Front. Comput. Sci., 2017, 11(1): 49–60
DOI 10.1007/s11704-016-5553-z

RESEARCH ARTICLE

Real-time and generic queue time estimation based on mobile crowdsensing

Jiangtao WANG[1,2,3], Yasha WANG (✉)[1,3,4], Daqing ZHANG[1,2,3], Leye WANG[5], Chao CHEN[6], Jae Woong LEE[7], Yuanduo HE[1,2,3]

1 Key Laboratory of High Confidence Software Technologies, Ministry of Education, Beijing 100871, China
2 School of Electronics Engineering and Computer Science, Peking University, Beijing 100871, China
3 Beida (Binhai) Information Research, Tianjin 300450, China
4 National Engineering Research Center of Software Engineering, Peking University, Beijing 100871, China
5 Network & Services Department, Institut Mines-Télécom/Télécom SudParis, Evry 91011, France
6 Department of Computer Science, Chongqing University, Chongqing 400044, China
7 Department of Mathematics and Computer Science, University of Central Missouri, Warrensburg MO 64093, USA

Abstract People often have to queue for a busy service in many places around a city, and knowing the queue time can be helpful for making better activity plans to avoid long queues. Traditional solutions to the queue time monitoring are based on pre-deployed infrastructures, such as cameras and infrared sensors, which are costly and fail to deliver the queue time information to scattered citizens. This paper presents CrowdQTE, a mobile crowdsensing system, which utilizes the sensor-enhanced mobile devices and crowd human intelligence to monitor and provide real-time queue time information for various queuing scenarios. When people are waiting in a line, we utilize the accelerometer sensor data and ambient contexts to automatically detect the queueing behavior and calculate the queue time. When people are not waiting in a line, it estimates the queue time based on the information reported manually by participants. We evaluate the performance of the system with a two-week and 12-person deployment using commercially-available smartphones. The results demonstrate that CrowdQTE is effective in estimating queuing status.

Keywords mobile crowdsensing, queue time estimation, opportunistic and participatory sensing

Received December 21, 2015; accepted August 9, 2016

E-mail: jiangtaowang@pku.edu.cn; yasha.wang@163.com

1 Introduction

People often wait in long lines in many places (e.g., supermarket, theme parks, transportation stations, and banks), and long queue time brings awful user experience. Knowing the real-time queue time can help a lot in our daily life. First, it can help us decide whether it is appropriate to visit some places right now or a certain time later. Second, it can also help us make better plans when we have to wait in a long queue. For example, if the current queue time of the railway ticket office is likely to make us miss the train, we will pick up speed. Third, managing queues can help service providers to allocate resources more efficiently (e.g., arranging the number of cashiers in the supermarket according to the queue time) and improve service quality. In summary, there is a need for a better understanding of the real-time queue time.

Most existing solutions to the queue monitoring problem rely on cameras [1–5] or special devices (e.g., infrared sensor, Wi-Fi access point or floor mats) deployed at specified locations [6–8]. These approaches have the following limitations. First, it incurs extra deployment cost. As a result,

service providers may not be willing to build such a system. Second, the usage scale of such system is limited, since it only works in the places where specific devices are deployed. Third, approaches based on cameras also lead to privacy issues. Fourth, these monitoring systems are developed for a specific place, so that it is hard to deliver the queue time information to the citizens distributed in different locations of the city.

With the pervasiveness of sensor-rich (e.g., accelerometer, GPS, and camera) mobile phones, mobile crowdsensing (MCS) [9,10] has become an emerging paradigm for large-scale, real-world sensing and information gathering, and it has proved its usefulness in a variety of application areas [11–16]. There are primarily two mobile crowdsensing paradigms [17]: 1) participatory sensing and 2) opportunistic sensing. In participatory sensing, users actively engage in sensing activities by manually determining when, where, what, and how to sense. In opportunistic sensing, sensing activities are fully automated without users' involvement.

