基于密度的停留点识别方法

李毓瑞，陈红梅，王丽珍，肖清
云南大学信息学院，云南 昆明 650091

摘要
从GPS轨迹点序列中识别停留点，是轨迹分析的重要预处理步骤，是用户行为分析、个性化兴趣点推荐等位置服务的基础，停留点识别方法的识别能力对位置服务的可用性和可靠性有根本性的影响。针对现有方法未考虑轨迹点的时间连续性或仅考虑时间连续性的一个方向所导致的停留点识别能力不足的问题，提出一种新的基于密度的停留点识别方法。该方法考虑了轨迹点的时空聚集，兼顾了轨迹点的时间连续性和方向性。在GeoLife数据集上的实验结果验证了该方法的识别能力强于基准方法，可以进一步识别基准方法不能识别的两类停留点。

关键词
停留点；密度；时间连续性；时间方向性

中图分类号：TP311.13 文献标识码：A doi: 10.11959/j.issn.2096-0271.2018052

Stay point identification based on density

LI Yurui, CHEN Hongmei, WANG Lizhen, XIAO Qing
School of Information Science and Engineering, Yunnan University, Kunming 650091, China

Abstract
Identifying stay points from GPS trajectory is an important preprocessing procedure of trajectory analysis and the foundation of location based service such as user behavior analysis and personal POI recommendation, and the capability of the stay point identification method has a fundamental impact on the availability and reliability of location based service. Existing methods for identifying stay points have some shortcomings due to not considering time continuity or only considering one direction of time continuity. A new method called stay point identification based on density (SPID) was proposed. SPID takes into account the spatial-temporal clustering of trajectory points, and the time directions and time continuity of trajectory points. The experimental results on Geolife dataset verify that SPID is better than the baseline methods, and can identify two kinds of stay points which can't be found by the baseline methods.

Key words
stay point, density, time continuity, time direction

1 引言

随着移动设备、移动互联网等技术的发展，用户、车辆等移动对象产生的GPS轨迹数据总量呈爆炸式增长。通过分析GPS轨迹数据，人们可以发现移动对象的时空模式，进而提供基于位置的服务，例如用户行为分析[1-2]、个性化兴趣点推荐[3-4]等。GPS轨迹数据的采样频率普遍较高，但其中的轨迹点并不是同等重要的，有些轨迹点仅仅是移动对象瞬时经过的地方，例如用户乘车经过的公交站，而有些轨迹点集合代表了移动对象在某个地方停留了一段时间，即停留点，例如代表用户在商场购物或在家中休息的轨迹点集合。停留点是移动对象在较小空间区域内停留了较长时间的轨迹点集合。从GPS轨迹数据中识别停留点，可以有效去除GPS轨迹数据中不重要的和冗余的信息，而得到的停留点序列有助于对GPS轨迹序列的深入理解。因此停留点识别是轨迹数据分析的重要预处理步骤，是位置服务的基础。

现有的停留点识别方法可以分为3类[5]：基于聚类策略的方法、基于概率策略的方法和基于区分策略的方法。基于聚类策略的方法是对GPS轨迹数据进行时空聚类，包括仅考虑空间因素的方法（如DBSCAN[6]）以及同时考虑时间因素的方法（如DJ-Cluster[7]和CB-SMoT[8]）。基于概率策略的方法通过概率模型，从GPS轨迹数据中推导频繁访问的地点，这些概率模型包括高斯混合模型[9]、贝叶斯模型[10]、条件随机场[11]。基于区分策略的方法通过分析GPS轨迹点之间的时间和空间差异，寻找停留点及其代表点[12-16]，其需要两个阈值：限定移动对象停留区域大小的距离阈值和限定移动对象停留时间长度的时间阈值。在这3类方法中，基于聚类策略的方法主要考虑轨迹点的时空邻近，基于概率策略的方法主要考虑轨迹点的访问频率，它们都没有考虑轨迹点的时间连续性和方向性。而基于区分策略的方法仅考虑了时间连续性的一个方向，没有考虑停留点中轨迹点的时空聚集。

为了更好地识别停留点，有必要考虑轨迹点的时间连续性和方向性以及时空聚集。如果不考虑轨迹点的时间连续性，可能会得到一些无意义的停留点[14]。如**图1**所示，用户在上班、购物、回家途中多次但不连续经过虚线所圈的路口，深色轨迹点之间的距离满足距离阈值且它们的累计时间满足时间阈值。如果不考虑时间连续性，它们会被判定为停留点，但是它们并不代表用户的停留行为。

如果仅考虑时间连续性的一个方向，可能会漏掉一些有意义的停留点。如**图2**所示，由5个轨迹点组成的轨迹点序列分布在较小的空间区域内且时间跨度较大，从而构成一个停留点，其中，轨迹点p_1到p_2的距离大于距离阈值，但轨迹点p_3到其他点的距离都小于距离阈值，并且p_1到p_5的时间跨度大于时间阈值，但p_2到p_5的时间跨度小于时间阈值。若仅考虑向前这个方向，由于p_1到p_2的距离大于距离阈值，而p_2到p_5的时间跨度不满足时间阈值，因此不能识别这个停留点[14]。但是，若以p_3为锚点，考虑向前和

