Density Peaks Clustering Approach for Discovering Demand Hot Spots in City-scale Taxi Fleet Dataset

Dongchang Liu
Institute of Automation
Chinese Academy of Sciences
Beijing, China 100190
Email: dongchang.liu@ia.ac.cn

Shih-Fen Cheng
School of Information Systems
Singapore Management University
80 Stamford Road, Singapore 178902
Email: sfcheng@smu.edu.sg

Yi-ping Yang
Institute of Automation
Chinese Academy of Sciences
Beijing, China 100190
Email: yiping.yang@ia.ac.cn

***Abstract*—In this paper, we introduce a variant of the density peaks clustering (DPC) approach for discovering demand hot spots from a low-frequency, low-quality taxi fleet operational dataset. From the literature, the DPC approach mainly uses density peaks as features to discover potential cluster centers, and this requires distances between all pairs of data points to be calculated. This implies that the DPC approach can only be applied to cases with relatively small numbers of data points. For the domain of urban taxi operations that we are interested in, we could have millions of demand points per day, and calculating all-pair distances between all demand points would be practically impossible, thus making DPC approach not applicable. To address this issue, we project all points to a density image and execute our variant of the DPC algorithm on the processed image. Experiment results show that our proposed DPC variant could get similar results as original DPC, yet with much shorter execution time and lower memory consumption. By running our DPC variant on a real-world dataset collected in Singapore, we show that there are indeed recurrent demand hot spots within the central business district that are not covered by the current taxi stand design. Our approach could be of use to both taxi fleet operator and traffic planners in guiding drivers and setting up taxi stands.**

I. Introduction

In many major cities (mostly Asian ones), taxis are considered an important and integral mode of public transport. A major challenge faced by operating taxis as a mode of public transport is how to more effectively balance supplies and demands, given that demands are inherently ad hoc and unpredictable, and taxi drivers are mostly self-interested and cannot be controlled centrally. Such structural reason is behind the inefficiency of taxi fleet operations in most cities. For example, in one study [1], it's shown that a typical taxi fleet can speed over 50% of time vacant. Such phenomenon is of great concern to city planners as vacant taxis don't just wait at fixed locations, instead, they usually drive around, burning fuels and congesting limited road space.

An important reason behind such inefficiency of taxi fleet operation is the asymmetry of demand information. In most cases, taxi drivers make their service decisions based on their limited and biased observations of past demand situations. As drivers don't have access to the global information, their decisions will inevitably be myopic and far away from being optimal. One way to mitigate such inefficiency is to make recurring demand *hot spots* a public information. For example, Figure 1 is one such illustration of demand hot spots in part of the central business district (CBD) in Singapore based on historical trip information. In Figure 1, darker areas are ones containing more trip originations, thus illustrating potential for generating demands. As we would expect, a number of these demand hot spots are taxi stands that are officially set up by traffic planners, yet there are also other demand hot spots that are formed unofficially. Recognizing these demand hot spots can help both drivers (to acquire jobs more effectively) and planners (to set up new taxi stands to cater to recurrent demands at fixed locations).

The major algorithmic framework we adopt for discovering demand hot spots is called the *density peaks clustering* (DPC) approach. The DPC is superior than classical clustering approaches for our application domain because it considers not just density of data points, but also distances between different density peaks. This allows clusters to be discovered even with noisy background. Unfortunately, to properly estimate densities within the DPC framework, distances between all pairs of data points need to be calculated a priori. As a result, DPC is not scalable even to a moderate-size data set.

The most important contribution of our paper is the development of a scalable variant of DPC that works with millions of data points, which is common for urban taxi fleet operations. At very high level, the major design idea we adopt is to first project all data points to a much smaller *density image*, which is then processed by the modified DPC algorithm. From our experiment results, we show that both execution time and memory requirement can be greatly reduced, yet the performance of our approach (approximating density estimation using the density image) is comparable to the original DPC (can be viewed as an exact approach). When applied to the real-world data set we obtained from a Singapore taxi fleet, we demonstrate that demand hot spots can be reliably found even when the data is of low-frequency and low-quality.