This paper proposes a crowdsensing-based system, named CrowdQTE, which monitors and provides real-time queue time information for multiple Point-of-Interests (POIs). On the client side, it collects the queue time information through participants' mobile phones. On the server side, the system aggregates the collected information from multiple participants to estimate the current queue time.

The basic intuition comes from the observation that the movement of people waiting in a line follows a similar pattern, i.e., standing for a while, moving one or several steps forward along with the queue and then standing again. This pattern is referred to as *standing-walking-standing* mode in this paper. According to the pattern, there is a chance to recognize user's queuing behavior by leveraging the accelerometer sensor embedded in the phone.

Although the above idea seems straightforward, many challenges arise in practice, because people's behavior is diverse in different queuing scenarios. First, people's behavior varies in the different places. For example, when people wait in certain places such as bank or hospital, they do not stand in a line and follow the standing-walking-standing pattern. Instead, they usually get a ticket from the queue management system first, and then wait until their turns. During the waiting time, they may sit, stand, stroll, or even go outside to do something else. Thus, it is necessary to differentiate whether a user's queuing behavior is standing-walking-standing or ticket-holding based on the place she is located. Second, even in the places where people wait in a line, it is still challenging to detect queuing mode due to similar patterns of other behaviors in the same POI. For example, a customer's movement of scanning and selecting goods in a supermarket can be identified incorrectly as in the queuing mode. If the goods that a customer is interested in are placed nearby to each other in the shelves, the customer may walk to a product and then stands for a little while (browsing the commodity information). Afterwards, he/she steps to the next product and stands again. In this situation, the customer's movement also follows the standing-walking-standing mode. Thus, distinguishing queuing mode from similar behaviors becomes challenging.

To tackle the above challenges, we first divide the queuing scenarios into two categories. One is referred to as the "*waiting-in-line*" scenario, where people wait in a line and follow the standing-walking-standing pattern. The other is referred to as "*ticket-holding*" scenario, where people obtain a ticket from the queue management system when they enter the POI and then wait until the service can be provided. Then, this paper leverages the opportunistic and participatory sensing to monitor the above two different scenarios, respectively. 1) In the waiting-in-line scenarios, we utilize the accelerometer sensor data to automatically detect the queueing behavior and calculate the queue time, which follows the opportunistic sensing paradigm [18]. In addition, we exploit other ambient contexts, such as the ambient sounds, to distinguish the queuing from other similar behaviors. 2) In the ticket-holding scenarios, the participants manually report the queuing status, which is referred to participatory sensing [19]. In particular, we propose a location-based planner based on a knowledge base to switch between the above two modes.

Specifically, we make the following contributions in this work:

- We develop a mobile platform for estimating queue time of multiple POIs by leveraging MCS in both the opportunistic and participatory sensing modes.
- We propose an automatic queuing mode identification algorithm based on accelerometer data. Besides, other ambient contexts are exploited collaboratively to improve the recognition accuracy.
- We evaluate the performance of the system with a two-week and 12-person deployment using commercially-available smartphones. The results demonstrate that CrowdQTE is effective in estimating queue time.

The rest of this paper is organized as follows. Section 2 summarizes the related work. In Section 3, we describe the design of CrowdQTE, and Section 4 proposes the algorithms queue monitoring. Section 5 is about the data aggregation and

queue time estimation. Section 6 describes the experimental method and results. Finally, Section 7 concludes this paper with discussions about future work.

2 Related work

2.1 Mobile crowdsensing

Mobile crowdsensing (MCS) is a new sensing paradigm that empowers ordinary people to contribute data gathered or generated from their mobile devices. It further aggregates heterogeneous crowdsourced data in the cloud to extract certain knowledge [20,21]. There have been numerous MCS-powered applications, such as public information reposting [13], environment monitoring [14,15], traffic planning [16], social context sensing [12,22], public safety [23], and social event replay [24]. In this paper we focus on a novel area of MCS applications, which aims to leverage crowd power to detect the queue time in multiple Point-of-Interests.

