具体可见 co_trans³中的文件。

随后进行 PVT 解算，由于本作业中没有提供 T 的数据，所以这里仅仅进行 PV 解算，代码如下

gps.m

```
1 clear;
2 close;
3 load L1.mat
4 [lng, lat]=gcj02towgs84(116.3324885066,40.0014938354);
5 pre=lla2ecef([lat, lng, 0]);
6 %以清华大学主楼为起始点，坐标来自高德地图
7
8 position_ecef=zeros(3,200);
9 position_wgs84=zeros(2,200);
10 position_gcj02=zeros(2,200);
11 position_bd09=zeros(2,200);
12 for i=1:200
13     rho=PseudoRange(:, i);
14     sat=SatInfo(:, i);
15     position_ecef(:, i)=pvt(rho, sat, pre);
16     lla=ecef2lla(position_ecef(:, i)');%转换为wgs84
17     lat=lla(1);
18     lng=lla(2);
19     position_wgs84(:, i)=[lng; lat];
20     [lng, lat]=wgs84togcj02(lng, lat);
```

¹这里参考了相关代码https://github.com/wandergis/coordTransform_py

²使用 matlab 2015b 自带函数

³执行本作业中的程序需要将 co_trans 文件夹添加到路径中，或者将其中的文件拷出

```

21 position_gcj02(:,i)=[lng,lat];
22 [lng,lat]=gcj02tobd09(lng,lat);
23 position_bd09(:,i)=[lng,lat];
24 end
25 v=sum((position_ecef(:,2:200)-position_ecef(:,1:199)).^2).^0.5;

```

pvt.m

```

1 function [pre]=pvt(rho,sat,pre)
2 %%%输出 pre为最终得到的坐标
3 %%%输入 rho为伪距
4 %%% sat为卫星坐标
5 %%% pre为预测的用户位置
6
7 %获得卫星的数量
8 [n,~]=size(rho);
9
10 %将卫星位置从结构体转换为n*3的矩阵
11 sat=reshape(struct2array(sat),3,n)';
12
13 error=100;
14 %循环条件，当计算修正位置小于20m时停止迭代
15 count=1;
16 %循环条件，防止结果不收敛而导致死循环
17 while(error>20||count>20)
18     count=count+1;
19     tmp_pre= repmat(pre,n,1);
20     %矩阵化，方便计算
21     r=(sum((sat-tmp_pre).^2,2)).^0.5;
22     delta_rho=rho-r;
23     H=-(sat-tmp_pre).*repmat(r.^-1,1,3);
24     delta_p=inv(H'*H)*H'*delta_rho;
25
26     error=sum(delta_p.^2)^0.5;
27     %error为delta_p的长度
28     pre=pre+delta_p';
29 end
30 return;

```

计算得到的位置点如图1⁴，速率如图2，单位为 m/s

b 单北斗

根据相关参考文献，“WGS84 坐标与 CGCS2000 坐标的差值在 cm 级范围内”⁵，所以这里直接用 WGS84 坐标系对北斗信号进行结算，相关代码见附录所附 beidou.m。计算得到的位置点如图3，速率如图4，单位为 m/s