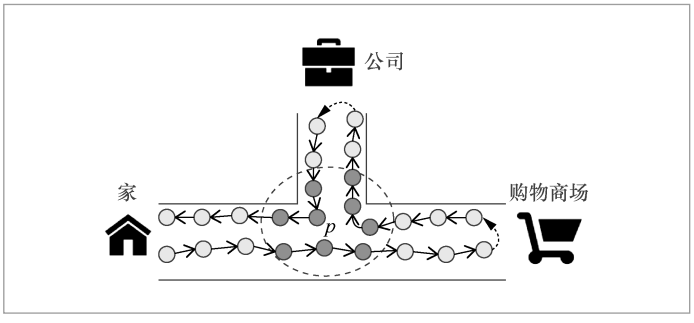


图1 不考虑时间连续性下的停留点

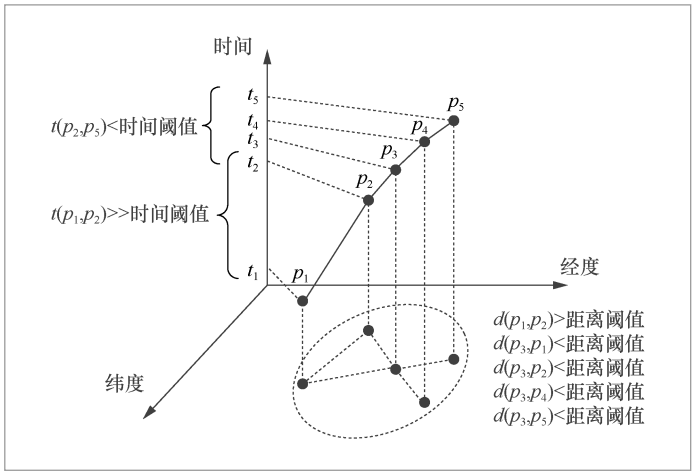


图 2 考虑时间方向性后的停留点

向后两个方向，即可识别这个停留点。

基于上述讨论，本文提出一种新的基于区分策略的方法：基于密度的停留点识别（stay point identification based on density，SPID）方法。本文主要贡献如下。

- 考虑了轨迹点的时空聚集，即计算轨迹点的密度区间及密度，生成候选集。根据密度，迭代地在候选集中进行识别、更新操作，得到停留点集。
- 兼顾了轨迹点的时间连续性和方向性，即在计算轨迹点的密度区间及密度时，沿时间维从向后和向前两个方向，搜索时间连续且满足距离阈值的轨迹点。
- 设计了有效的基于密度的停留点识别方法，并通过在GeoLife数据集上的实验，验证了该方法的识别能力优于基准方法，可以进一步识别基准方法不能识别的两类停留点。

2 相关工作

现有的停留点识别方法可以分为3类：基于聚类策略的方法、基于概率策略的方法和基于区分策略的方法。本节将分别对这3类方法进行介绍。

2.1 基于聚类策略的方法

基于聚类策略的方法设计距离度量，采用聚类算法，聚类轨迹点为停留点，包括仅考虑空间因素的方法和同时考虑时空因素的方法。Ashbrook D等人[17]采用*K*-means聚类算法识别停留点；Toyama N等人[18]分析了聚类半径对停留点识别结果的影响；Zhou C Q等人[7]基于DBSCAN聚类算法提出停留点识别方法DJ-Cluster；Palma A T等人[8]在DBSCAN算法中引入时间因素，提出识别方法CB-SmoT；Zimmermann M等人[19]引入时间因素提升OPTICS聚类算法在停留点识别中的效果；Cao X等人[20]针对汽车轨迹数据，采用OPTICS（ordering points to identify the clustering structure）算法和*K*-means算法识别停留点。

2.2 基于概率策略的方法

基于概率策略的方法建立概率模型，从GPS轨迹数据中推导频繁访问的地点。Zhang K S等人[21]使用高斯混合模型，提出一种停留点在线学习算法。Liao L等人[22]提出了一种基于条件随机场的停留点识别算法。Nurmi P等人[10]提出了一种基于狄利克雷过程的非参数停留点识别方法。

2.3 基于区分策略的方法

基于区分策略的方法通过分析轨迹点在时间和空间上的差异，识别停留点及其代表点。Hariharan R等人[12]从锚点出发，沿时间维向前选择满足时间阈值的子轨迹，然后根据距离阈值判断子轨迹是否构成一个停留点，并将停留点中到其他轨迹点的最大距离最小化的轨迹点作为代表

点。Li Q N等人[14]从锚点出发，沿时间维向前选择满足距离阈值的子轨迹，然后根据时间阈值判断子轨迹是否构成了一个停留点，并采用停留点中轨迹点的平均位置点作为代表点。Liu J H等人[13]和Pérez-Torres R等人[15]分别采用Hariharan方法[12]和Li Q N等人的方法[14]预处理轨迹点序列，从中提取停留点。Pavan M等人[16]在Li Q N等人的方法[14]的基础上考虑了速度。

与上述方法不同，本文提出一种新的基于区分策略的方法——基于密度的停留点识别方法，该方法考虑了轨迹点的时空聚集，兼顾了轨迹点的时间连续性和方向性。

3 相关概念及问题描述

如**图3**所示，GPS轨迹点p(<latitude, longitude>, time)表示移动对象在时间time位于坐标<latitude, longitude>的位置，p.time表示轨迹点p的时间，p.longtitude表示经度，p.latitude表示纬度。将移动对象的轨迹点按时间有序连接，即得到移动对象的轨迹点序列Traj = $p_1 \rightarrow p_2 \rightarrow p_3 \rightarrow \cdots \rightarrow p_{n-1} \rightarrow p_n$。

定义1 轨迹点的密度区间

给定GPS轨迹点序列Traj，其中，任意轨迹点p_a的密度区间p_a.interval=[p_{as}, p_{ae}]是满足下列条件的连续子序列：

- $\forall p_i \in p_a$.interval, $d(p_a, p_i) \leqslant d_{th}$；
- $d(p_a, p_{as-1}) > d_{th}$，且$d(p_a, p_{ae+1}) > d_{th}$。

其中，p_{as}和p_{ae}分别表示密度区间的起点和终点，d_{th}是距离阈值，$d(\cdot)$是距离函数，本文采用的是球面距离。

定义2 轨迹点的密度

给定轨迹点p_a的密度区间p_a.interval=[p_{as}, p_{ae}]，p_a的密度p_a.density定义为其密度区间中的轨迹点数目，即：

$$p_a.\text{density} = ae - as + 1 \quad (1)$$

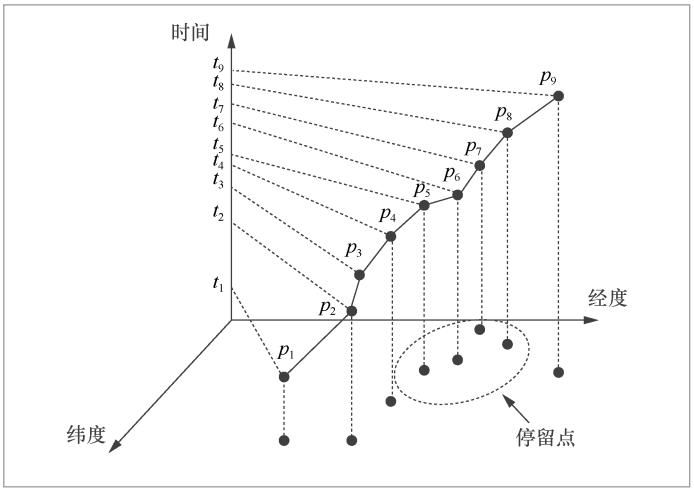


图 3 GPS 轨迹点、轨迹点序列和停留点

定义3 轨迹点的时间跨度

给定轨迹点p_a的密度区间p_a.interval=[p_{as}, p_{ae}]，p_a的时间跨度p_a.timespan定义为其密度区间起点与终点的时间差，即：