There are primarily two mobile crowd sensing paradigms [17]: 1) participatory sensing and 2) opportunistic sensing. In participatory sensing, users actively engage in sensing activities (e.g., take a picture and record a video) by manually determining when, where, what, and how to sense. In opportunistic sensing, sensing activities are fully automated without user involvement (e.g., scan Wi-Fi signals and record noise samples).

2.2 Traditional queue monitoring

Traditional solutions to the line wait-time monitoring problem are based on pre-deployed infrastructures such as camera placement [1–5], sensor deployment [6] or monitoring signals generated through Bluetooth [7] or Wi-Fi capable devices [8]. However, these solutions are usually unscalable, costly to deploy and mostly designed for specific places. In addition, camera-based solutions also lead to privacy concerns.

2.3 MCS-based queue monitoring

To the best of our knowledge, there are only a few research works [25,26] that detect queue time by using crowdsourced mobile phone data. Bulut et al. [25] presented LineKing, a crowdsourced line wait-time monitoring service. The approach utilizes the Wi-Fi or location proximity to estimate queue lines in coffee shops. The key idea of LineKing is that it detects the entrance and exit in high precision by exploiting the unique fingerprint of WAP beacons in the coffee shop. Li et al. [26] proposed a collaborative approach to distinguish queuing mode from non-queuing mode by comparing the sensor data similarities among people in a certain place.

Both Refs. [25] and [26] have strong assumptions in order to obtain a high accuracy. The work [25] assumes that a user enters the shop, buys a coffee and leaves immediately. Other activities a user may perform in the shop are not taken into considerations. However, if a participant drinks coffee inside the shop for killing times, the estimated waiting time will be inaccurate. Similarly, if a participant spends a lot of time on selecting goods in a supermarket and only waits for a short period of time in a line, the system will still estimate a long waiting time, which is obviously inaccurate. In contrast, our system does not rely on this assumption, and thus can be used in multiple POIs. In Ref. [26], the author assumes that there is an adequate number of users in a line using the proposed mobile crowdsourcing application, so that the collaborative learning algorithm can detect the queuing by analyzing the similarity among the participant's behavior. However, in realistic scenarios, the collaborative approach does not work well when only a few participants installed the waiting time detection application in a certain place. In contrast, our system performs well even if there are only a few participants who have installed our application.

3 System design overview

Figure 1 gives a brief overview of CrowdQTE system, which is divided into phone side (or client side) and server side.

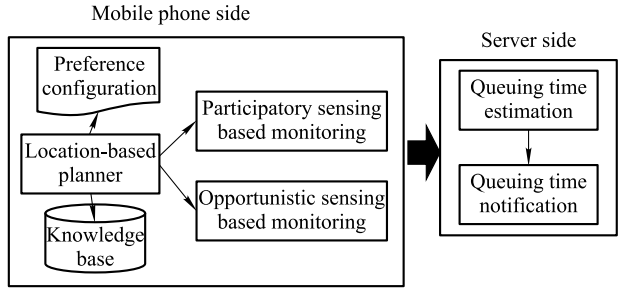


Fig. 1 CrowdQTE: system overview

On the phone side, CrowdQTE collects the queue time information through user's mobile phone. 1) In the waiting-in-line scenarios, CrowdQTE utilizes the accelerometer sensor data to automatically detect the queueing behavior and calculate the queue time, which follows the opportunistic sensing paradigm. 2) In the ticket-holding scenarios, the participants manually report the queuing status, which is referred to as participatory sensing. In particular, a location-based planner

is designed based on a knowledge base to switch between the above two modes.

On the server side, the system first aggregates the collected information from multiple participants to estimate the current target queue time. Then, based on the estimated queue time, CrowdQTE provides the service for end-users to query information through mobile clients or web browsers.

