# 基于偏移量计算的在线GPS轨迹数据压缩

樊庆富,张 磊,刘磊军,鲍苏宁,房 晨
FAN Qingfu, ZHANG Lei, LIU Leijun, BAO Suning, FANG Chen

中国矿业大学 计算机科学与技术学院,江苏 徐州 221116
School of Computer Science and Technology, China University of Mining and Technology, Xuzhou, Jiangsu 221116, China

**FAN Qingfu, ZHANG Lei, LIU Leijun, et al. Online GPS trajectory data compression based on offset calculation. Computer Engineering and Applications, 2017, 53(8):254-259.**

**Abstract:** In view of the fact that the existing online Global Positioning System(GPS) trajectory data compression algorithm based on the offset calculation cannot effectively be applied to the key point insufficient evaluation, this paper proposes a new online trajectory data compression algorithm based on the offset calculation named Key-Predecessor Fix Algorithm (KPFA). This algorithm calculates the Synchronization Euclidean Distance(SED)and the accumulated offset to find out the key points with large amount of information. At the same time, it sets the threshold to correct the trajectory points between the current key point and the last key point to keep the trajectory information more completely. The experimental results show that compared with the Opening Window Time Ratio(OPW-TR)and Spatial Quality Simplification Heuristic Efficient(SQUISH-E) algorithms the average SED error of KPFA is the smallest and the running time is the fastest maintenance in 100, 000 ms when the compression ratio is the same. The KPFA has higher accuracy on the amount of information of the trajectory point evaluation and the running time is more stable.
**Key words:** online trajectory compression; synchronous Euclidean distance; offset calculation; evaluation error

**摘 要:**针对现有基于偏移量计算的在线GPS轨迹数据压缩算法不能有效评估关键点的问题,提出基于偏移量计算的在线GPS轨迹数据压缩算法——关键点前继修正算法(KPFA)。该算法通过计算同步欧式距离(SED)累积偏移量来发现轨迹点中信息量较大的关键点,同时设置阈值对关键点之前和上一个关键点之后的轨迹点进行修正,更好地保留轨迹信息。实验结果表明,和按时间比例的开窗算法(OPW-TR)及启发式空间质量简化算法的改进算法(SQUISH-E)相比,压缩率相同时KPFA的平均SED误差最小,并且运行时间最快且维持在100 000 ms。KPFA算法对轨迹点的信息量评估准确度更高,运行时间更稳定。
**关键词:**在线轨迹压缩;同步欧式距离;偏移量计算;评估误差
**文献标志码:**A **中图分类号:**TP312 doi:10.3778/j.issn.1002-8331.1510-0297

## 1 引言

近年来随着全球定位系统(Global Positioning System,GPS)、智能移动设备(Smart Mobile Device,SMD)等定位设备的发展和成熟,使得基于时间和空间特性的轨迹数据的采集和存储呈现出爆炸式的增长[1]。GPS轨迹数据的数据量巨大给信息挖掘和利用造成很大的困难,因此,GPS轨迹数据压缩成为目前研究的热点[2]。GPS轨迹数据压缩指的是利用检测消除GPS轨迹点中

**基金项目:**中央高校基本科研业务费专项资金(No.2014XK10);教育部博士点基金(No.20110095110010);江苏省自然科学基金(No.BK20130208);中国矿业大学大学生创新创业基金资助大学生创新项目(No.DC201641)。
**作者简介:**樊庆富(1992—),男,硕士研究生,主要研究方向:移动对象轨迹数据挖掘,E-mail:fan_qingfu@163.com;张磊(1977—),男,博士,副教授,主要研究方向:移动对象轨迹数据挖掘;刘磊军(1991—),男,硕士研究生,主要研究方向:移动对象轨迹数据挖掘;鲍苏宁(1991—),男,硕士研究生,主要研究方向:移动对象轨迹数据挖掘;房晨(1991—),男,硕士研究生,主要研究方向:智能信息处理。
**收稿日期:**2015-10-29 **修回日期:**2016-03-01 **文章编号:**1002-8331(2017)08-0254-06
CNKI网络优先出版:2016-03-25, http://www.cnki.net/kcms/detail/11.2127.TP.20160325.2017.038.html

的冗余点，同样的轨迹数据，在经过消除冗余的压缩后，其潜在的数据挖掘速度得到极大的提高[3]。对于GPS轨迹数据的有损压缩，相比无损压缩可以更大程度地减少数据量的存储[4-5]，其实际应用价值更大。而在线GPS轨迹数据压缩[6]能够实现零延时同步压缩是当今互联网普及的大背景下更适合作为实际应用算法的解决方案。