This paper is organized as follows. We introduce related works in Section II. In Section III, we formally introduce basic steps of our method to detect and locate hot spots without priors. In Section IV, we evaluate the efficiencies of different clustering techniques for detecting demand hot spots. Finally, we discuss the experiment results and how we can improve the performance of these methods.

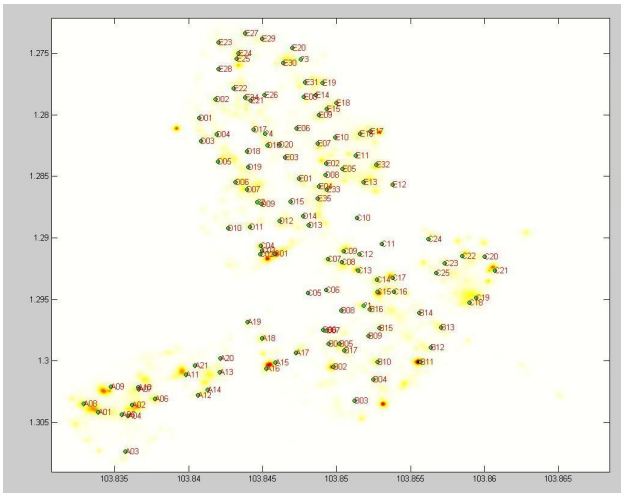

Fig. 1. An example of demand hot spots, identified by plotting the density of trip originations.

II. Related Work

For the past two decades, we have seen great progress in the development of tracking technologies. In particular, with the proliferation of sensor-rich smartphones, it's becoming easier and cheaper to track individuals or vehicles. The emergence of these data sets that possess both spatial and temporal information have posted great challenges for researchers, and spurred great interests in developing new technologies for extracting insights and making real-time decisions/recommendations.

From the literature, we can see that there are at least three major streams of ongoing research: (1) mapping technology: for example, enriching existing digital maps from collected GPS traces [2], (2) pattern discovery: detecting patters such as flocking, encountering, and leadership [3], and (3) recommender development: such as providing driving directions for taxi drivers [4]. Our research falls under the second category; in particular, we focus on identifying hot spots of trip originations in a taxi fleet.

Finding hot spots can be achieved mostly by variants of clustering algorithms, for example, CLARANS [5], CURE [6], and GDBSCAN [7], just to name a few. In recent years, a family of approaches based on kernel density estimation (KDE) has been steadily gaining popularity due to its ease of usage and visually appealing outputs [8]. The best known algorithm in this family is the mean shift algorithm, which is first proposed in [9]. Mean shift is a density-based, non-parametric clustering method and it has the following advantages: 1) data driven, 2) self-proved, 3) requires only one parameter: "window size (bandwidth)". However, choosing window size is usually difficult, and inappropriate choice could lead to undesirable merging of modes or "shadow" modes being generated.

The difficulty with parameter setting in not just mean shift algorithm but also classical clustering algorithms has been recently addressed by Rodriguez and Laio [10]. The proposed "density peaks clustering" (DPC) algorithm searches for clusters using two dimensions instead of one by including distances among density peaks. The intuition behind the DPC algorithm is that cluster centers are usually characterized by a higher density when compared to neighboring region (ideas used in many existing clustering approaches), yet for a clustering center to be truly unique, it should also be far enough from other high-density points (new ideas incorporated by the DPC algorithm). This new design can generate a reasonable number of clusters without intervention (a big issue for classical approaches such as k-mean algorithm), and is relatively robust against outliers and also the geometry of the cluster (the way points cluster together). However, the DPC algorithm is extremely computationally intensive as distances between all pairs of points need to be computed. A major idea experimented in this paper is to perform clustering not on the raw data but on the derived density images. If a good density image that is representative of the raw data can be produced, the resulting clusters should be a good approximation of the ones generated using raw data. The modified DPC algorithm, which operates on density images instead of raw data points, allow the speed to be greatly improved and memory requirement to be greatly reduced.