$$p_a.\text{timespan} = p_{ae}.\text{time} - p_{as}.\text{time} \quad (2)$$

事实上，轨迹点的密度是移动对象在以此点为中心、距离阈值为半径的区域内的轨迹点数目。在轨迹点采样频率一定的情况下，轨迹点密度越大，移动对象在该区域停留的时间越长。

定义4 停留点

给定GPS轨迹点序列Traj、距离阈值d_{th}和时间阈值t_{th}，停留点sp=[p_s, p_e]是满足以下条件的连续子序列：

- $\forall p_i, p_j \in \text{sp}, d(p_i, p_j) \leqslant 2d_{th}$；
- $p_e.\text{time} - p_s.\text{time} \geqslant t_{th}$。

例如，在**图3**中，如果连续轨迹点子序列[p_5, p_8] = $p_5 \rightarrow p_6 \rightarrow p_7 \rightarrow p_8$满足定义4的停留点条件，即其中任意两个轨迹点的距离小于或等于$2d_{th}$，起点p_5和终点p_8的时间差大于或等于t_{th}，则连续子序列[p_5, p_8]为一个停留点。

给定GPS轨迹点序列Traj、距离阈值d_{th}和时间阈值t_{th}，停留点识别的基本任务就是从Traj中找出尽可能多的、互不相交

的、满足定义4条件的停留点。

4 基于密度的停留点识别方法

本文所提的基于密度的停留点识别方法的处理框架如**图4**所示，主要包括两个步骤：密度计算和停留点识别。

4.1 密度计算

4.1.1 算法思想

在密度计算步骤中，依次以GPS轨迹点序列Traj中的每个轨迹点p_a为锚点，根据距离阈值d_{th}，沿时间维向后搜索和向前搜索，得到p_a的密度区间p_a.interval =[p_{as},p_{ae}]，基于密度区间计算p_a的密度p_a.density=ae−as+1和时间跨度p_a.timespan = p_{ae}.time−p_{as}.time，然后根据时间阈值，生成候选停留点列表。

(1)后向搜索

后向搜索是以轨迹点p_a为锚点，沿时间维向后搜索与p_a的距离小于或等于距离阈值d_{th}的在时间上连续的所有轨迹点，直至最后一个满足条件的轨迹点p_{as}，即搜索满足下列条件的轨迹点p_i：

- $0 \leqslant as \leqslant i < a$；
- $\forall i,\ as \leqslant i < a,\ d(p_a, p_i) \leqslant d_{th}$；
- 如果$0 \leqslant as-1$，则$d(p_a, p_{as-1}) > d_{th}$。

因此，后向搜索过程是从锚点p_a出发，沿时间维重复如下步骤（初始j=1）：

① 向后搜索一个轨迹点$p_i=p_{a-j}$，判断p_{a-j}是否已经超过GPS轨迹点序列Traj的起点，即判断$a-j$是否小于0，若是，则最后一个满足条件的轨迹点$p_{as}=p_{a-j+1}=p_0$，后向搜索过程停止，否则执行第②步。

② 判断p_{a-j}与锚点p_a的距离是否大于距离阈值d_{th}，即判断$d(p_a,p_{a-j})$是否大于d_{th}，若是，则最后一个满足条件的轨迹点$p_{as}=p_{a-j+1}$，后向搜索过程停止，否则执行第③步。

③ $j=j+1$，转第①步，继续搜索。

(2)前向搜索

前向搜索是以轨迹点p_a为锚点，沿时间维向前搜索与p_a的距离小于或等于距离阈值d_{th}的在时间上连续的所有轨迹点，直至最后一个满足条件的轨迹点p_{ae}，即搜索满足下列条件的轨迹点p_i：

- $a < i \leqslant ae \leqslant |Traj|-1$；
- $\forall i,\ a < i \leqslant ae,\ d(p_a, p_i) \leqslant d_{th}$；
- 如果$ae+1 \leqslant |Traj|-1$，则$d(p_a, p_{ae+1}) > d_{th}$。

因此，前向搜索过程是从锚点p_a出发，沿时间维重复如下步骤（初始j=1）：

① 向前搜索一个轨迹点$p_i=p_{a+j}$，判断p_{a+j}是否已经超过GPS轨迹点序列Traj的终点，即判断$a+j$是否大于|Traj|−1，若是，则最后一个满足条件的轨迹点$p_{ae}=p_{a+j-1}=p_{|Traj|-1}$，前向搜索过程停止，否则执行第②步。

② 判断p_{a+j}与锚点p_a的距离是否大于距离阈值d_{th}，即判断$d(p_a,p_{a+j})$是否大于d_{th}，若是，则最后一个满足条件的轨迹点$p_{ae}=p_{a+j-1}$，后向搜索过程停止，否则执行第③步。