3.1 Queue monitoring

Since the CrowdQTE adopts two different sensing modes in different scenarios, it is important for the system to determine which mode should be applied. A knowledge base is established offline, which stores corresponding attributes for different POIs. For a specific POI, its attributes include its location fingerprint (cellular tower and Wi-Fi), detection mode (opportunistic/participatory) and required contexts. The knowledge base is expandable so that new POIs can be added easily. Besides, participants' preferences are pre-defined through the client side, including privacy preserving, energy consuming management, data traffic management policy, usage habits and incentives. Typical options include when and where to collect data, under what energy condition for a phone to work, in which network connection (e.g., Wi-Fi, 3G/4G, GPRS) to upload results of measurement, etc. Both the knowledge base and preference configuration is lightweight, so that it can be downloaded in the mobile phone side when participants are recruited.

At the runtime, the location-based planner determines the participant's location in an energy-efficient way. First, the system collects and stores the cell ID of each cellular tower in which the POI is located, and the geographical position of each cell tower. The phone-side application gets the current cell ID and calculates the distance between the current location and the target POIs. When the current location is far away (e.g., more than 5 KM) from any of the target POIs, the system conducts the above operation every 15 minutes. Otherwise, the cell ID based localization will be operated every 5 minutes. Second, when the system detects that the participant is near a POI (i.e., within a corresponding cellular tower's scale), the localization switches to the Wi-Fi based approach. The system will calculate the similarity between the Wi-Fi fingerprint between current location and the target POI in this cell. If the similarity is less than a threshold, then we consider the participant is inside a specific POI.

Once a participant is detected inside a POI, the planner will determine the sensing mode (participatory/opportunistic) based on the information in the knowledge base, and then invoke the corresponding procedure: (1) In the ticket-holding scenarios, the planner invokes the participatory sensing based queue monitoring, which invites participants to manually report the queuing conditions. The details are described in Section 4.1. (2) In the waiting-in-line scenarios, the planner invokes the opportunistic sensing based queue monitoring, which utilizes the accelerometer sensor data to automatically detect the queueing behaviors, calculate and report the queue time. Besides, the sensing module exploits other ambient contexts, such as the ambient sounds, to distinguish the queuing mode from other similar behaviors (e.g., scanning and selecting goods). The automatic queuing mode identification algorithm will be described in detail in Section 4.2.

3.2 Data aggregation, incentive, and service provision

In the server side, CrowdQTE first divides data from different phones in a given POI into data groups according to the sensing time, then eliminates noisy data from each group, and at last calculates the average of valid queue time/condition as the estimation. The aggregation and estimation algorithm will be introduced in details in Section 5.

To compensate the consumption of device-oriented resources (e.g., battery, network cost) and human attention (e.g. answering questions), we also design corresponding incentive mechanism. In the opportunistic sensing mode, as most of the consumption comes from the devices, each participant will be paid a fixed amount of monetary reward for every sample (i.e., the estimated queuing time) reported to the server. In the participatory sensing mode, as the major resource consumed is participants' attention for answering the question, each participant will be paid a fixed amount of monetary reward for every two answers in the same POI. Note that single answer cannot enable the system to estimate the queuing time, the reward will only be given if a participant has answered a couple of questions.

CrowdQTE provides the service for end-users to query information through mobile clients or web browsers. In addition to querying the absolute queue time (e.g., 10 minutes), it is more useful to provide the users with visualized queuing status. Thus CrowdQTE integrates the digital map to give user a more intuitive understanding of queuing status of POIs within a certain region. We divide the queuing status into four levels, which are represented by four different colors[1)] (i.e., red, orange, yellow, green). An end-user specifies a geographical region and selects the interested type of POI, and

[1)] The mapping from absolute minutes to color is POI-specific

then the system can display the queue time information in the digital map (Fig. 2), through which the end-user can make a better plan. Moreover, end-users can subscribe notification service by publishing their visiting plans of a POI to the server-side, including the candidate period of time for visiting, and the queue time that she/he can bear with. As long as all constraints are satisfied, CrowdQTE will automatically notify the users through the mobile client or text messages.

