실내 궤적 스트리밍 데이터의 실시간 스테이 포인트 탐지 및 핫스팟 시각화

라마단 하니, 유스티아완 요가, 권준호
부산대학교
hani042@pusan.ac.kr, yoga017@pusan.ac.kr, jhkwon@pusan.ac.kr

Real-Time Stay Points Detection and Hotspot Visualization of Indoor Trajectory Streaming Data

Hani Ramadhan, Yoga Yustiawan, Joonho Kwon
Pusan National University

요 약

One well-known application of big trajectory data analysis is hotspot detection, which identifies the most crowded area in a certain amount of time. Most of the hotspot detection techniques employ stay point detection to consider important spots. The popular stay point detection technique however commonly needs to analyze a full trajectory set. Thus, it is not appropriate to stream data. Because of that, we propose a real-time stay point detection approach on continuous stream data. Our experiment shows that our approach is more efficient than the non-streaming approach using simulated indoor trajectory stream. We also show detected hotspot using the grid-map based visualization.

1. Introduction

In recent years, massive application of big trajectory data has been benefited to improve people's lifestyle. However, most of the trajectory data is heavily studied on outdoor data, while research on indoor data is limited by its noisy characteristic of data collection[1][2]. Inside a building, people can still benefit the real-time indoor trajectory data analysis case, such as indoor hotspot detection [3] to whether avoid congestion in a pathway or visit the same interesting spot [4].

Presenting and detecting hotspots in a real-time manner to the users are challenging in terms of dealing with the streaming indoor position data. Several studies on hotspot detection [5][6][7][8] benefit from stay points detection algorithms [9][10][11][12][13].

However, there is minimal study on stay points detection which use the indoor streaming data [10]. Most of the research on stay point detection mainly use outdoor GPS data [9][11][12][13], while the research on indoor data exploit different source of position data, such as RFID sensor [8] and Wi-Fi signal strength [1], as capturing GPS signal inside a building may not yield accurate readings. On the other hand, stream data processing in real-time fashion needs to be computationally efficient. The research on stay point detection [9][11][12][13] rarely discuss about their performance of real-time processing using streaming trajectory data set.

Hence, we introduce a modification in the stay point detection to be applicable on real-time streaming data processing. Our approach's main difference with the previous stay point detection algorithm is that our approach does not need to wait for full trajectory to be processed and performs in O(N) complexity, while the previous approach needs to process a full trajectory and has $O(N^2)$ computation time, which is nearly not applicable to the nature of streaming data. In this preliminary study, using simulated stream data set, we compare the performance of the stay point detection based on our approach and the well-known approach. Then, we visualize the hotspot areas based on grid map using simulated indoor stream data set.

2. Proposed Method

In this section, we will show the main flow of our system, and discuss the modified stay point detection algorithm and grid-map based hotspot visualization.

Figure 1 shows the flow of our proposed method. First, user's mobile phone will produce position

estimations using indoor positioning. These estimations are streamed between a fixed time interval. Then, we will find the positions set as trajectory stored in the main memory of the mobile phone. Then, using the mobile phone, we extract the stay points from the trajectory using our modified stay point detection algorithm. After that, the mobile phone actively streams the stay points to the server as soon as a single stay point identified. After that, the server will identify which areas that considered as hotspot. Finally, the hotspots and the stay points are visualized in real-time fashion on a web-based visualization page.

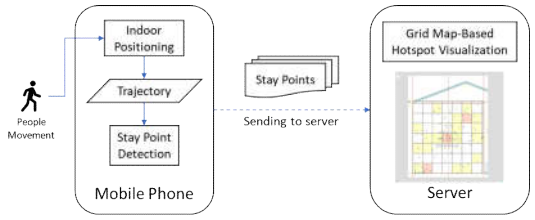

Figure 1 Flow of our trajectory stream processing

2.1. Streaming Stay Point Detection Algorithm

We provide our approach to detect stay point for stream data in Figure 2. The data here are limited to points in n-dimensional space with timestamp for each point. There are some noticeable differences from previous well-known method. First, our approach focuses on producing a single stay point (or no stay point) for a set of positions in a time interval window up to *t* time units before (for example 30 seconds before), which eliminates the time threshold for all points. Second, we try to avoid using $O(N^2)$ complexity to iterate the proximity of all points in the set, which is performed in the previous approach in stay point detection. Thus, we only compute the consecutive points' distance and make use of the centroid for the proximity of the points using distance threshold *distT*. Third, we introduce a parameter of covered ratio threshold *ratioT*. If it is set high enough, it tolerates few out-of-reach points from the centroid using the same distance threshold. Thus, a stay point can still be produced from the batch although a few points from the batch considered as noises or false positives.