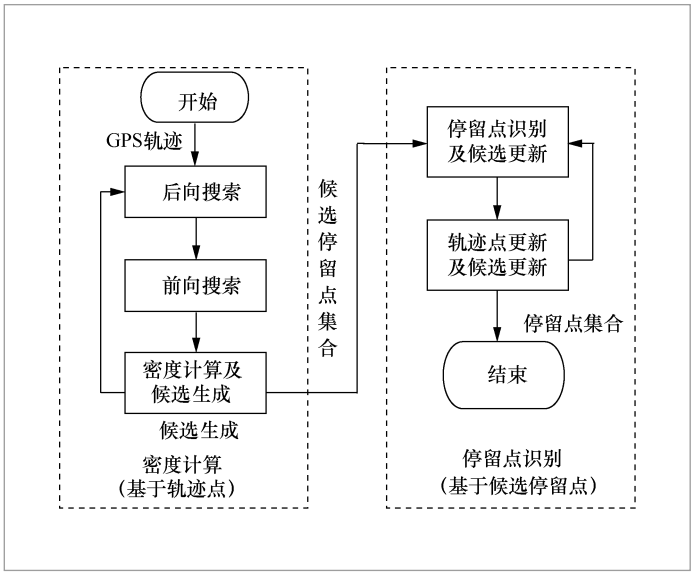


图 4 算法框架

③ $j=j+1$，转第①步，继续搜索。

（3）密度计算及候选生成

通过后向搜索和前向搜索，所有介于p_{as}和p_{ae}之间的轨迹点p_i(as≤i≤ae)即构成了锚点p_a的密度区间p_a.interval = [p_{as}, p_{ae}]，利用式（1）和式（2）即可计算p_a的密度p_a.density和时间跨度p_a.timespan。

如果锚点p_a的时间跨度小于时间阈值，即如果p_a.timespan<t_{th}，则p_a的密度区间p_a.interval = [p_{as}, p_{ae}]不可能是一个停留点，可以直接剪枝，否则p_a.interval = [p_{as}, p_{ae}]是一个候选停留点，将p_a及p_a.interval = [p_{as}, p_{ae}]放入按密度降序排列的候选列表CL中，即CL={(p_u, [p_{us}, p_{ue}]), …, (p_v, [p_{vs}, p_{ve}])}满足下列条件：

- ∀(p_a,[p_{as},p_{ae}])∈CL, p_a.timespan≥t_{th}；
- p_u.density≥…≥p_v.density。

在之后的停留点识别步骤中，将从候选列表CL中按密度从大到小的顺序识别停留点。

4.1.2 算法描述

密度计算Computing-Density的算法描述如算法1所示。

算法1：密度计算Computing-Density。

输入：GPS轨迹点序列Traj，距离阈值d_{th}，时间阈值t_{th}。

输出：候选停留点列表CL。

步骤：

1. **for each** p_a∈Traj
2. p_{as} := backwardSearching(p_a, d_{th});
3. p_{ae} := forwardSearching(p_a, d_{th});
4. p_a.interval := [p_{as}, p_{ae}];
5. p_a.density := ae-as+1;
6. p_a.timespan := p_{ae}.time-p_{as}.time;
7. **If** p_a. timespan ≥ t_{th} **then**
8. CL.SortInsert(p_a, [p_{as}, p_{ae}]);
9. **end for**
10. **return** CL

在算法1中，backwardSearching(p_a, d_{th})和forwardSearching(p_a, d_{th})实现前向搜索和后向搜索，CL.SortInsert(p_a,[p_{as}, p_{ae}])将候选停留点按密度降序插入候选列表CL中。

设GPS轨迹点序列的长度为n，轨迹点密度区间的平均长度为l。在算法1中，前向搜索和后向搜索的时间复杂性为$O(n\cdot l)$，密度计算及候选生成的主要开销是候选列表的排序，时间复杂性为$O(n\text{lb}n)$，通常$l<<n$，因此算法1的时间复杂性为$O(n\text{lb}n)$。

4.2 停留点识别

4.2.1 算法思想

候选列表CL中的候选停留点已经满足距离阈值和时间阈值，但是这些候选停留点的密度区间可能重叠，因此在停留点识别步骤中，将基于CL按密度从大到小迭代地进行识别更新操作，得到不相交的停留点，直至CL为空。

（1）停留点识别及候选更新

因为候选列表CL中的候选停留点已按密度降序排列，所以停留点识别及候选更新过程如下。

① 从CL中选取当前密度值最大的候选停留点，即选取CL中的第一个候选停留点p_u = [p_{us}, p_{ue}]，作为停留点sp= [p_{us}, p_{ue}]。

② 根据当前停留点sp=[p_{us}, p_{ue}]，对于所有介于起点p_{us}和终点p_{ue}之间的轨迹点p_i，如果p_i及其密度区间p_i.interval = [p_{is}, p_{ie}]在CL中，则将(p_i, [p_{is}, p_{ie}])从CL中删除。

（2）轨迹点更新及候选更新

当前停留点sp=[p_{us},p_{ue}]识别之后，还需更新所有受其影响的轨迹点的密度区间，进而还需再次更新候选列表CL，更新

过程如下。

① 对于CL中每个候选停留点p_i=[p_{is}, p_{ie}]，如果其与停留点sp = [p_{us}, p_{ue}]相交，则缩小p_i的密度区间p_i.interval = [p_{is}, p_{ie}]，即如果$p_{us} \leqslant p_{ie} \leqslant p_{ue}$，则ie=us-1；如果$p_{us} \leqslant p_{is} \leqslant p_{ue}$，则is=ue+1。

② 对于CL中每个缩小的候选停留点p_i=[p_{is},p_{ie}]，根据式(1)和式(2)，重新计算其密度p_i.density和时间跨度p_i.timespan，如果其时间跨度小于时间阈值，即如果p_i.timespan<t_{th}，则p_i = [p_{is}, p_{ie}]不再是一个候选停留点，可以剪枝，将(p_i, [p_{as}, p_{ae}])从CL中删除；否则按其密度p_i.density调整在CL中的位置。

4.2.2 算法描述

停留点识别Identifying-Staypoint的算法描述如算法2所示。

算法2：停留点识别Identifying-Staypoint。

输入：候选停留点列表CL，时间阈值t_{th}。

输出：停留点集合SP。

步骤：

```
1. while CL!=∅ do
2.   sp := CL[0].[p_us, p_ue];
3.   UpdatingOne(sp, CL);
4.   UpdatingTwo(sp, CL, t_th);
5.   SP.insert(sp);
6. end while
7. return SP
```

在算法2中，UpdatingOne(sp, CL)根据当前停留点，完成候选列表的第一次删除更新。UpdatingTwo(sp, CL, t_{th})根据当前停留点和时间阈值，缩小受影响轨迹点的密度区间，完成候选列表的第二次删除更新以及排序。SP.insert(sp)将当前停留点插入停留点集合。