⁴该图为调用百度地图 API 绘制，相关代码见 map/gps.html，请使用 Chrome 浏览器打开

⁵WGS84 和 CGCS2000 坐标转换研究彭小强



图 1: GPS 定位结果

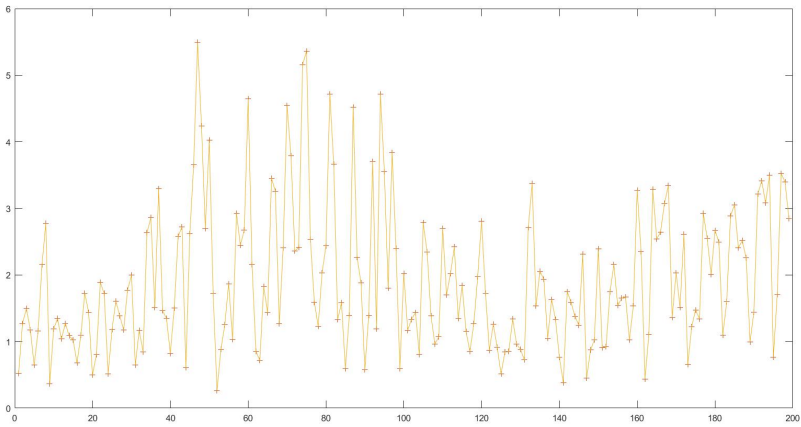


图 2: GPS 定位速率



图 3: 北斗定位结果

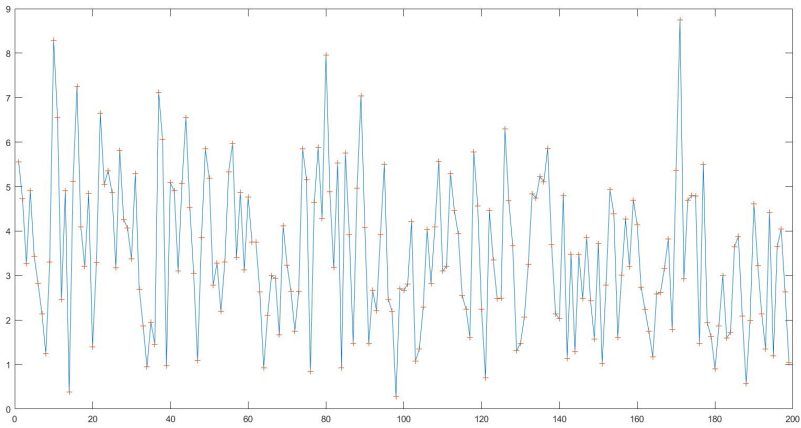


图 4: 北斗定位速率



图 5: 联合定位结果

2 联合定位(多模接收): 自己实现一种定位解算算法, 联合使用 GPS、北斗的数据文件计算用户的位置与速度。并将该定位结果与单独使用 GPS、北斗时的定位解算结果进行比较? 给出你的结论?

根据上一问的结论可以看出, 200 个定位点之间的距离非常小, 即整个定位过程中用户几乎没有移动, 这里将北斗的 7 颗卫星信息直接附在 GPS 信号之后, 即 PVT 解算中以 14 颗星进行解算。代码见附录 merge.m 计算得到的位置点如图5, 速率如图6, 单位为 m/s, 从定位图可以看出, 多模接受得到的 200 个定位点更加集中, 可见其精度更高。

3 在导航系统使用中, 一般希望收到的卫星数越多越好, 即希望采用更多的卫星来进行定位解算, 这也是目前国际上开发新卫星导航系统的出发点之一。在此, 试改变 GPS、北斗可用卫星数, 然后再定位, 并比较定位解算结果的变化? 给出你的结论?

只是用 GPS 的前五颗卫星数据进行 PVT 解算, 相关代码见附录 reduce.m.

所得结果如图7, 速率如图8, 单位为 m/s, 从定位图可以看出, 卫星减少后定位的区域变大, 可见

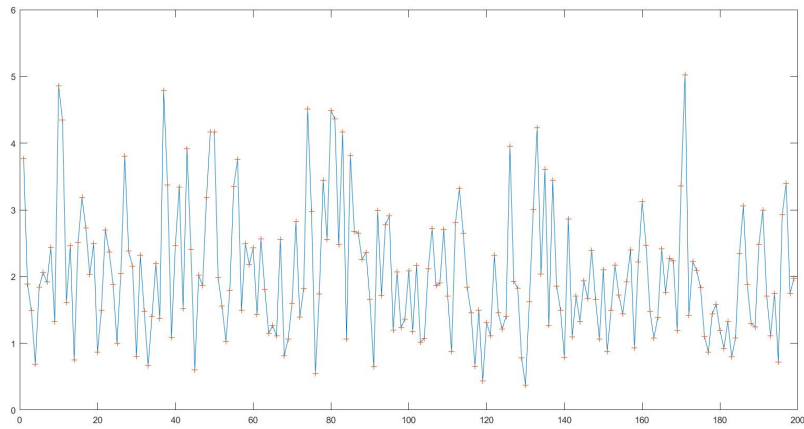


图 6: 联合定位速率

会出现更大的误差。

4 附录

beidou.m

```

1 clear;
2 close;
3 load B1.mat
4 [lng, lat]=gcj02towgs84(116.3324885066,40.0014938354);
5 pre=lla2ecef([lat, lng, 0]);
6 %以清华大学主楼为起始点, 坐标来自高德地图
7 position_ecef=zeros(3,200);
8 position_wgs84=zeros(2,200);
9 position_gcj02=zeros(2,200);
10 position_bd09=zeros(2,200);
11 for i=1:200
12     rho=PseudoRange(:, i);
13     sat=SatInfo(:, i);
14     position_ecef(:, i)=pvt(rho, sat, pre); %PVT解算
15     lla=ecef2lla(position_ecef(:, i)');
16     lat=lla(1);
17     lng=lla(2);
18     position_wgs84(:, i)=[lng; lat];
19     [lng, lat]=wgs84togcj02(lng, lat); %转换为gcj02
20     position_gcj02(:, i)=[lng; lat];
21     [lng, lat]=gcj02tobd09(lng, lat); %转换为bd09
22     position_bd09(:, i)=[lng; lat];
23 end
24 v=sum((position_ecef(:, 2:200)-position_ecef(:, 1:199)).^2).^0.5;