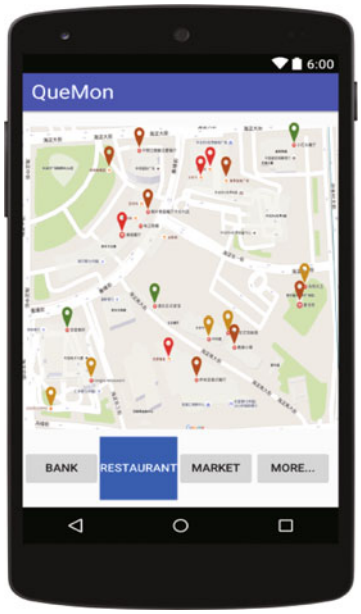


Fig. 2 Queuing status visualization

4 Queue monitoring

4.1 Participatory sensing mode

A naive approach is that when starting to wait, the participants press the button "waiting starts", and when it is his/her turn to accept the service, they press the button "waiting ends". Then the mobile phone side will record the queue time and upload to the server. However, this approach has a disadvantage that participants must remember to press a button when the waiting starts and ends. Otherwise, the calculated queue time would be inaccurate.

Therefore, we adopt the following mechanism to improve the accuracy with acceptable intrusiveness. First, in those ticket-holding scenarios, we assume that a waiting starts when a participant enters a certain place (e.g., hospital and bank lobby), so that the system automatically labels the time at the beginning of the waiting. It is a reasonable assumption, because according to our observations in most of the ticket-holding scenarios, participants will get a ticket from the queue management machine as soon as they enter certain POIs. Then CrowdQTE adopts a strategy by disturbing participants only twice. Once a participant is detected entering the POI, the question about "how many people is in front of you" is delivered to the client side, and the participant answers the question. Then after a short period of time (N minutes)[2], the same question will be asked again (Fig. 3). The system can estimate the queue time based on the time stamp and answers.

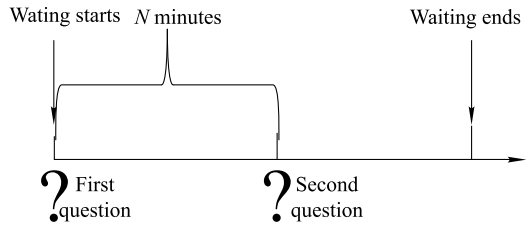


Fig. 3 Timing for reminder: an example

Assume that the number of people in front of a participant is A and B when the two questions are asked, respectively. The estimated queue time T can be calculated as $T = A * N/(A - B)$.

4.2 The opportunistic sensing mode

In this section, we will introduce how to detect the queue time when the opportunistic sensing mode is selected by the planner. In order to better explain the challenges and ideas, a use case of opportunistic sensing mode based queuing scenario is presented as follows. *Tom gets into the supermarket with a CrowdQTE equipped smartphone. CrowdQTE recognizes the location as soon as he enters one recorded supermarket. In the supermarket, Tom walks through shelves, browses goods, stops once in a while to read the commodity information, and puts some goods into his shopping cart. After finishing his shopping plan, Tom goes for checking out in a queue. Tom's movement in the queue repeats the pattern of standing-walking-standing.*

We divide the customer's movement during shopping in supermarkets into two modes, the queuing mode and the non-queuing mode. If a customer is in a queuing mode during the time period $[t_s, t_e]$, then two conditions must be satisfied:

• Condition 1 Before t_s and after t_e, the customer takes a continuous movement. The continuous movement is defined as a walk or running lasting for at least the time interval of

[2] Note that the setting of parameter N is relevant to the type of POI, which can be adjusted if we can collect adequate amount of historical data form these places

T_m.

Before t_s, the customer moves continuously to the end of a queue, and after t_e, she/he moves continuously again to leave the queue. T_m is set to 3 seconds in CrowdQTE.

• Condition 2 Between t_s and t_e, a customer's movement consists of interleaved queue-waiting and queue-moving. The queue-waiting is defined as a continuous still status lasting for the time less than T_w.