基于偏移量计算的在线GPS轨迹数据压缩，是直接以同步欧氏距离评定轨迹点的信息量大小并定位关键点的算法。这种算法的特点是可以较好地保存轨迹的空间信息，使用简化后的轨迹就可以通过时间戳定位到和原轨迹相同的位置。Meratnia等[7]提出的按时间比例的自顶向下算法(Top-Down Time Ratio，TD-TR)是一种基于偏移量计算的离线GPS轨迹数据压缩算法，它以同步欧氏距离完全代替垂直距离的方法对经典的道格拉斯-普克算法(Douglas-Peucker，DP)[8]进行改进，其简化效果非常理想，在以后的在线GPS轨迹数据压缩算法的提出和改进中都有TD-TR的影子。Keogh EJ提出的开窗算法(Opening Window，OPW)[9]，Jonathan Muckell等人提出的Spatial QUalIty Simplification Heuristic(SQUISH)算法[10]和改进后的算法SQUISH-E[11]以及算法[12-14]都属于基于偏移量计算的轨迹数据压缩算法。开窗算法提出窗口的概念，在包含轨迹中的一部分点的窗口中进行迭代，使窗口不断更新，完成轨迹简化，算法的优点是可以进行同步在线压缩。改进后的按时间比例的开窗算法(Opening Window Time Ratio，OPW-TR)使用同步欧氏距离(Synchronous Euclidean Distance，SED)代替垂直距离，考虑了GPS轨迹中的时间信息。SQUISH算法提出了缓冲区概念，将取得的点全部放入一个缓冲区，缓冲区的大小会随着轨迹点的增多而动态调整，使误差不至于过大。该算法的优点是可以设置需要的压缩率。而改进后的SQUISH-E算法不仅可以设置压缩率还可以设置误差的阈值。开窗算法和SQUISH-E算法是简化效果较好的在线压缩算法，代表了这类算法的两种研究方向。其主要区别在于确认轨迹点中新的关键点后对轨迹点信息量的修正方法不同。开窗算法在确定一个轨迹点后，会通过重新计算来评估接下来的轨迹点，而SQUISH-E算法是计算所有轨迹点的信息量，删除非关键点，然后重新分配非关键点的信息，以达到修正轨迹点信息量。

上述基于偏移量计算的在线GPS轨迹压缩算法都是通过部分临近点评估单个点的信息，这样可以定位到信息量较大的关键点，但是一些信息量相对较小的关键点仍然有可能漏掉。为了减少评估时产生的误差，提出一种基于偏移量计算的在线GPS轨迹数据压缩算法——关键点前继修正算法，通过计算同步欧式距离(SED)累积偏移量来评估轨迹点中信息量较大的关键点，随后对当前关键点和上一个关键点之间的轨迹点使用回溯修正的方法：迭代计算两个关键点之间所有轨迹点的SED，并与同步欧式距离规定阈值进行比较，保留SED大的轨迹点作为新的关键点，对可能漏掉的关键点重新评定，以进一步提高信息量评估准确度。

## 2　相关概念

文中用到的度量轨迹信息时使用的定义：同步欧式距离、同步欧式距离的基准点、轨迹点中的关键点如下。

**定义1**(同步欧氏距离)　同步欧氏距离(Synchronous Euclidean Distance，SED)是指原路径上的点和这个点在简化后的路径上按时间比例对应的点之间的欧氏距离。图1中 $pp'$ 即同步欧氏距离，$pq$ 为垂直距离。

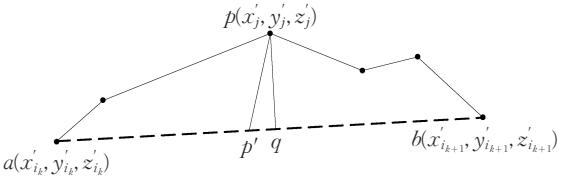


**图1　SED和垂直距离**

对于原路径 $P$ 上的点 $p$，它在简化路径 $P'$ 上的对应点 $p'$ 通过以下公式计算其坐标：

$$x_j'=x_{i_k}'+\frac{t_j'-t_{i_k}'}{t_{i_{k+1}}'-t_{i_k}'}\cdot(x_{i_{k+1}}'-x_{i_k}'),\ t_{i_k}'<t_j'<t_{i_{k+1}}' \tag{1}$$

$$y_j'=y_{i_k}'+\frac{t_j'-t_{i_k}'}{t_{i_{k+1}}'-t_{i_k}'}\cdot(y_{i_{k+1}}'-y_{i_k}'),\ t_{i_k}'<t_j'<t_{i_{k+1}}' \tag{2}$$

通过 $p'$ 的位置计算 $p$ 与 $p'$ 的同步欧氏距离：

$$SED(p_i,p_i')=\sqrt{(x_i-x_i')^2+(y_i-y_i')^2} \tag{3}$$

**定义2**(同步欧氏距离的基准点)　同步欧氏距离的基准点是指计算同步欧氏距离轨迹点的前继点和后继点中各自简化后的轨迹中第一个保留点。

**定义3**(轨迹点中的关键点)　关键点指轨迹中信息量较大，需要被保留的点。不同的算法对于信息量的定义不同，相应的对于关键点的定义也不相同，同一种算法由于参数设置的不同，其关键点也会出现相应的变动。

## 3　关键点前继修正算法

关键点前继修正算法(Key-Predecessor Fix Algorithm，KPFA)是通过计算同步欧式距离(SED)累积偏移量来发现轨迹点中信息量较大的关键点，随后对关键点之前和上一个关键点之后的轨迹点回溯修正：迭代计算两个关键点之间所有轨迹点的SED，并与同步欧式距离规定阈值进行比较，保留SED大的轨迹点作为新的关键点，对可能漏掉的关键点重新评定。算法步骤如下：