There are many ways in which density images can be generated. In our research, we draw our design idea from the field of image processing, in particular, the well-adopted Otsu binarization [11] which can turn a gray-scale image into a binary image (only black and white). Otsu's algorithm is just one of many candidate approaches, another notable algorithm is based on mathematical morphological operations [12].

III. The Improved DPC Algorithm

As noted earlier, the original DPC algorithm is not scalable due to its requirement of having to compute all-pair distances. An important technique we introduce to make DPC algorithm scalable is the use of a density image projection. In this section, we will first introduce the original DPC algorithm and then extend the discussion to our DPC variant.

A. Density Peaks Clustering

The original DPC algorithm is described in Algorithm 1. Compared to the classical clustering approaches, DPC algorithm is special in how it chooses cluster centers. Firstly, the DPC algorithm find cluster centers by using *densities*, which is inherently a two-dimensional measure, and is better than just distance. Secondly, besides densities, DPC algorithm also calculates *relative densities*, which depends on the distance between density peaks. Because of these two innovations, the DPC algorithm is able to effectively identify cluster centers even when the dataset is noisy.

However, the strength of DPC algorithm is also its weakness. To properly calculate densities at all points, distances between all pairs of data points are needed. Given N points, this implies that a distance matrix of size N by N will be needed. As the matrix is dense, this preprocessing step will consume huge amount of memory space and also execution

time. Besides this issue, we also find DPC algorithm inadequate, as we need to identify not just cluster centers, but also the contours of clusters. For these two reasons, we propose to create a variant of the DPC algorithm.

Algorithm 1 Density Peaks Clustering

1: Initialize input $P = \{p_1, p_2, \ldots, p_n\}$, output $C = \bigotimes$.
2: Parameters: M: number of cluster centers, η: ratio for calculating neighborhood distance d_{nb} (default = 2%).
3: Calculate distance matrix between all pairs of points.
4: Define neighborhood distance d_{nb} to be the K^{th} smallest distances, $K = \eta \cdot n(n-1)/2$.
5: Estimate local density of each point (ρ_i) with (1).
6: Compute distance between density peaks (δ_i) with (2). For point with largest density, we assign its δ with the largest distance.
7: Select cluster centers based on $\gamma = \rho \cdot \delta$, points with top M largest γ will be selected as cluster centers.
8: Assign points to different clusters based on ρ in descending order.
9: Remove halo points (points in low density areas) if needed.
10: Output cluster result C.

$$\rho_i = \sum_j \exp\{-1.0 \cdot (d_{ij}/d_{nb})\} \tag{1}$$

$$\delta_i = \min_{j:\rho_j > \rho_i} (d_{ij}) \tag{2}$$

B. Improved Density Peaks Clustering Algorithm

An important innovation introduced to improve the performance of DPC algorithm and expand its capability is through the use of a *density image*. Instead of processing all data points directly, we will first generate a density image from the data points, and then modify the DPC algorithm to read this image as input instead. Details about the improved DPC algorithm are described in Algorithm 2. The density image from point projecting improved the DPC algorithm from the following aspects:

1) As the density image is created via straightforward projection, it requires little memory, and is extremely fast.
2) With a density image, we can easily select a representative point from each lattice of the image. This allows us to eliminate all the redundant points (either points that overlap, or are very close to each other), and thus speed up both the calculation of all-pair distance and estimation of densities.
3) Since densities of the dataset at all locations are now represented as an image, we can easily apply image filters to remove background noises.