According to our training data set[3)], the possibility for the time of a queue-waiting to exceed 180 seconds is very small (0.12%), and T_w is set to 180 seconds in CrowdQTE. If a customer stands still for more than 180 seconds, CrowdQTE considers she/he is no longer in a queue. Also according to our training data set, with the possibility of 99.95%, the time of a queue-moving is less than 3 seconds (T_m is set to 3 seconds as above). Therefore, if a customer walks for more than 3 seconds, CrowdQTE considers she/he is no longer in a queue either.

Except for the queuing mode, all other customers' movements are considered to be in the non-queuing mode.

We found that, during the process of scanning and selecting goods, it is still possible for a customer's movement to satisfy the above two conditions, which leads to erroneous judgments. In order to solve the problem, we add a third condition as follows.

• Condition 3 The queuing processes occur nearby the cashiers, and the customer in a queue is gradually moving closer to a cashier. This condition is identified by leveraging ambient sounds.

CrowdQTE collects and processes sensing data aiming at recognizing the queuing mode. For the above three conditions describing the queuing mode, Conditions 1 and 2 are inspected based on the analysis of acceleration data gathered by the phone embedded accelerometers, and Condition 3 is inspected by analyzing the acoustic data gathered by microphone sensors. Considering the limitation of computing, storage and energy in phones, where data collection and queuing mode recognition are implemented, we design relatively simple but still effective algorithms.

The movement of customers in supermarkets includes moving (walking or running) and still (standing or seating). The difference of accelerometer data between these two movement statuses is considerably obvious. Figure 4 shows the acceleration data when a customer is standing still (Fig. 4(a)) and walking (Fig. 4(b)).

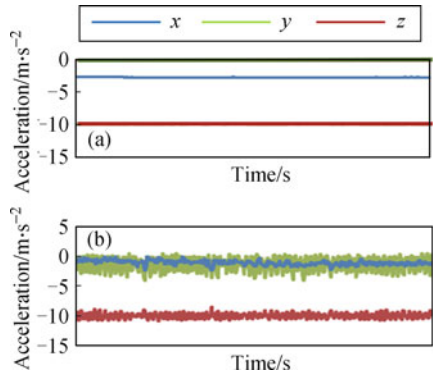


Fig. 4 Acceleration data when a customer is (a) standing and (b) walking

CrowdQTE collects acceleration data with a sampling rate of 5Hz. Every sample is taken as a 3-dimensional vector denoted as (X, Y, Z). Movement status at a given time point, which is called a movement primitive, is judged by calculating the angle between the current sample vector and the adjacent sample vector. There are two types of movement primitives, still or moving. By statistically analyzing a 400 minutes' training data set, we select 5 degrees as the threshold to differentiate still and moving, which reaches to an accuracy of 96.7%. The movement primitive is identified to be still when the angle between adjacent sample vectors is less than 5 degrees, otherwise to be moving, as illustrated in Fig. 5(a). Moreover, we use 0 and 1 to represent still or moving primitives respectively, and obtain a 0/1 series, which is called the movement identification series (MIS), as shown in Fig. 5(b).

Next, the MIS is processed as the following steps:

1) Block partitioning A block is a fragment of the MIS, which contains a series of continuous 0s or 1s, and is referred to as 0-block or 1-block respectively. The number of 0s or 1s in a block is referred to as the length of the block.

2) Noise removing The isolated 0 or 1 in the MIS is probably caused by noises in sensing data. Based on the statistical analysis of the training data set, CrowdQTE considers the block whose length is less than 2 as noise one, and the 0/1 value in a noise block will be modified and thus making the block be merged into the adjacent blocks.