(1)将轨迹第一个点选为关键点，等待后续点的到来。

(2)以第一个点和第三个点为基准点,求第二个点的同步欧氏距离,将第二个点的同步欧氏距离作为其权重值,并将第二个点作为当前点。

(3)如果当前点的权重值低于规定阈值,则继续步骤(4)。如果当前点的权重值高于规定阈值,则跳到步骤(5)。

(4)将当前点的下一个点作为当前点。当前点的权重值计算为上一个当前点的权重值加上当前点的同步欧氏距离(以相邻点为基准点),继续步骤(3)。

(5)当前点为新的关键点。以上一个关键点为起点,新的关键点为终点,使用修正算法(Fix Algorithm,FA)找出其中的关键点。将当前点的下一个点作为当前点,当前点的权重值为当前点的同步欧氏距离(以相邻点为基准点),继续步骤(3)。

算法如下:

**算法** 1 关键点前继修正(KPFA)算法

**输入** 同步欧氏距离阈值 $sed$;原轨迹 $P=\{p_1,p_2,\cdots,p_n\}=\{(x_1,y_1,t_1),\cdots,(x_n,y_n,t_n)\}$

**输出** 简化轨迹 $P'=\{p'_{i1},p'_{i2},\cdots,p'_{im}\}=\{(x'_{i1},y'_{i1},t'_{i1}),\cdots,(x'_{im},x'_{im},t'_{im})\}$

```
BEGIN:
(1)value=0.0
(2)for each point p in P do
(3)  if p is the first then
(4)    set p as the key point
(5)    continue
(6)  end
(7)  dist=SED(p)(基准点为p-1和p+1)
(8)  value=value+dist
(9)  if value>sed then
(10)   FA(last key,p,sed)
(11)   set p as the key point
(12)   value=0.0
(13)   continue
(14)  end
(15) end
END
```

算法开始将权重值设为0(步骤(1)),将第一个点保存为关键点(步骤(3)~(6)),set $p$ as the key point表示将 $p$ 点保存为关键点。如果点 $p$ 非轨迹首点,则以其相邻点未基准点计算同步欧氏距离(步骤(7)),将同步欧氏距离和积累的权重值相加作为点 $p$ 的权重值(步骤(8)),如果权重值小于阈值 $sed$ 则重复执行步骤(3)~(14)的逻辑,如果权重值大于阈值 $sed$ ,则修正上一个关键点 $last\_key$ 和点 $p$ 之间的点,并将 $p$ 设为关键点(步骤(11)),将权重值归0(步骤(12)),然后重复执行步骤(3)~(14)的逻辑。

修正算法对可能漏掉的一些信息量相对较小的关键点重新评定:迭代计算两个关键点之间所有轨迹点的 $SED$ ,并与规定阈值进行比较,保留 $SED$ 大于规定阈值的轨迹点作为新的关键点。修正算法如下:

**算法** 2 修正算法(FA)

**输入** 需要修正的轨迹起点 $start$ ;需要修正的轨迹终点 $end$ ;同步欧氏距离阈值 $sed$

**输出** 起点 $start$ 和终点 $end$ 之间的所有关键点

```
BEGIN:
(1)max_dist=0.0
(2)for each point p in (start,end) do
(3)  dist=SED(p)(基准点为start和end)
(4)  if dist>max_dist then
(5)    max_dist=dist
(6)    max_point=p
(7)  end
(8)  end
(9)  if max_dist>sed then
(10)   FA(start,max_point,sed)
(11)   set max_point as the key point
(12)   FA(max_point,end,sed)
(13) end
END
```

修正算法首先求出起点和终点之间同步欧氏距离最大的点(步骤(2)~(8)),如果此点的同步欧氏距离大于规定阈值,则按顺序执行以下逻辑:对起点到此点的开区间内的点使用修正算法,将此点保存为关键点,对此点到终点的开区间内的点使用修正算法。如果此点的同步欧氏距离小于规定阈值则算法结束。

算法运行示例如图2所示:

(1) $A$ 为轨迹第一个点,为关键点。以 $A$ 和 $C$ 为基

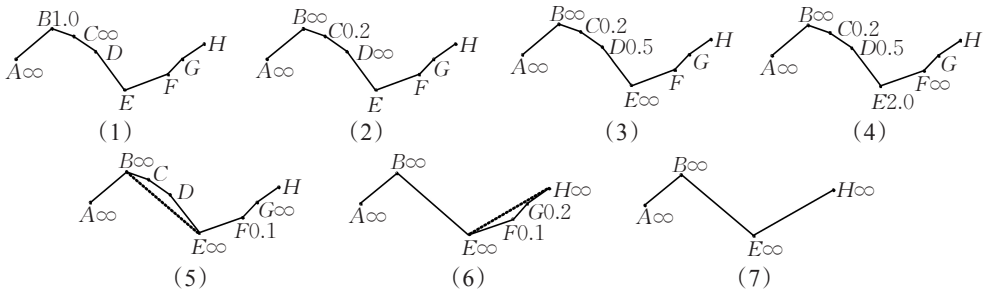


**图2　关键点前继修正算法**

准点求 $B$ 的同步欧氏距离，结果为1.0，因为 $B$ 是关键点后第一个点，所以其权重值为1.0。大于规定阈值，$B$ 为关键点。

(2)以 $B$ 和 $D$ 为基准点求 $C$ 的同步欧氏距离，结果为0.2，因为 $C$ 是关键点后第一个点，所以其权重值为0.2，小于规定阈值。

(3)以 $C$ 和 $E$ 为基准点求 $D$ 的同步欧氏距离，结果为0.3，其权重值为0.5(0.2+0.3)，小于规定阈值。

(4)以 $D$ 和 $F$ 为基准点求 $E$ 的同步欧氏距离，结果为1.5，其权重值为2.0(0.5+1.5)，大于规定阈值，$E$ 为关键点。

(5)产生了新的关键点，并且新的关键点和前一个关键点之间有其他点，需要进行修正(虚线连接部分)。以 $E$ 和 $G$ 为基准点求 $F$ 的同步欧氏距离，结果为0.1，其权重值为0.1，小于规定阈值。

(6)以 $F$ 和 $H$ 为基准点求 $G$ 的同步欧氏距离，结果为0.1，其权重值为0.2(0.1+0.1)，小于规定阈值。$H$ 为轨迹最后一个点，为关键点，修正 $E$ 和 $H$ 之间的点(虚线连接部分)。

(7)简化最终结果。

## 4　实验及分析

基于关键点前继修正算法，开发了GPS轨迹数据压缩原型系统。实验程序的编写使用的是C++语言，IDE使用的是微软公司的Microsoft Visual Studio；处理器为Intel Core i5-3470 3.20 GHz，3 201 MHz，4核，4个逻辑处理器；物理内存为4 GB。

实验数据集采用的是微软亚洲研究院Geolife数据集[15]，包括了2007年4月到2012年8月期间共收集了182名志愿者的GPS轨迹数据，总量达到17 621条轨迹，总距离达到129 295 km，总时长达到50 176 h，Geolife数据集包含信息巨大，本文设计的实验不需要全部信息，因此对数据集进行预处理筛选后形成本文实验的数据集，每条轨迹至少含有100个轨迹点，一共有15 659条轨迹，22 542 203个轨迹点，1.01 GB的数据量。

### 4.1　对比算法

本文使用按时间比例的自顶向下算法(TD-TR)、按时间比例的开窗算法(OPW-TR)和SQUISH-E算法作为关键点前继修正算法(KPFA)的对比算法。其中TD-TR算法虽然是离线压缩算法，其优秀的压缩结果可以作为一个参照。SQUISH-E算法由于可以设置压缩倍率和同步欧氏距离两个参数，本文将SQUISH-E算法设置为两种，第一种将其压缩率设置为1，表示为 SQUISH-E(sed)，第二种将其同步欧氏距离阈值设置为0，表示为SQUISH-E(ratio)，当把压缩率设置为1时，算法起作用的是其离线部分，所以SQUISH- E(sed)的类型为离线算法。算法详细情况见表1。

**表1　算法详细情况**

| 算法 | 类型 | 参数 | 时间复杂度 |
|---|---|---|---|
| TD-TR | 离线 | sed | $O(n^2)$ |
| OPW-TR | 在线 | sed | $O(n^2)$ |
| SQUISH-E(sed) | 离线 | sed | $O(n\mathrm{lb}n)$ |
| SQUISH-E(ratio) | 在线/离线 | ratio | $O(n\mathrm{lb}(n/ratio))$ |
| KPFA | 在线 | sed | $O(n^2)$ |

### 4.2　实验分析

实验中需要用到的压缩率、平均SED误差，定义如下。

**定义4**(压缩率) 实验中使用的压缩率是指压缩后的点数占压缩前点数的比例。仅考虑压缩率时，压缩率数值越小效果越好。

**定义5**(平均SED误差) 计算为以下公式：

$$\overline{SED}=(\sum_{i=1}^{n}SED(p_i))/n \tag{4}$$

公式(4)中 $p_i$ 指原轨迹上的第 $i$ 个点，$n$ 指原轨迹的点数，$SED(p_i)$ 指以 $p_i$ 在简化轨迹上的前继和后继为基准点计算 $p_i$ 的同步欧氏距离。