设候选列表中初始候选停留点数目为m，候选列表更新次数为k。在算法2中，候选列表每次更新的时间复杂性为$O(m^2)$，每次排序的时间复杂性为$O(m\text{lb}m)$，因此算法2的时间复杂性为$O(km^2)$。

4.3 讨论

本文所提的基于密度的停留点识别方法是一种基于区分策略的方法，这类方法找到的停留点满足定义4的条件，即这类方法找到的停留点是正确的。但是这类方法不能保证找到所有的停留点，即这类方法是不完备的。

定理1 基于密度的停留点识别方法找到的停留点是正确的。

证明：(1) SPID方法找到的每个sp都是某个轨迹点p_a的密度区间p_a.interval = [p_{as}, p_{ae}]

$\because \forall p_i \in p_a$.interval=[$p_{as}$.$p_{ae}$]

$d(p_a,p_i) \leqslant d_t$

$\therefore \forall p_i, p_j \in p_a$.interval=[$p_{as}$.$p_{ae}$]

$d(p_i,p_j) \leqslant 2d_{th}$

即sp满足定义4中的条件1。

(2) SPID方法找到的每个sp都是从候选列表CL中选取的，而CL的初始生成以及迭代更新都对每个候选进行了时间阈值测试，使$\forall(p_a,[p_{as}, p_{ae}]) \in CL, |p_{ae}.\text{time}-p_{as}.\text{time}| \geqslant t_{th}$成立。即sp满足定义4中的条件2。

5 实验与分析

5.1 实验设计

基于区分策略的停留点识别方法能保证找到的停留点是正确的，但是不能保证找到所有的停留点，因此设计了如下实验。

(1) 实验目的

验证基于密度的停留点识别方法能否

找到更多的停留点。

(2)数据集

实验采用的数据集是来自微软亚洲研究院的GeoLife数据集。数据集采集了182名用户从2007年4月份到2012年8月份的GPS轨迹，轨迹数目共计17 621条，轨迹长度共计1 292 951 km，轨迹持续时间共计20 176 h。这些轨迹数据由不同GPS设备在不同采样频率下采集，91.5%的轨迹数据是在较高采样频率下采集的。

(3)对比方法

实验选取的对比方法是参考文献[14,16]提出的两个方法，分别以作者姓氏命名为Li方法和Pavan方法。Li方法的基本思想是：首先以GPS轨迹点序列的起始点为锚点，沿时间维向前选择满足距离阈值的子轨迹，根据时间阈值判断子轨迹是否构成一个停留点。若是，则以子轨迹后面的轨迹点为新锚点；否则以锚点后面的轨迹点为新锚点，然后重复上述过程，直至整个轨迹点序列遍历完成。Li方法是现有基于区分策略方法中表现最优的方法[5]。但是Li方法仅考虑了时间连续的一个方向。Pavan方法在判断子轨迹是否构成停留点中增加了速度阈值的限定，以排除无意义的停留点。

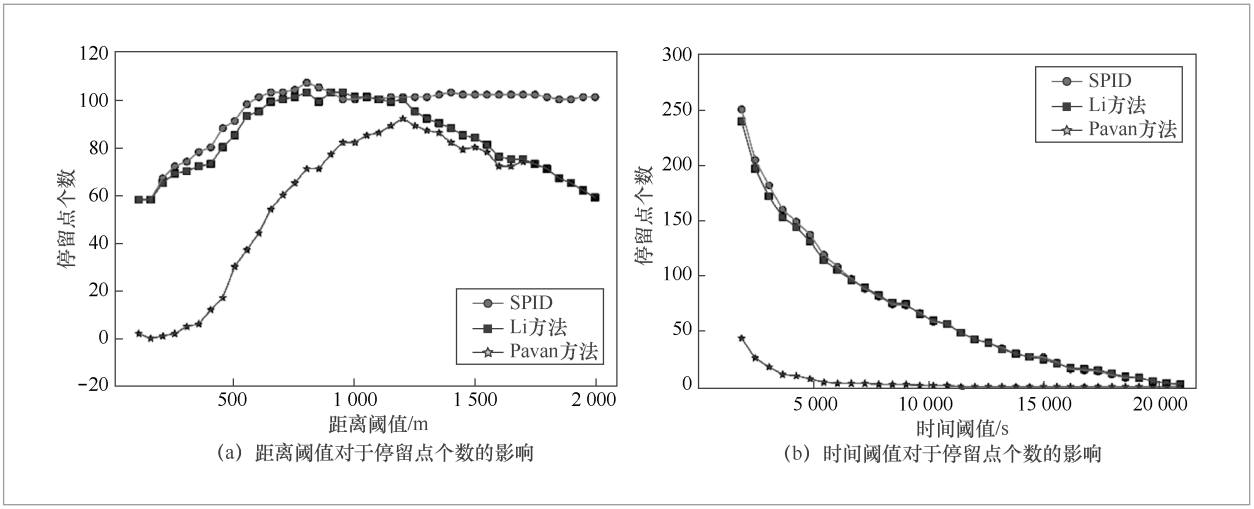
5.2 实验结果

首先，对比了3种方法中距离阈值和时间阈值对停留点个数的影响；然后，进一步分析了3种方法识别的停留点的差异。

5.2.1 阈值对于停留点个数的影响

实验对比了3种方法在不同距离阈值和时间阈值下的停留点个数。**图5(a)**显示了时间阈值t_{th}=1 800 s时，距离阈值对停留点个数的影响，**图5(b)**显示了距离阈值d_{th}=200 m时，时间阈值对停留点个数的影响。t_{th}=1 800 s和d_{th}=200 m是Li方法的最优阈值。Pavan方法的结果均是在速度阈值为2 m/s的情况下得到的。

如**图5(a)**所示，在绝大多数距离阈值情况下，SPID识别的停留点个数比对比方法多。当距离阈值小于1 250 m时，SPID和对比方法的停留点个数变化趋势都是随着距离阈值的增加而增加的。在距离阈值超过1 250 m之后，对比方法识别的停留点个数变化呈现随着距离阈值的增加而迅速下