2.2. Grid Map based Hotspot Visualization

The visualization begins in the server as soon as stay points are streamed from the mobile device. The visualization makes use of the indoor floorplan map that divided into blocks of area units called grid map. If the area block contains a considerable amount of stay points from different users more than other blocks in the room, it is considered as hotspot. The grid map we use in this approach is a fixed grid map as the distribution of the block areas does not follow the density of the stay points. The example of fixed grid-map based hotspot visualization is shown in Figure 3.

Algorithm 1: Stream Stay Point Detection (P, distT, t, ratioT)

Input: Streaming Position/Points Data *P*, distance threshold *distT*, time boundary *t*, ratio threshold *ratioT*
Output: Single stay point *sp*

```
1   P* ← P.subset(currentTime, currentTime − t)
    //Get points from current time until t time units before
2   i=0
3   while i < P*.length−1 do:
4     d ← dist(pi, pi+1) //distance of consecutive positions
5     if d >distT then
6       return null
7     i ← i+1
8   c ← getCentroid(P*)//set c as mean of all positions in P*
9   covered ← 0
10  for each point p in P* do:
11    if dist(c,p) <= distT then
12      covered ← covered +1
13  coveredRatio ← covered/P*.length
14  if coveredRatio < ratioT then
15    return null
16  return c as stay point sp
```

Figure 2 Algorithm for Stream Stay Point Detection

The characteristic of real-time stream data processing establishes another constraint of the hotspot detection, which is related to time. An outdated stay point should not be considered in hotspot detection as it does not represent real-time manner. Thus, we should remove stay points later than a time threshold in the visualization step.

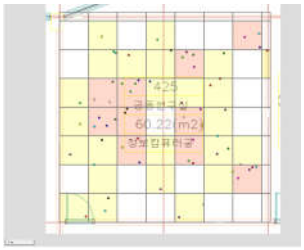

Figure 3 Example of Grid Map Hotspot Visualization

3. Experimental Result and Discussion

In this section we compare the stay point detection execution time using our approach and the previous approach, which later named non-stream approach. Then, we visualize the result of the grid map-based hotspot visualization. Both experiments use the simulated streaming data set of random movement of objects.

3.1. Stay Point Detection

We perform the stay point detection on Android smartphone with 1.6 GHz CPU speed and 3 GB RAM. In this experiment, we force the non-stream approach to work on time interval-style of trajectory set. Both the stream and the non-stream stay point detection approaches processed 32 batches of 30 seconds time interval of single object's random movement in. Surprisingly, both approaches detect 32 exact similar stay points of each batch. The comparison of their averaged execution time is provided in Figure 4.

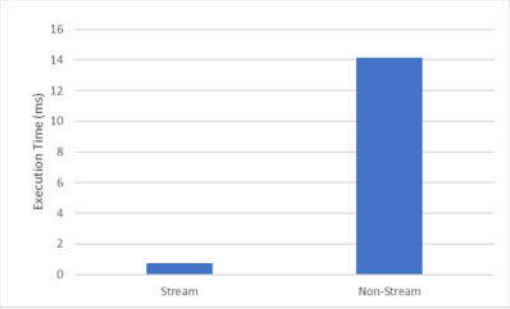

Figure 4 Comparison of stay point detection approaches

From Figure 4 we see that our stream approach executed the stay points in less than 2 ms. While the non-stream approach needs nearly 10 times longer than our approach. However, this experiment is highly dependent to the data set. Thus; in the future, we would like to examine how our approach compare to the non-stream approach on the other data set.

3.2. Hotspot Visualization

We performed experiment using stream simulated synthetic data of 10 random objects' movements inside our laboratory and the first floor of a fictitious inn. The interior obstacles are ignored; thus, the movements are only bounded by the exterior walls. The grid map is built on 1x1 m^2 area blocks division. The hotspot visualization is built under LeafletJS library [14] and customized into indoor environment.

Figure 5 Hotspot Visualizations of Our Experiment: (Left) Our Laboratory, (Right) Fictitious Inn

From Figure 5, we see that our approach can identify some hotspots that are spread in our two experiments using simulated data. Although the fixed grid map can easily visualize hotspots based on its area blocks, we see that there is a possibility to identify hotspots over the boundary of the grids. Thus, another approach using dynamic grid map to identify the areas of hotspot is also feasible. On the other hand, other approaches such as density-based clustering may be applied also as it ignores grid boundaries.