如图3所示，$A$、$C$、$E$、$H$ 四点因为在简化后轨迹中保留了下来，所以其同步欧氏距离误差计算为0，其他点中 $B$ 点以 $A$ 和 $C$ 为基准点、$D$ 点以 $C$ 和 $E$ 为基准点、$F$ 和 $G$ 以 $E$ 和 $H$ 为基准点分别计算同步欧氏距离。平均SED误差计算为同步欧氏距离之和与点数的比值。

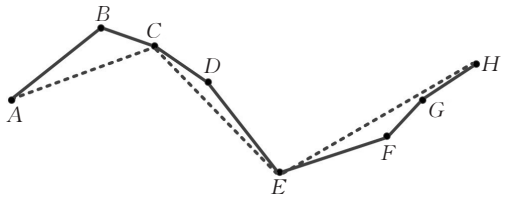


**图3　平均SED误差**

本文使用同步欧氏距离阈值和压缩率作为算法参数，前者选取1 m、2 m、3 m、4 m、5 m、6 m、10 m、20 m、40 m作为参数。适用算法为TD-TR、OPW-TR、SQUISHE(sed)和KPFA。后者选取2，3，4，10作为参数，适用算法为SQUISH(ratio)。

表2为以SED阈值为参数的实验结果，表3为以压缩倍率为参数的实验结果。从表2和表3中可以看出算法本身由于参数的不同其压缩率和平均SED误差有浮动，算法随着SED阈值或压缩倍率的增加压缩率逐渐减小，平均SED误差逐渐增大。总体来说，在相同参数的情况下，压缩率TD-TR<KPFA<SQUISH-E(sed)<SQUISH-E(ratio)<OPW-TR。可以看出KPFA算法的效果好于SQUISH-E(ratio)和OPW-TR两种在线压缩算法。离线压缩算法TD-TR的效果最好，因为在对关键点信息进行评估时参考整条轨迹的信息，更容易找出最优解。

表4为算法的运行时间统计。从表4中可以看出

表2 以SED阈值为参数的实验结果

| 算法 | SED阈值/m | 压缩后点数 | 压缩率/% | 平均SED误差/m |
|---|---|---|---|---|
| TD-TR | 1 | 13 074 675 | 58.00 | 0.197 536 |
| TD-TR | 2 | 9 617 946 | 42.67 | 0.498 723 |
| TD-TR | 3 | 7 764 051 | 34.44 | 0.816 326 |
| TD-TR | 4 | 6 585 772 | 29.22 | 1.137 220 |
| TD-TR | 5 | 5 762 671 | 25.56 | 1.458 830 |
| TD-TR | 6 | 5 149 882 | 22.85 | 1.778 260 |
| TD-TR | 10 | 3 705 089 | 16.44 | 3.048 240 |
| TD-TR | 20 | 2 308 862 | 10.24 | 6.175 690 |
| TD-TR | 40 | 1 430 830 | 6.35 | 12.169 600 |
| OPW-TR | 1 | 16 449 007 | 72.97 | 0.102 950 |
| OPW-TR | 2 | 13 003 069 | 57.68 | 0.293 530 |
| OPW-TR | 3 | 10 891 596 | 48.32 | 0.510 958 |
| OPW-TR | 4 | 9 449 943 | 41.92 | 0.740 517 |
| OPW-TR | 5 | 8 398 139 | 37.26 | 0.975 607 |
| OPW-TR | 6 | 7 584 518 | 33.65 | 1.216 870 |
| OPW-TR | 10 | 5 590 324 | 24.80 | 2.199 130 |
| OPW-TR | 20 | 3 567 859 | 15.83 | 4.713 450 |
| OPW-TR | 40 | 2 204 481 | 9.78 | 9.816 990 |
| SQUISHE(sed) | 1 | 14 154 270 | 62.79 | 0.147 494 |
| SQUISHE(sed) | 2 | 11 025 951 | 48.91 | 0.344 379 |
| SQUISHE(sed) | 3 | 9 268 429 | 41.12 | 0.543 547 |
| SQUISHE(sed) | 4 | 8 105 032 | 35.95 | 0.739 955 |
| SQUISHE(sed) | 5 | 7 262 985 | 32.22 | 0.933 586 |
| SQUISHE(sed) | 6 | 6 616 748 | 29.35 | 1.125 060 |
| SQUISHE(sed) | 10 | 5 006 501 | 22.21 | 1.880 170 |
| SQUISHE(sed) | 20 | 3 320 237 | 14.73 | 3.701 210 |
| SQUISHE(sed) | 40 | 2 143 227 | 9.51 | 7.216 860 |
| KPFA | 1 | 13 665 690 | 60.62 | 0.193 422 |
| KPFA | 2 | 10 385 041 | 46.07 | 0.481 548 |
| KPFA | 3 | 8 576 141 | 38.04 | 0.783 892 |
| KPFA | 4 | 7 404 271 | 32.85 | 1.086 670 |
| KPFA | 5 | 6 568 672 | 29.14 | 1.386 900 |
| KPFA | 6 | 5 933 030 | 26.32 | 1.685 290 |
| KPFA | 10 | 4 384 452 | 19.45 | 2.860 550 |
| KPFA | 20 | 2 813 318 | 12.48 | 5.708 740 |
| KPFA | 40 | 1 762 166 | 7.82 | 11.196 500 |