Algorithm 2 Improved Density Peaks Clustering

1: Initialize input $P = \{p_1, p_2, \ldots, p_n\}$, output $C = \bigotimes$.
2: Parameter: M: number of cluster centers, H: a fixed window size.
3: Project all GPS points by their locations with H.
4: Remove isolated pixels and smooth the density image I.
5: Binarize I with Otsu's threshold and identify points in high density areas or white area on binary image B.
6: Select deputy points $P' = \{p'_1, p'_2, \ldots, p'_m\}$ from each valid white pixel on B, one pixel per deputy point.
7: Get density of each point (ρ_i) from the pixel value of I.
8: Compute distance between density peaks (δ_i) with (2). For point with largest density, we assign its δ with the largest distance.
9: Select cluster centers based on $\gamma = \rho \cdot \delta$, points with top M largest γ will be selected as cluster centers.
10: Assign points to different clusters based on ρ in descending order.
11: Output cluster result C.

IV. Computational Experiments

A. The Taxi Fleet Dataset

The evaluation of our improved DPC algorithm is based on a real-world dataset from a taxi fleet in Singapore. As we are concerned about demand hot spots, we will first extract origins of all completed taxi trips. Available fields and sample inputs from the trip records are summarized in Table I. For the purpose of our study, we choose only trips originating from a small area within the central business district of Singapore in September, 2009. After this preprocessing, we are left with 168,668 trip records. All these trips are shown as a scatter plot in Figure 2.

TABLE I
Available fields from the taxi trip dataset.

Name	Definition	Example
VID	Identity of vehicle	8
TID	Identity of trip	2164E1003
STime	Start time of the trip	1267374600
SLongitude	Longitude of trip origin	103.90583
SLatitude	Latitude of trip origin	1.39938
ETime	End time of a trip	1267374960
ELogitude	Longitude of trip destination	103.92881
ELatitude	Latitude of trip destination	1.38951
Distance	Total traveled distance	6.3
Fare	Fare incurred	870

B. Parameter Selection

Both Algorithms 1 and 2 need to have their respective algorithm-level parameters. For Algorithm 1, we need to specify the number of cluster centers, M, and the ratio for calculating neighborhood distance, K. For Algorithm 2, we need to specify the number of cluster centers, M, and the projection window size, H. Empirically speaking, setting M for most clustering algorithms will require some trial runs. However, as our improved DPC algorithm depends on the density image,

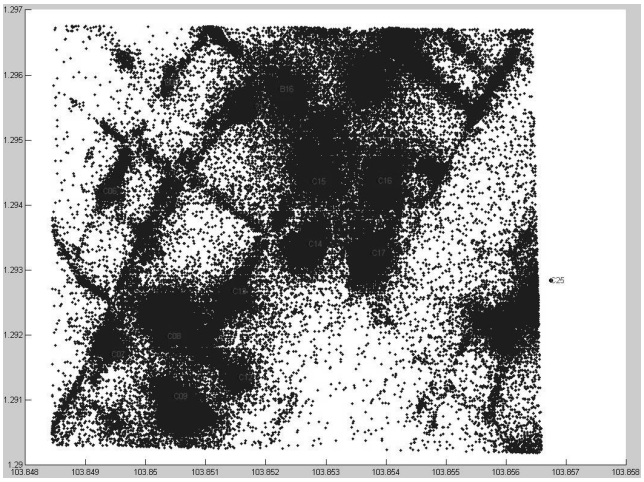

Fig. 2. Scatter plot of all trip origins in the area of study.

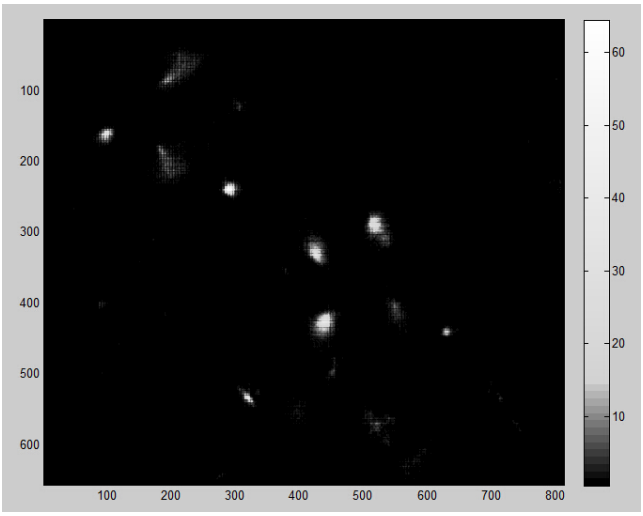

Fig. 3. The density image based on projection.