(a) 距离阈值对于停留点个数的影响　(b) 时间阈值对于停留点个数的影响

图 5 阈值对于停留点个数的影响

降的趋势，而SPID方法则保持稳定。对比方法识别的停留点个数下降的原因是较大的距离阈值使每次选择的子轨迹变长，并且不断地对后续的子轨迹选择产生影响，这种影响的累积使得一些空间和时间上邻近的停留点被合并。而SPID方法由于从当前密度最大的候选开始，迭代地进行识别、更新，避免了这种合并。

如**图5(b)**所示，在绝大多数时间阈值情况下，SPID方法和Li方法识别的停留点个数比较接近。SPID方法和对比方法识别的停留点个数都随着时间阈值的增长而下降。当时间阈值超过21 600 s(6 h)时，识别的停留点个数接近于0，这是因为停留时间超过6 h是很少见的。

从**图5**还可以看出，在大多数阈值情况下，Pavan方法识别的停留点个数少于Li方法，这是因为其在识别过程中需要满足对于速度阈值的限定，因此它过滤了不满足速度阈值限定的停留点，比如用户在公园里跑步时形成的不满足速度阈值限定的停留点或者用户在景点内乘坐游览车观光时形成的不满足速度阈值限定的停留点。速度阈值使得算法识别出的停留点更加规整，但被速度阈值过滤掉的停留点依然对研究用户行为有参考价值。

5.2.2 不同方法的停留点比较

本节分析SPID方法能识别但对比方法不能识别的两类停留点。

第一类停留点如**图6**所示，在GPS轨迹中出现了“跳跃”。从**图6(a)**所示的GPS子轨迹可以看出，轨迹点3095到轨迹点3096的时间跨度远大于相邻轨迹点之间的时间跨度。从**图6(b)**所示的轨迹点相对位置可以看出，轨迹点3095到轨迹点3096的距离也远大于相邻轨迹点之间的距离。产生这种情形的原因可能是用户从一个门进入高楼或者地下建筑物，然后从另一个门出去，高楼和地下建筑物屏蔽了信号，使得这段轨迹产生了“跳跃”。在对比方法中，轨迹点3093、轨迹点3094和轨迹点3095会被依次选为锚点，由于它们与轨迹点3096的距离超过距离阈值，对比方法不能识别这个包含“跳跃”的停留点。而在SPID中，轨迹点3101会被选为锚点，并从两个方向搜索，由于它与轨迹点3095和轨迹点3096的距离都小于距离阈值，从而可以识别这个包含“跳跃”的停留点。

第二类停留点如**图7**所示，在GPS轨迹中出现了“暂时离开”。**图7(a)**为SPID识别的停留点，**图7(b)**和**图7(c)**分别为对比方法识别的两个停留点。事实上，**图7(a)**的子轨迹是**图7(b)**和**图7(c)**子轨迹的连接。从**图7(a)**可以看出，对比方法识别的停留点1的终点和停留点2的起点都与停留区域的中心相距较远，出现了“暂时离开”。在对比方法中，“暂时离开”使一个时间跨度较长的停留点被识别成两个在空间上重合、时间跨度较短的停留点。而在SPID中，由于从前向和后向两个方向搜索密度区间，并根据密度大小识别停留点，从而可以识别这种包括“暂时离开”的停留点。

5.2.3 示例

某用户的一段GPS轨迹点序列Traj见**表1**，距离阈值d_{th}=100 m，时间阈值t_{th}=300 s。

(1)密度计算

以轨迹点p_{220}为例，展示后向搜索、前向搜索、密度计算及候选列表生成的过程。

- 后向搜索：初始j=1，因为$a-j$=220-1=219>0，$d(p_{220}, p_{219})$=2.469<d_{th}=100，未达到停止条件，所以$j:=j+1$=2；重复此过程，直到j=11，因为$a-j$=220-11=209>0，

$d(p_{220}, p_{209})$=105.993>d_{th}=100，达到停止条件，所以后向搜索过程结束，轨迹点p_{as}=p_{210}作为锚点p_{220}密度区间的起点。

● 前向搜索：初始j=1，因为$a+j$=221<|Traj|−1=409，$d(p_{220}, p_{221})$=1.357<d_{th}=100，未达到停止条件，所以j:=j+1=2；重复此过程，直到j=38，因为$a+j$=220+38= 258<|Traj|−1=409，$d(p_{220}, p_{258})$=102.781>d_{th}=100，达到停止条件，所以前向搜索过程结束，轨迹点p_{ae}=p_{257}作为锚点p_{220}密度区间的终点。

● 密度计算：密度值p_{220}.density=257−210+1=48，时间跨度p_{220}.timespan=p_{ae}.time− p_{as}.time=3 002。

● 候选列表生成：由于p_{220}.timespan=3 002>t_{th}=300，锚点及其密度区间(p_{220}, [p_{210}, p_{257}])构成一个候选停留点，将(p_{220}, [p_{210}, p_{257}])按其密度值插入候选列表CL中的适当位置。至此，锚点p_{220}的密度计算过程结束。

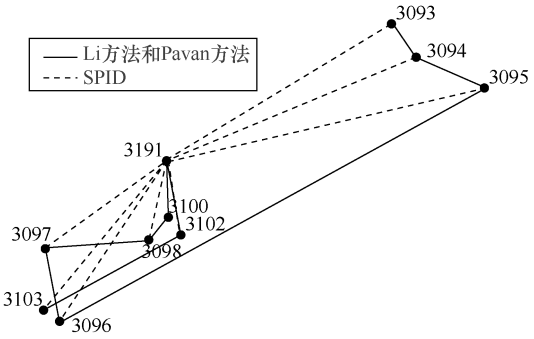
当对GPS轨迹点序列Traj中的所有轨迹点都计算密度后，可以得到按密度降序排列的初始候选列表CL，见**表2**。