表3 以压缩倍率为参数的实验结果

| 算法 | 压缩倍率 | 压缩后点数 | 压缩率/% | 平均SED误差/m |
|---|---|---|---|---|
| SQUISHE(ratio) | 2 | 11 257 777 | 49.94 | 0.568 004 |
| SQUISHE(ratio) | 3 | 7 503 179 | 33.29 | 1.423 990 |
| SQUISHE(ratio) | 4 | 5 625 460 | 24.96 | 2.451 910 |
| SQUISHE(ratio) | 10 | 2 245 583 | 9.96 | 11.771 300 |

算法本身由于参数的不同其运行时间也有一些浮动，总体来说，算法的运行时间KPFA<TD-TR<OPW-TR<SQUISH-E。其中TD-TR的运行时间在参数设为1和2时会比OPW-TR大，但是大部分情况下要小于OPW-TR。运行时间最长的SQUISH-E算法的运行时间相比其他算法要高出很多。可以看出KPFA算法的运行时间最短且稳定在10 000 ms左右。

表4 算法运行时间统计

| 算法 | SED阈值/m | 运行总时间/ms |
|---|---|---|
| TD-TR | 1 | 208 887 |
| TD-TR | 2 | 193 343 |
| TD-TR | 3 | 183 421 |
| TD-TR | 4 | 176 139 |
| TD-TR | 5 | 170 588 |
| TD-TR | 6 | 166 153 |
| TD-TR | 10 | 153 861 |
| TD-TR | 20 | 137 949 |
| TD-TR | 40 | 123 100 |
| OPW-TR | 1 | 146 596 |
| OPW-TR | 2 | 180 428 |
| OPW-TR | 3 | 216 595 |
| OPW-TR | 4 | 243 537 |
| OPW-TR | 5 | 265 793 |
| OPW-TR | 6 | 290 474 |
| OPW-TR | 10 | 372 312 |
| OPW-TR | 20 | 506 531 |
| OPW-TR | 40 | 746 184 |
| SQUISH-E(sed) | 1 | 2 159 100 |
| SQUISH-E(sed) | 2 | 2 426 764 |
| SQUISH-E(sed) | 3 | 2 575 782 |
| SQUISH-E(sed) | 4 | 2 677 757 |
| SQUISH-E(sed) | 5 | 2 743 738 |
| SQUISH-E(sed) | 6 | 2 795 634 |
| SQUISH-E(sed) | 10 | 2 929 184 |
| SQUISH-E(sed) | 20 | 3 069 068 |
| SQUISH-E(sed) | 40 | 3 158 018 |
| SQUISH-E(ratio) | 2 | 2 107 138 |
| SQUISH-E(ratio) | 3 | 2 225 498 |
| SQUISH-E(ratio) | 4 | 2 296 641 |
| SQUISH-E(ratio) | 10 | 2 278 568 |
| KPFA | 1 | 112 599 |
| KPFA | 2 | 112 073 |
| KPFA | 3 | 110 926 |
| KPFA | 4 | 110 299 |
| KPFA | 5 | 109 464 |
| KPFA | 6 | 108 677 |
| KPFA | 10 | 109 209 |
| KPFA | 20 | 106 063 |
| KPFA | 40 | 105 099 |

对于算法本身由于参数的不同造成的运行时间的浮动，则可以体现出算法的稳定性。在图4中展示了TD-TR、OPW-TR和KPFA的运行时间受参数的影响情况。其中OPW-TR算法随着同步欧氏距离阈值的增加，运行时间有明显的上升趋势。OPW算法在定位关键点时，每出现一个新的点时，就要重新计算“窗口”内所有点的同步欧氏距离，而随着同步欧氏距离阈值的增加，其“窗口”所容纳的点的数量也随之增加，重新计算的次数也随之增加，这就造成算法运行时间上升的趋势。而

TD-TR算法随着同步欧氏距离阈值的增加，运行时间有下降趋势，这是由于随着阈值的增加TD-TR算法的迭代次数下降造成的。KPFA算法的表现则非常稳定，并且运行时间要远小于另外两种算法，其根本原因在于KPFA算法在计算关键点权重值的过程中使用上次计算的结果，区别于OPW-TR算法的全部重新计算，这在很大程度上可以直接减少计算量。