we can count the blobs in the binary image, and use it to estimate M. For K, we follow the literature on DPC algorithm, and use the recommended formula: $K = 2\% \cdot n \cdot (n-1)/2$.

The projection window size parameter, H, is the critical parameter that will be required in our improved DPC algorithm. The purpose of H is for the construction of lattices when we project geographical coordinates to the density image. Because of this, H should be chosen to match the accuracy of the device that generates GPS coordinates. For the particular taxi fleet that we study, the upper bound of the resolution is around 0.0001 degree (10 meters in real distance); the lower bound of the resolution, on the other hand, is set to be around 1% of the upper bound. By testing the H parameter on simulated data, we find out that the best value for H will be around 10^{-5}, which is used for the rest of our experiments.

C. The Improved DPC Algorithm

The first step in the improved DPC algorithm is to project all geographical coordinates onto a density graph. To achieve this, we will need to first use the parameter H in setting up the image lattices. Each lattice is essentially an H by H square on the original graphical plane. The density image is essentially a collection of pixels, where each pixel stores the proxy density value of the corresponding square lattice. To correct for the potential errors and outliers, we remove pixels whose values are 5 times more than the sum of their 5×5 neighbors. Finally, to produce the final image, we smoothen the image by applying a 3×3 Gaussian kernel to it. The density image in greyscale is plotted in Figure 3. The binary density image (containing only black and white) can finally be obtained by applying the well-known parameter-free Otsu's method. The outcome can be found in Figure 4.

This binary density image can be further processed to obtain the mask image for filtering original data points. The processing steps are simple: just remove all components that are smaller than 200 pixels and fill white space that are too small. The resulting mask image is shown in Figure 5.

After these above steps, the number of data points is reduced from 168,668 to 27,656 points. In the mean time, we also get all density values of these remained points. We can use pixel values corresponding to their locations as estimated density. These 27,656 points and their densities (defined as P' in Algorithm 2) are the actual inputs for calculating density peaks. After this step, we could obtain a 2-dimension array that stores both the density and peak distance of every point. Finally, to generate cluster center, we use the product of density and peak distance $\gamma = \rho \cdot \delta$ as the criterion. The collection of M points with largest γ are selected as cluster centers. With these selected cluster centers, we can then label all remaining points that pass the binary mask (from Figure 5). The color-coded clustering result is shown in Figure 6.

In the final step of our algorithm, we would identify contours around all points belonging to the same clusters. Figure 7 illustrates these cluster centers and contours. These contours are more meaningful for driver guidance as we can easily detect whether drivers are within identified high-demand zones. The alpha-numerical labels beside some of the cluster centers are the official taxi stands in this part of the central business district.

D. Comparison Against the Original DPC Algorithm

To demonstrate the efficiency and effectiveness of our improved DPC algorithm, we apply the original DPC algorithm to analyze the raw dataset with 168,668 data points. In terms of execution time, the original DPC algorithm uses 216.7 minutes, while our improved DPC algorithm uses only 1.3 minutes. In terms of memory usage, the original DPC algorithm consumes up to 238GB of memory while the improved DPC algorithm uses only 7.7MB of memory. The clustering results of the original DPC algorithm can be found in Figure 8 and Figure 9. A major difference we can spot visually is smoother cluster contours and better clustering of all raw data points. However, given the amount of computational resources that would be required to obtain this marginal advantage, we