(2) 停留点识别

对候选列表CL中的每个候选停留点进行停留点识别、轨迹点更新。以第一个

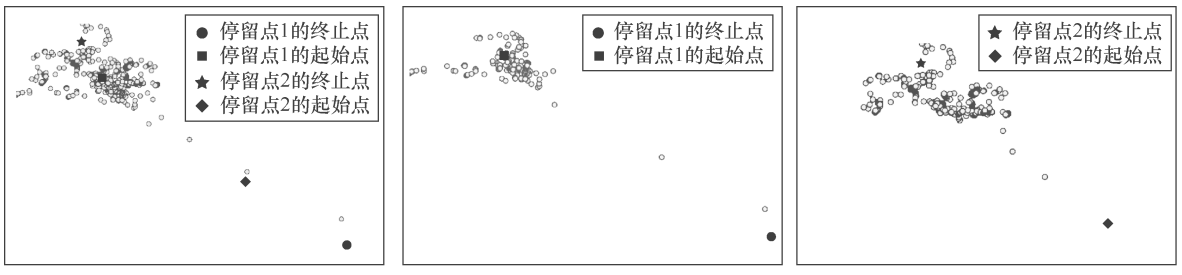
轨迹点	纬度	经度	日期	时间
3093	39.999 966	116.327 382	2009-06-17	06:21:11
3094	39.999 957	116.327 367	2009-06-17	06:21:16
3095	39.999 953	116.327 352	2009-06-17	06:21:21
3096	39.999 855	116.327 279	2009-06-17	09:33:37
3097	39.999 876	116.327 309	2009-06-17	09:33:42
3098	39.999 885	116.327 307	2009-06-17	09:33:47
3099	39.999 885	116.327 307	2009-06-17	09:33:50
3100	39.999 893	116.327 315	2009-06-17	09:33:52
3101	39.999 910	116.327 338	2009-06-17	09:33:57
3102	39.999 889	116.327 308	2009-06-17	09:34:02
3103	39.999 858	116.327 285	2009-06-17	09:33:07

(a) GPS子轨迹数据



(b) 相对位置

图 6 包含“跳跃”的停留点



(a) SPID识别的停留点 (b) 基准方法识别的停留点1 (c) 基准方法识别的停留点2

图 7 包含“暂时离开”的停留点

表1 GPS轨迹点序列

序号	纬度	经度	时间
0	40.007 219	116.319 595	2009-05-18 06:46:48
…	…	…	…
209	39.991 537	116.331 743	2009-05-18 08:29:04
210	39.991 524	116.331 918	2009-05-18 08:29:09
211	39.991 540	116.332 044	2009-05-18 08:29:14
…	…	…	…
219	39.991 312	116.332 921	2009-05-18 08:29:54
220	39.991 304	116.332 948	2009-05-18 08:29:59
221	39.991 312	116.332 960	2009-05-18 08:30:04
…	…	…	…
256	39.991 530	116.331 946	2009-05-18 09:19:06
257	39.991 562	116.331 887	2009-05-18 09:19:11
258	39.991 557	116.331 789	2009-05-18 09:19:16
…	…	…	…
409	40.001 156	116.326 798	2009-05-18 09:37:51

表2 初始候选列表

锚点	起点	终点
220	210	257
…	…	…
242	212	257
…	…	…
367	358	404
…	…	…
69	60	98
…	…	…
101	74	106

表3 更新后的候选列表

锚点	起点	终点
367	358	404
…	…	…
69	60	98
…	…	…
101	74	106

候选停留点(p_{220}, [p_{210}, p_{257}])为例，展示停留点识别及候选列表更新、轨迹点更新及候选列表更新的过程。

停留点识别及候选列表更新：当前候选列表CL中的第一个候选停留点(p_{220}, <p_{210},p_{257}>)已经满足停留点的条件，故构成了一个停留点sp=(p_{220}, <39.991 304, 116.332 948>, [2009-05-18 08:29:09, 2009-05-18 09:19:11], [210, 257])。依次判定介于起点p_{us}=p_{210}和终点p_{ue}=p_{257}之间的轨迹点，将出现在候选列表CL中的候选轨迹点p_{220}、p_{221}、p_{224}、p_{223}、p_{222}、p_{225}、p_{242}、p_{241}、p_{240}、p_{238}、p_{239}、p_{237}、p_{236}、p_{235}、p_{234}、p_{233}、p_{229}、p_{227}、p_{232}、p_{231}、p_{228}、p_{230}、p_{226}及其密度区间从CL中删除。

轨迹点更新及候选列表更新：当前候选列表CL中没有候选轨迹点的邻域与p_{220}的密度区间[210, 257]相交，故CL中的候选轨迹点及其密度区间不需更新，CL也不需要更新。在对候选停留点(p_{220}, [p_{210}, p_{257}])进行判定之后，候选列表CL更新后的结果见**表3**。

在经过3次迭代之后，候选列表CL变为空，其中，大部分候选停留点由于与停留点相交，被更新策略删除了。最终得到的停留点集合见**表4**。

表 4 停留点集合

代表点	坐标	停留时间	密度区间
p_{220}	<39.991 304, 116.332 948>	[2009-05-18 08:29:9,2009-05-18 09:19:11]	[210,257]
p_{367}	<39.999 844, 116.326 921>	[2009-05-18 09:27:32,2009-05-18 09:37:26]	[358,404]
p_{69}	<40.000 392, 116.326 610>	[2009-05-18 06:51:48,2009-05-18 08:17:54]	[60,98]

6 结束语

本文在考虑轨迹点时空聚集的同时，考虑了轨迹点的时间连续性和方向性，提出了一种新的基于密度的识别停留点方法。首先，以锚点为中心，沿时间维从向后和向前两个方向搜索时间连续且满足距离阈值的轨迹点，形成锚点的密度区间；然后，根据时间阈值生成候选集，再根据密度迭代地在候选集中进行识别、更新操作，得到停留点集；最后，设计有效的算法，并通过实验验证了新方法是有效的，可以识别基准方法不能识别的两类停留点。在未来的工作中，将进一步研究停留点的语义标注以及基于具有语义的停留点研究位置服务。

参考文献:

[1] HERDER E, SIEHNDEL P, KAWASE R. Predicting user locations and trajectories[M]. Heidelberg: Springer International Publishing, 2014: 86-97.

[2] YANG Y, ZHENG Y, CHEN Y K, et al. Mining individual life pattern based on location history[C]//The 2009 10th International Conference on Mobile Data Management: Systems, Services and Middleware, May 18-20, 2009, Taipei, China. Washington, DC: IEEE Computer Society, 2009: 1-10.

[3] HORANONT T, PHITHAKKITNUKOON S, SHIBASAKI R. Sensing urban density using mobile phone gps locations: a case study of Odaiba Area, Japan[M]//Nature of Computation and Communication. Heidelberg: Springer International Publishing, 2014: 146-155.