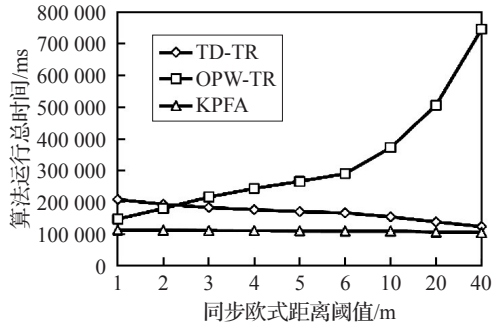


**图4　参数设置对算法运行时间的影响**

从图5中可以看出，总体上在同一压缩率水平下，平均同步欧氏距离误差TD-TR<SQUISH(sed)<KPFA<OPW-TR<SQUISH(ratio)。误差越小即对轨迹的信息量评估准确度更高，所以轨迹的信息量评估准确度为TD-TR>SQUISH(sed)>KPFA>OPW-TR>SQUISH(ratio)。可以看出在三种在线压缩算法中KPFA的评估准确度高于OPW-TR和SQUISH(ratio)两种算法。压缩率的值越大，误差相差越小，这是因为压缩率的值越大，保存下来的点越多，而保存下来的点都是信息量相对较大的点，因此平均误差就会稳定在一个趋于0的很小范围内。

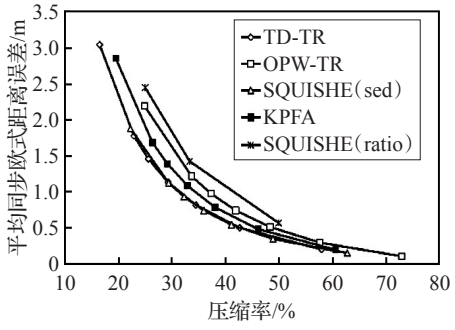


**图5　压缩率与误差的关系**

本文研究的是在线压缩算法，就不同算法而言，TD-TR和SQUISH(sed)算法的效果最好，主要原因是这两种算法是离线压缩算法，它们在对关键点信息进行评估时参考整条轨迹的信息，这样更容易找出最优解。一般离线压缩的压缩效果都是要好于在线压缩的。但运行时间太长。KPFA算法是三种在线压缩算法中表现最好的，相比于OPW-TR算法，KPFA算法的提高在于对关键点前继的修正，通过修正将漏掉的关键点重新定位，以提高算法对轨迹点信息量的评估准确度。算法缩小了在对轨迹点评估时产生的误差，进一步提高了算法的性能。

## 5　结束语

本文首先分析基于偏移量计算的在线压缩算法对轨迹进行压缩时产生误差的主要原因。由于对GPS轨迹点进行实时评估时只能通过轨迹的局部信息进行评估，所以产生误差在所难免，只能设法减少而不能完全消除。因此，本文针对在线压缩的这一特点提出基于偏移量计算的在线GPS轨迹数据压缩算法——关键点前继修正算法，通过计算同步欧式距离累积偏移量确定信息量大的关键点，然后回溯修正发现漏掉的关键点。本文的算法相对其他在线压缩算法对轨迹点的信息量评估准确度更高，运行时间更快更稳定。本文算法属于有损压缩算法对轨迹的压缩数据无法恢复。今后，研究将对此不足进一步改进。

## 参考文献：

[1] International Telecommunication Union.World telecommunication/ICT development report 2010[EB/OL].[2013-10-12]. http://www.itu.Int/ITUD/ict/publications/wtdr_10/index.html.

[2] Yan Z.Towards semantic trajectory data analysis:a conceptual and computational approach[C]//Proceedings of the VLDB 2009 PhD Workshop Co-Located with the 35th International Conference on Very Large Data Bases.New York:ACM,2009:81-83.

[3] Giannotti F,Nanni M,Pinelli F,et al.Trajectory pattern mining[C]//Proceedings of the 13th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining,2007:330-339.

[4] Chen M,Xu M,Franti P.Compression of GPS trajectories[C]// Proceedings of the 2012 Data Compression Conference(DCC). Piscataway:IEEE,2012:62-71.

[5] Chen M,Xu M,Franti P.Compression of GPS trajectories using optimized approximation[C]//Proceedings of the 2012 21st International Conference on Pattern Recognition(ICPR). Piscataway:IEEE,2012:3180-3183.

[6] Ivanov R.Real-time GPS track simplification algorithm for outdoor navigation of visually impaired[J].Journal of Network and Computer Applications,2012,35(5):1559-1567.

[7] Meratnia N,By R A D.Spatiotemporal compression techniques for moving point objects[J].Lecture Notes in Computer Science,2004:765-782.

[8] Douglas D H,Peucker T K.Algorithms for the reduction of the number of points required to represent a digitized line or its caricature[J].Cartographica:The International Journal for Geographic Information and Geovisualization, 1973,10(2):112-122.