```



图 7: 减少卫星后 GPS 定位结果

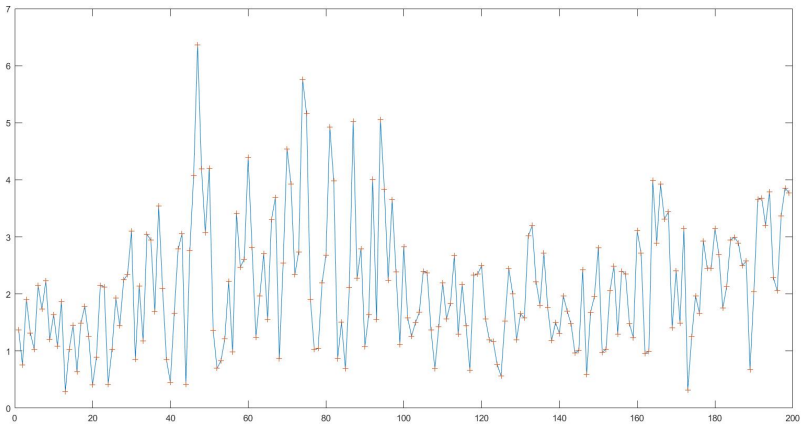


图 8: 减少卫星后 GPS 定位速率

merge.m

```

1  clear;
2  close;
3  load L1.mat
4  P=PseudoRange;
5  sat=SatInfo;
6  load B1.mat;
7  PseudoRange=[P;PseudoRange];
8  SatInfo=[sat;SatInfo];
9  [lng,lat]=gcj02towgs84(116.3324885066,40.0014938354);
10 pre=lla2ecef([lat,lng,0]);
11 %以清华大学主楼为起始点,坐标来自高德地图
12
13 position_ecef=zeros(3,200);
14 position_wgs84=zeros(2,200);
15 position_gcj02=zeros(2,200);
16 position_bd09=zeros(2,200);
17 for i=1:200
18     rho=PseudoRange(:,i);
19     sat=SatInfo(:,i);
20     position_ecef(:,i)=pvt(rho,sat,pre);
21     lla=ecef2lla(position_ecef(:,i)');
22     lat=lla(1);
23     lng=lla(2);
24     position_wgs84(:,i)=[lng,lat];
25     [lng,lat]=wgs84togcj02(lng,lat);
26     position_gcj02(:,i)=[lng,lat];
27     [lng,lat]=gcj02tobd09(lng,lat);
28     position_bd09(:,i)=[lng,lat];
29 end
30 v=sum((position_ecef(:,2:200)-position_ecef(:,1:199)).^2).^0.5;

```

reduce.m

```

1  clear;
2  close;
3  load L1.mat
4  %仅仅保留前五个卫星的数据
5  PseudoRange=PseudoRange(1:5,:);
6  SatInfo=SatInfo(1:5,:);
7  [lng,lat]=gcj02towgs84(116.3324885066,40.0014938354);
8  pre=lla2ecef([lat,lng,0]);
9  %以清华大学主楼为起始点,坐标来自高德地图
10
11 position_ecef=zeros(3,200);
12 position_wgs84=zeros(2,200);
13 position_gcj02=zeros(2,200);
14 position_bd09=zeros(2,200);
15 for i=1:200
16     rho=PseudoRange(:,i);
17     sat=SatInfo(:,i);
18     position_ecef(:,i)=pvt(rho,sat,pre);
19     lla=ecef2lla(position_ecef(:,i)');
20     lat=lla(1);

```



```
21     lng=lla(2);  
22     position_wgs84(:,i)=[lng;lat];  
23     [lng,lat]=wgs84togcj02(lng,lat);  
24     position_gcj02(:,i)=[lng;lat];  
25     [lng,lat]=gcj02tobd09(lng,lat);  
26     position_bd09(:,i)=[lng;lat];  
27 end  
28 v=sum((position_ecef(:,2:200)-position_ecef(:,1:199)).^2).^0.5;
```