3) Block tagging As mentioned above, a queue-waiting is less than 180 seconds, and a queue-moving is less than 3 seconds. As adopting a 5Hz sampling rate, the length of the 0-block representing queue-waiting is less than 900, and the length of the 1-block representing queue-moving is less than 15. CrowdQTE scans the MIS, and tags the 0-blocks whose length less than 900 as queue-waiting, tags the other 0-blocks as continuously-still, tags the 1-blocks whose length less than

[3)] The training datasets consist of queuing activity records from the volunteers. It records the start and end time of queuing, and the start and end time of each sub-activity (walk and stand) during the queuing

15 as queue-moving, and the other 1-blocks as continuously-moving.

4) Candidate queuing series recognizing CrowdQTE scans MIS with block tags, and all blocks series satisfying the following constraints are picked as candidate queuing series. (a) Constraint 1: the candidate queuing series are between two blocks tagged as continuously-moving. (b) Constraint 2: there is no block tagged as continuously-still in the series.

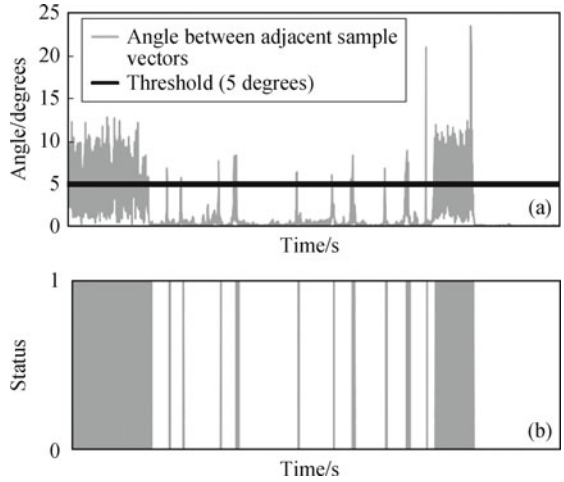


Fig. 5 Acceleration data transformation and interpretation. (a) Angles between adjacent sample vectors; (b) translating the acceleration data into a 0/1 series, called MIS (0: still; 1: moving)

CrowdQTE exploits other ambient contexts to distinguish the queuing mode from other similar behaviors. In this supermarket case, it utilizes ambient acoustic features to distinguish the queuing pattern from browsing and selecting goods. We conducted an investigation in 22 supermarkets varying in size in Beijing. We found that all supermarkets are equipped with POS machine and stylus printers for receipts. The POS machine makes a beep sound while scanning the barcode of a good, and the stylus printer makes a particular noisy sound while printing a receipt after a customer pays the bill. These two kinds of sounds are distinguishable and also pervasive in every supermarket, which can only be detected by phones near the cashiers. The accuracy of queuing recognition is significantly improved by combining the movement mode detection with the POS/printer noise recognition in CrowdQTE.

The above idea is implemented by using the MFCC [27] features, which are commonly used in audio recognition systems for speaker's identity or environmental sounds (e.g., music) automatic recognition. We collect the sound made by POS machines and stylus printers from different supermarkets, from which we extract the MFCC features as sound benchmarks. Then we store these benchmarks in CrowdQTE server. Based on the user's location and preferences, corresponding benchmarks are downloaded into phones. When both Condition 1 and Condition 2 are satisfied, microphones will collect acoustic data. The acoustic data is then divided into 2-second sound frames, from which the MFCC are extracted and compared to the benchmarks.

MFCCs have been shown to work well for structured sounds such as speech and music, but their performance degrades in the presence of noise. MFCCs are also not effective in analyzing noise-like signals that have a flat spectrum. Environmental audio contains a large and diverse variety of sounds, including those with strong temporal domain signatures, such as chirpings of insects and sounds of rain. These sounds are typically noise-like with a broad flat spectrum, which cannot be effectively modeled by MFCCs. In this work, we exploits the matching pursuit (MP) algorithm [28] to analyze environmental sounds. MP provides a way to extract time–frequency domain features that can classify sounds when only using frequency-domain features (e.g., MFCCs) fails. The process includes the decomposition of a signal from a dictionary of atoms, which yields the best set of functions to form an approximate representation.

