

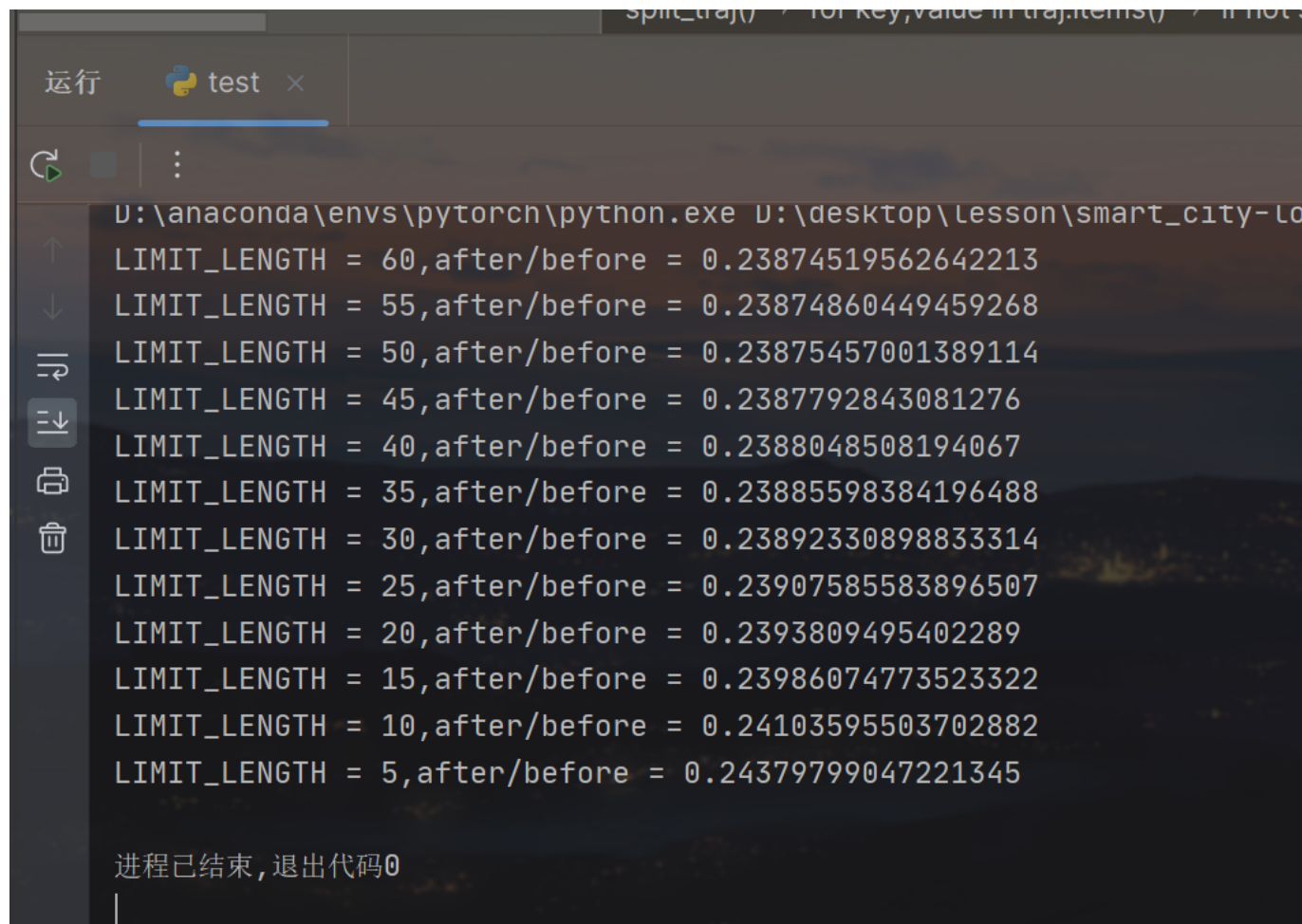
实验1

轨迹分段

根据LIMIT_DT, LIMIT_DISTANCE, LIMIT_POINT

但是没有做根据LIMIT_LENGTH, 因为自己写的筛选的算法貌似有问题, 如果是不需要分割的轨迹段, 所有的轨迹段根据这个筛选之后都只剩余了两个点,

而且测试了不同LIMIT_LENGTH, 筛选之后的前后的总轨迹点的比例几乎没有变化, 所以认为自己没写对



```
U:\anaconda\envs\pytorch\python.exe U:\desktop\lesson\smart_city-Lo
LIMIT_LENGTH = 60,after/before = 0.23874519562642213
LIMIT_LENGTH = 55,after/before = 0.23874860449459268
LIMIT_LENGTH = 50,after/before = 0.23875457001389114
LIMIT_LENGTH = 45,after/before = 0.2387792843081276
LIMIT_LENGTH = 40,after/before = 0.2388048508194067
LIMIT_LENGTH = 35,after/before = 0.23885598384196488
LIMIT_LENGTH = 30,after/before = 0.23892330898833314
LIMIT_LENGTH = 25,after/before = 0.23907585583896507
LIMIT_LENGTH = 20,after/before = 0.2393809495402289
LIMIT_LENGTH = 15,after/before = 0.23986074773523322
LIMIT_LENGTH = 10,after/before = 0.24103595503702882
LIMIT_LENGTH = 5,after/before = 0.24379799047221345

进程已结束, 退出代码0
```

写了两个筛选的代码, 都放在了注释里面

```
if not segment_loc_tmp and len(traj_mercator) > 0:
    LIMIT_POINT:
        # filter 过滤邻近冗余坐标值 返回索引 根据tmp_traj计算每两个点之间的距离，若其之间的距离小于LIMIT_LENGTH
        # 轨迹长度小于该值的将被过滤掉
        # 过滤方法1
        # filter_distance_index = np.where(diff_distance > LIMIT_LENGTH)[0]
        # filter_timestamp = [timestamp[i] for i in filter_distance_index]
        # filter_traj = [traj_raw[i] for i in filter_distance_index]
        # res[str(traj_id)] = [filter_timestamp, filter_traj]
        # traj_id += 1

        # 过滤方法2 从第一个点开始，向后查找点，计算距离，若距离大于limit，添加点，循环
        # n = len(traj_mercator)
        # filter_index = []
        # queue = deque(range(1,n))
        #
```

第一个就是简单的np.where

第二个就是从第一个点开始向后查找点，直至两个点之间的距离大于limit，添加到列表中，往后类推

两个代码都没成功

轨迹去噪和简化

没啥问题