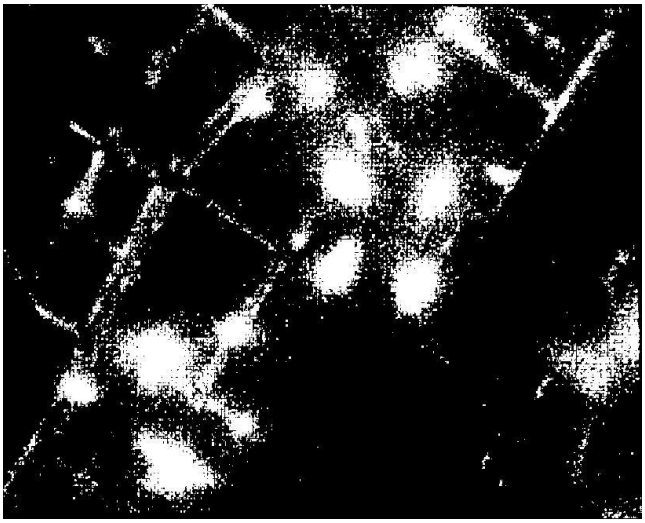

Fig. 4. Binary image obtained using Otsu's method.

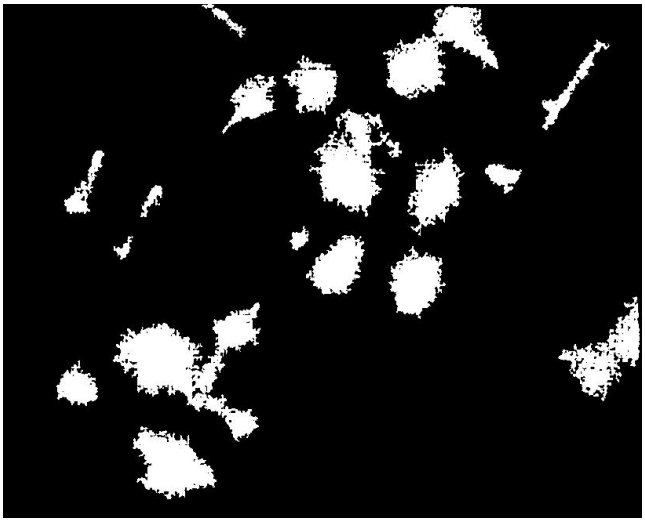

Fig. 5. Corresponding binary mask for data filtering.

would consider this trade-off worthy of the sacrifice we suffer in the solution quality.

To verify our computational results, we use a list of official taxi stands from the Land Transport Authority of Singapore[1]. The list contains locations and identifications of all taxi stands in the CBD of Singapore. For the illustration purpose, we focus on a small area which contains 13 official taxi stands. The results are summarized in Table II: of 13 taxi stands, 11 can be easily detected, while the remaining 2 are missing (B08 and B14) due to very low trip origination counts. Interestingly, there are a number of identified hot spots that are not in the official taxi stand list. Some of them are near major hotels, some are near major shopping malls, and some are parking lines along main roads. It's very interesting to note that although there are two centers that are identified due

[1]http://www.lta.gov.sg/

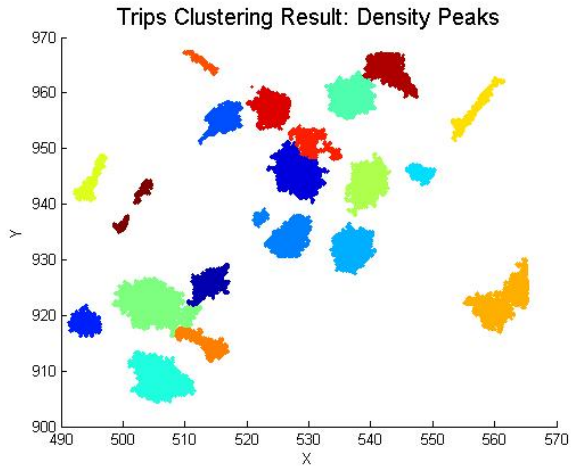

Fig. 6. The obtained cluster centers and all points belonging to these clusters using binary mask from Figure 5.