[4] ZHENG V W, ZHENG Y, XIE X, et al. Collaborative location and activity recommendations with GPS history data[C]//The 19th International Conference on World Wide Web, April 26-30, 2010, Raleigh, USA. New York: ACM Press, 2010: 1029-1038.

[5] LEE C, YOON G, HAN D. A probabilistic place extraction algorithm based on a superstate model[J]. IEEE Transactions on Mobile Computing, 2013, 12(5): 945-956.

[6] ESTER M, KRIEGEL H P, XU X W. A density-based algorithm for discovering clusters a density-based algorithm for discovering clusters in large spatial databases with noise[C]//The 2nd International Conference on Knowledge Discovery and Data Mining, August 2-4, 1996, Portland, Oregon. Palo Alto: AAAI Press, 1996: 226-231.

[7] ZHOU C Q, FRANKOWSKI D, LUDFORD P, et al. Discovering personally meaningful places[J]. ACM Transactions on Information Systems, 2007, 25(3): 12.

[8] PALMA A T, BOGORNY V, KUIJPERS B, et al. A clustering-based approach for discovering interesting places in trajectories[C]//The 2008 ACM Symposium on Applied Computing, March 16-20, 2008, Fortaleza, Brazil. New York: ACM Press, 2008: 863-868.

[9] ZHANG K S, LI H F, TORKKOLA K, et al.

Adaptive learning of semantic locations and routes[C]//The 3rd International Conference on Location-And Context-Awareness, September 20-21, 2007, Oberpfaffenhofen, Germany. Heidelberg: Springer-Verlag, 2007: 193-210.

[10] NURMI P, BHATTACHARYA S. Identifying meaningful places: the non-parametric way[C]//The 6th International Conference on Pervasive Computing, May 19-22, 2008, Sydney, Australia. Heidelberg: Springer-Verlag, 2008: 111-127.

[11] LIAO Z X, LI S C, PENG W C, et al. On the feature discovery for App usage prediction in smartphones[C]//The 13th International Conference on Data Mining, December 7-10, 2013, Dallas, USA. Piscataway: IEEE Press, 2013: 1127-1132.

[12] HARIHARAN R, TOYAMA K. Project lachesis: parsing and modeling location histories[C]//Geographic Information Science, Third International Conference, GIScience 2004, October 20-23, 2004, Adelphi, USA. Heidelberg: Springer, 2004: 106-124.

[13] LIU J H, WOLFSON O, YIN H B. Extracting semantic location from outdoor positioning systems[C]//The 7th International Conference on Mobile Data Management, May 10-12, 2006, Nara, Japan. Washington, DC: IEEE Computer Society, 2006: 73.

[14] LI Q N, ZHENG Y, XIE X, et al. Mining user similarity based on location history[C]//The 16th ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems, November 5-7, 2008, Irvine, USA. New York: ACM Press, 2008: 1-10.

[15] PÉREZTORRES R, TORRES-HUITZIL C, GALEANAZAPIÉN H. Full on-device stay points detection in smartphones for location-based mobile applications[J]. Sensors, 2016, 16(10): 1693.

[16] PAVAN M, MIZZARO S, SCAGNETTO I, et al. Finding important locations: a feature-based approach[C]//IEEE International Conference on Mobile Data Management, June 15-18, 2015, Pittsburgh, USA. Washington, DC: IEEE Computer Society, 2015: 110-115.

[17] ASHBROOK D, STARNER T. Using GPS to learn significant locations and predict movement across multiple users[J]. Personal & Ubiquitous Computing, 2003, 7(5): 275-286.

[18] TOYAMA N, OTA T, KATO F, et al. Exploiting multiple radii to learn significant locations[C]//The 1st International Workshop, International Symposium on Location- and Context-Awareness, May 12-13, 2005, Oberpfaffenhofen, Germany. Heidelberg: Springer, 2005: 157-168.

[19] ZIMMERMANN M, KIRSTE T, SPILIOPOULOU M. Finding stops in error-prone trajectories of moving objects with time-based clustering[C]// International Conference on Intelligent Interactive Assistance and Mobile Multimedia Computing, November 9-11, 2009, Rostock-Warnemünde, Germany. Heidelberg: Springer, 2009: 275-286.

[20] CAO X, CONG G, JENSEN C S. Mining significant semantic locations from GPS data[J]. Proceedings of the Vldb Endowment, 2010, 3(1): 1009-1020.

[21] ZHANG K S, LI H F, TORKKOLA K, et al. Adaptive learning of semantic locations and routes[C]//The 3rd International Conference on Location-And Context-Awareness, September 20-21, 2007, Oberpfaffenhofen, Germany. Heidelberg: Springer-

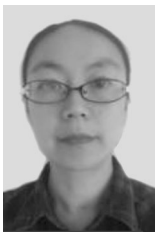
Verlag, 2007: 193-210.
[22] LIAO L, FOX D, KAUTZ H. Extracting places and activities from gps traces using hierarchical conditional random fields[J]. International Journal of Robotics Research, 2007, 26(1): 119-134.

作者简介

李毓瑞(1989-),男,云南大学信息学院硕士生,主要研究方向为空间数据挖掘。

陈红梅(1976-),女,博士,云南大学信息学院副教授,主要研究方向为数据挖掘、空间数据挖掘等。

王丽珍(1962-),女,博士,云南大学信息学院教授,博士生导师,主要研究方向为数据库、数据挖掘、计算机算法等。



肖清(1975-),女,云南大学信息学院讲师,主要研究方向为数据挖掘、空间数据挖掘等。

收稿日期: 2018-08-15
通信作者: 陈红梅, hmchen@ynu.edu.cn
基金项目: 国家自然科学基金资助项目(No.61662086, No.61472346, No.61762090);云南省自然科学基金资助项目(No.2015FB14, No.2016FA026);云南大学"东陆中青年骨干教师"培养计划(No.WX069051)
Foundation Items: The National Natural Science Foundation of China(No.61662086, No.61472346, No.61762090), The National Natural Science Foundation of Yunnan Province(No.2015FB14, No.2016FA026), The Program for Young and Middle-aged Skeleton Teachers of Yunnan University (No.WX069051)