（下转266页）

[3] 刘云,赵晓芳,陈军.HyperTransport端设备接口的设计与实现[J].计算机工程与设计,2008,29(7).
[4] Anderson D,Trodden J.Hypertransport system architecture[M].[S.l.]:Addison Wesley,2003.
[5] 阮利,秦广军,肖利民.基于龙芯多核处理器的云计算节点机[J].通信学报,2013,34(12).
[6] 孔向忠.面向SMP架构DSP的嵌入式实时操作系统研究与实现[D].西安:西安电子科技大学,2013.
[7] Reinefeld A,Doebbelin R,Schuett T.Analyzing the performance of SMP memory allocators with iterative MapReduce applications[J].Parallel Computing,2013,39(12).
[8] 李玲玲.虚拟化环境下的多核NUMA架构性能优化系统CNA[D].杭州:浙江大学,2014.
[9] 施继成,陈海波,臧斌宇.面向多处理器虚拟机的动态NUMA方法[J].小型微型计算机系统,2015(4).
[10] 夏军,徐炜遐,庞征斌.用于减少远程Cache访问延迟的最后一次写访问预测方法[J].国防科技大学学报,2015(1).
[11] 曹越,顾乃杰,任开新.一种面向多核系统的Linux任务调度算法[J].计算机工程,2015(2).
[12] 徐地,武成岗,冯晓兵.一个支持访存带宽敏感调度的跨执行优化方法[J].计算机学报,2014(7).
[13] David D,Virendra J M,Shavit N.Lock cohorting:a general technique for designing NUMA Locks[J].ACM SIGPLAN Notices,2012,47(8).
[14] 刘珂.多核处理器Cache一致性的改进[J].西安邮电大学学报,2015,20(2).
[15] 贾小敏,张民选,齐树波.片上多核Cache资源管理机制研究[J].计算机科学,2011,38(6).
[16] 潘国腾,窦强,谢伦国.基于目录的Cache一致性协议的可扩展性研究[J].计算机工程与科学,2008,30(6).
[17] 苏淑霞.基于SMP的Linux进程调度算法的研究[J].信息与电脑:理论版,2014(12).
[18] 王磊.并行计算技术综述[J].信息技术,2008(10).
[19] 迟利华,胡庆丰,刘杰.面向FT1000微处理器的STREAM并行计算与优化[J].计算机工程与科学,2014,36(12).
[20] Cheung B W L,Wang C L,Lau F C M.LOTS:a software DSM supporting large object space[C]//2004 IEEE International Conference on Cluster Computing,San Diego,California,2004.
[21] 鲍庆元.基于龙芯3号多片互联的服务器桥片实现与一致性研究[D].太原:太原理工大学,2014.

(上接253页)

[12] Chiang C Y,Wang Y R,Chen S H.Punctuation generation inspired linguistic features for Mandarin prosodic boundary prediction[C]//Proceedings of the 37th IEEE International Conference on Acoustics,Speech and Signal Processing,Kyoto,Japan,2012:4597-4600.
[13] Lafferty J,McCallum A,Pereira F.Conditional random fields:probabilistic models for segmenting and labeling sequence data[C]//Proceedings of the 18th International Conference on Machine Learning.[S.l.]:Morgan Kaufmann Press,2001:282-289.
[14] Zhao Ziping,Ma Xirong.Active learning for the prediction of prosodic phrase boundaries in Chinese speech synthesis systems using conditional random fields[C]//Proceedings of the 16th International Conference on Software Engineering,Artificial Intelligence,Networking and Parallel/Distributed Computing,Takamatsu,Japan,2015:1-5.
[15] Fernandez R,Ramabhadran B.Discriminative training and unsupervised adaptation for labeling prosodic events with limited training data[C]//INTERSPEECH 2010,Japan,2010:1429-1432.
[16] 古力米热·依玛木,艾斯卡尔·艾木都拉.维吾尔语句韵律层级的人工标注规则研究[C]//第三届全国少数民族青年自然语言信息处理、第二届全国多语言知识库建设联合学术研讨会论文集,乌鲁木齐,2010:179-182.

(上接259页)

[9] Keogh E,Chu S,Hart D,et al.An online algorithm for segmenting time series[C]//ICDM 2001:Proceedings IEEE International Conference on Data Mining.Piscataway:IEEE,2001:289-296.
[10] Muckell J,Hwang J H,Patil V,et al.SQUISH:an online approach for GPS trajectory compression[C]//Proceedings of the 2nd International Conference on Computing for Geospatial Research & Applications.New York:ACM,2011.
[11] Muckell J,Olsen Jr P W,Hwang J H,et al.Compression of trajectory data:a comprehensive evaluation and new approach[J].GeoInformatica,2014,18(3):435-460.
[12] Birnbaum J,Meng H C,Hwang J H,et al.Similarity-based compression of GPS trajectory data[C]//2013 Fourth International Conference on Computing for Geospatial Research and Application(COM.Geo),2013:92-95.
[13] El Mahrsi M K,Potier C,Hébrail G,et al.Spatiotemporal sampling for trajectory streams[C]//Proceedings of the 2010 ACM Symposium on Applied Computing,2010:1627-1628.
[14] Liu G,Iwai M,Sezaki K.An online method for trajectory simplification under uncertainty of GPS[J].Information and Media Technologies,2013,8(3):665-674.
[15] Geolife GPS trajectories data sample[EB/OL].[2012-08-09].http://research.microsoft.com//en- us/downloads/b16d359d-d164-469e-9fd4-daa38f2b2e13/.