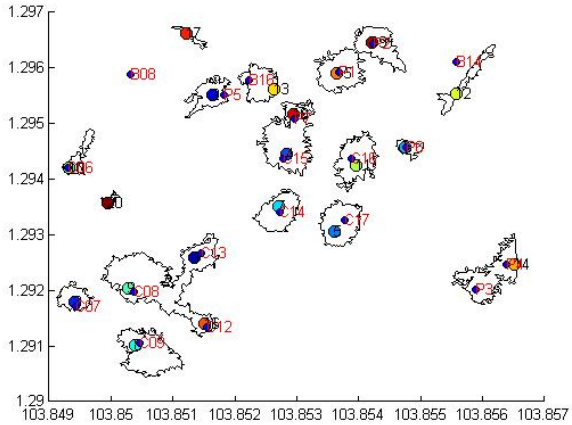

Fig. 7. Cluster centers and contours obtained from Figure 6.

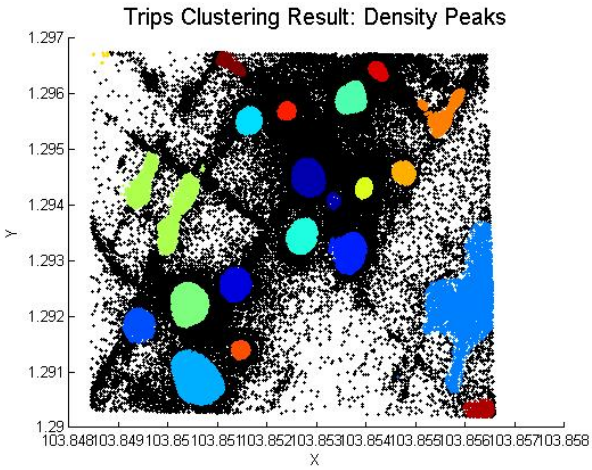

Fig. 8. Clustering results from the original DPC algorithm.

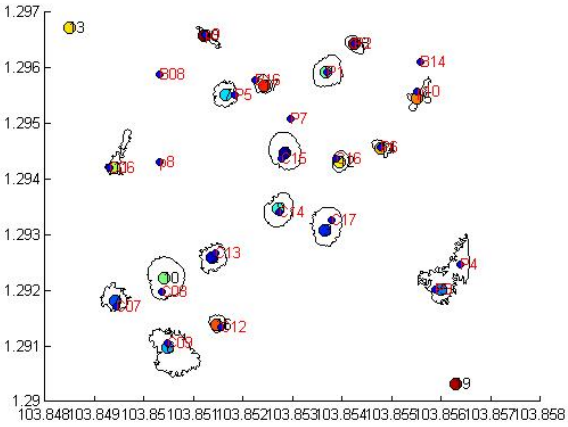

Fig. 9. Clustering contours and centers from the original DPC algorithm.

TABLE II
SUMMARY OF CLUSTERING RESULTS.

Ground Truth	Improved DPC	Original DPC
C17 taxi stand	05	03
C16 taxi stand	11	12
C15 taxi stand	04	01
C14 taxi stand	07	08
C13 taxi stand	01	02
C12 taxi stand	16	16
C09 taxi stand	08	06
C08 taxi stand	09	10
C07 taxi stand	03	04
C06 taxi stand	10	11
B16 taxi stand	13	17
B08 taxi stand	missing	missing
B14 taxi stand	missing	missing
P1 parking lines	15	09
P2 parking lines	19	18
P3 parking lines	missing	05
P4 parking lines	14	missing
P5 near hotels	02	07
P6 near hotels	06	14
P7 parking lines	18	missing
P8 parking lines	20	missing
P9 parking lines	17	20
P10 parking lines	12	15
Noises	missing	13
Noises	missing	19

to signal noises in the original DPC, it disappeared after the generation of density image. It shows that by approximating density clustering using density images, we could eliminate some of the noises along the process.