4. Conclusion

In this paper, we present a new approach on stay points detection for hotspot visualization using streaming indoor position data in real-time manner. Our approach showed promising results to detect stay points and show hotspot areas using grid map-based visualization. However, as our study is still in the preliminary research stage, improvement on hotspot visualization techniques are considered as future works. In addition, we will use real stream of position data from mobile phone using indoor positioning techniques.

Acknowledgement

This work was partly supported by the National Research Foundation of Korea (NRF) grant funded by the Korea government (*MSIT) (No. 2018R1A5A7059549) and by Capacity Enhancement Program for Scientific and Cultural Exhibition Services through the National Research Foundation of Korea (NRF) funded by Ministry of Science and ICT (NRF-2018X1A3A1069642).

References

[1] S. He and S.-H. G. Chan, "Wi-Fi Fingerprint-Based Indoor Positioning: Recent Advances and Comparisons," *IEEE Commun. Surv. Tutor.*, vol. 18, no. 1, pp. 466–490, 2016.

[2] T. Prentow, A. Thom, H. Blunck, and J. Vahrenhold, "Making Sense of Trajectory Data in Indoor Spaces," in *2015 16th IEEE International Conference on Mobile Data Management*, Pittsburgh, PA, USA, 2015, pp. 116–121.

[3] J. Jiang, X. Lv, Y. Zhang, and S. Wang, "Research on Hotspots Finding in Indoor Space Based on Regression Analysis," in *Proceedings of the 2018 International Conference on Computing and Data Engineering - ICCDE 2018*, Shanghai, China, 2018, pp. 98–102.

[4] X. Xiao, Y. Zheng, Q. Luo, and X. Xie, "Finding similar users using category-based location history," in *Proceedings of the 18th SIGSPATIAL International Conference on Advances in Geographic Information Systems - GIS '10*, San Jose, California, 2010, p. 442.

[5] Y. Zheng, "Trajectory Data Mining: An Overview," *ACM Trans. Intell. Syst. Technol.*, vol. 6, no. 3, pp. 1–41, May 2015.

[6] P. Jin, J. Du, C. Huang, S. Wan, and L. Yue, "Detecting Hotspots from Trajectory Data in Indoor Spaces," in *Database Systems for Advanced Applications*, vol. 9049, M. Renz, C. Shahabi, X. Zhou, and M. A. Cheema, Eds. Cham: Springer International Publishing, 2015, pp. 209–225.

[7] P. Zhao, K. Qin, X. Ye, Y. Wang, and Y. Chen, "A trajectory clustering approach based on decision graph and data field for detecting hotspots," *Int. J. Geogr. Inf. Sci.*, pp. 1–27, Aug. 2016.

[8] Y. Wang, B. Sommer, F. Schreiber, and H. Reiterer, "Clustering with Temporal Constraints on Spatio-Temporal Data of Human Mobility," *ArXiv180700546 Cs Stat*, Jul. 2018.

[9] B. P. L. Lau *et al.*, "Extracting Point of Interest and Classifying Environment for Low Sampling Crowd Sensing Smartphone Sensor Data," *ArXiv170103379 Cs*, Jan. 2017.

[10] G. Elmamooz, B. Finzel, and D. Nicklas, "Towards Understanding Mobility in Museums," in *Datenbanksysteme für Business, Technologie und Web (BTW 2017) - Workshopband*, Bonn, 2017, pp. 127–136.

[11] Q. Li, Y. Zheng, X. Xie, Y. Chen, W. Liu, and W.-Y. Ma, "Mining user similarity based on location history," in *Proceedings of the 16th ACM SIGSPATIAL international conference on Advances in geographic information systems - GIS '08*, Irvine, California, 2008, p. 1.

[12] J. Yuan, Y. Zheng, L. Zhang, Xi. Xie, and G. Sun, "Where to find my next passenger," in *Proceedings of the 13th international conference on Ubiquitous computing - UbiComp '11*, Beijing, China, 2011, p. 109.

[13] N. J. Yuan, Y. Zheng, L. Zhang, and X. Xie, "T-Finder: A Recommender System for Finding Passengers and Vacant Taxis," *IEEE Trans. Knowl. Data Eng.*, vol. 25, no. 10, pp. 2390–2403, Oct. 2013.

[14] V. Agafonkin, "Leaflet — an open-source JavaScript library for interactive maps." [Online]. Available: https://leafletjs.com/. [Accessed: 07-Sep-2018].