V. CONCLUSIONS

From experiments in last section, we show that 1) by combining DPC algorithm with density image analysis, we could improve the original DPC algorithm in both time and memory consumption; 2) the improved DPC algorithm is not only much more efficient, but it also identifies contours besides just cluster centers; and 3) compared with ground truth information, the improved DPC algorithm indeed performs better than the traditional DPC algorithm.

In particular, total execution time required for our numerical instances is reduced from hundreds of minutes to only slightly over one minute. Memory consumption is also cut down from 238GB to 7.75MB. Therefore, the method we proposed has advantages in dealing with traces coming from a large-scale data set. Finally, we show that our methods can obtain cluster contours besides just cluster centers. These contours are significant in the real-time use of demand hot spots, cause they would provide taxi drivers a global reference of taxi demands and tell them in real time whether the taxi is in a demand hot spot area or not.

The proposed method can also be used to generate long-term demand hot spots with yearly collected taxi operational data in the whole CBD of Singapore. The data scale may be tens of millions. While the original DPC may fail because of memory and time requirement, our DPC variant will still scale well. Long-term demand hot spots are useful for traffic planner to reconfigured taxi stands and parking lines, or even provide guidance to drivers.

ACKNOWLEDGMENT

This research was supported in part by the National Research Foundation Singapore through the Singapore-MIT Alliance for Research and Technology's Future Urban Mobility IRG research programme and the Corp Lab@University scheme.

REFERENCES

[1] S.-F. Cheng and X. Qu, "A service choice model for optimizing taxi service delivery," in *12th International IEEE Conference on Intelligent Transportation Systems*, 2009, pp. 66–71.

[2] S. Schroedl, K. Wagstaff, S. Rogers, P. Langley, and C. Wilson, "Mining GPS traces for map refinement," *Data Mining and Knowledge Discovery*, vol. 9, no. 1, pp. 59–87, 2004.

[3] J. Gudmundsson, M. van Kreveld, and B. Speckmann, "Efficient detection of motion patterns in spatio-temporal data sets," in *12th Annual ACM International Workshop on Geographic Information Systems*. ACM, 2004, pp. 250–257.

[4] J. Yuan, Y. Zheng, X. Xie, and G. Sun, "T-drive: enhancing driving directions with taxi drivers' intelligence," *IEEE Transactions on Knowledge and Data Engineering*, vol. 25, no. 1, pp. 220–232, 2013.

[5] R. T. Ng and J. Han, "CLARANS: A method for clustering objects for spatial data mining," *IEEE Transactions on Knowledge and Data Engineering*, vol. 14, no. 5, pp. 1003–1016, 2002.

[6] S. Guha, R. Rastogi, and K. Shim, "CURE: An efficient clustering algorithm for large databases," in *ACM SIGMOD Record*, vol. 27, 1998, pp. 73–84.

[7] J. Sander, M. Ester, H.-P. Kriegel, and X. Xu, "Density-based clustering in spatial databases: The algorithm GDBSCAN and its applications," *Data Mining and Knowledge Discovery*, vol. 2, no. 2, pp. 169–194, 1998.

[8] A. J. Brimicombe, "Cluster detection in point event data having tendency towards spatially repetitive events," in *8th International Conference on GeoComputation*, 2005.

[9] Y. Cheng, "Mean shift, mode seeking, and clustering," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 17, no. 8, pp. 790–799, 1995.

[10] A. Rodriguez and A. Laio, "Clustering by fast search and find of density peaks," *Science*, vol. 344, no. 6191, pp. 1492–1496, 2014.

[11] N. Otsu, "A threshold selection method from gray-level histograms," *Automatica*, vol. 11, no. 285-296, pp. 23–27, 1975.

[12] R. M. Haralick, S. R. Sternberg, and X. Zhuang, "Image analysis using mathematical morphology," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. PAMI-9, no. 4, pp. 532–550, 1987.