When waiting in a line, a customer is moving continuously towards a cashier. The POS machine and printer are surely to be used before the customer leaves the queue. Consequently, for every candidate queuing series, when the sound of a POS machine or a printer is identified in its time duration, it is regarded as queuing series. Finally, when the queuing process is finished, the queue time can be measured by counting the length of the recognized queuing series.

5 Queue time estimation

After collecting a participant's queue time either in opportunistic or participatory sensing manner, the CrowdQTE's phone-side sends a measurement (called a queue time measurement instance, MI) along with current time stamps and POI ID to the server. After receiving the data coming from different mobile phones, the server-side will make an aggregation and estimate the current queue time.

A naive idea of the estimation is described as follows. First, CrowdQTE defines a time window with a fixed length (which is set to be 20 minutes in the current version). Then CrowdQTE put MIs belonging to the same POI and same time window into a same group (called MI-Group), and calculate the average value of each MI-Group as the queue time estimation.

In the naive method described above, we assume that ev-

ery MI is accurate and effective. However, it is not always the case. On the one hand, because precision of mobile phone sensors is limited and noise always exists in the environment, error may occur in the queuing mode recognition. On the other hand, participants' abnormal behaviors may also interrupt the queuing. For example, one may give up queuing when some emergent issues happen, such as suddenly recalling buying something else and leaving the queue for the sale zone again. As a result, in the aggregation process, the server must evaluate each MI's validity and eliminate invalid MIs.

CrowdQTE evaluates MI's validity on the basis of the assumption: participants waiting in a line tend to choose the shortest queue. So in the same POI and within the same time window (assume to be 20 minutes), the queue time measurements of different participants should be close to each other.

Based on this assumption, in a MI-Group, the MIs that are obviously distant from other MIs should be detected and eliminated from group. The algorithm evaluating MI's validation is described as follows.

Denote a MI-Group as $MIG = \{m_1, m_2, \ldots, m_i, \ldots, m_n\}$, where m_i is the value of each *MI* in the MI-Group, and n is the number of members in the MI-Group.

• Step 1 Compute the average value of members in a *MIG*, denoted as $\overline{m}$, $\overline{m} = \sum_{i=1}^{n} m_i/n$.

• Step 2 Compute $MIG' = \{m'_1, m'_2, \ldots, m'_i, \ldots, m'_n\}$, where $m'_i = m_l/\overline{m}$.

• Step 3 Compute the standard deviation of MIG', $S = \frac{\sum_{i=1}^{n}(m'_i-1)^2}{n-1}$.

• Step 4 Compare S and threshold D (a constant value), if $S > D$, execute Step 5, otherwise consider each *MI* in the *MIG* to be valid and exit. (Currently, D is set to be 0.2 in CrowdQTE, which is near-optimal according to our multi-round experiment.)

• Step 5 If $n > 3$, execute Step 6, otherwise consider each *MI* in the *MIG* to be valid, and exit. (Explanation: when $n \leqslant 3$, the number of *MI* is not sufficient enough to judge whether there is invalid *MI*, and all *MI*s are considered to be valid in this case.)

• Step 6 Select m'_d from MIG', where $m'_d = \max\{|m'_i - 1||m'_i \in MIG'\}$, label m_d to be invalid, and eliminate it from MIG. Set $MIG = MIG - \{m_d\}$, and go back to Step 1.

6 Evaluation

6.1 Data collection and experiment setups

We assume the data collected through the participatory sensing mode is accurate, so that we do not validate scenarios such as hospital and bank lobby. In this experiment, we evaluate the system's performance when the opportunistic sensing mode is adopted.

In our experiments, 12 volunteers participated in two big supermarkets. Experiments were conducted for 20 times, with 10 times in each supermarket. In each experiment, 12 volunteers got into a supermarket, each carrying a smartphone with the pre-installed CrowdQTE phone-side application (using three brands of smartphones including Samsung Nexus S, Huawei T8950, and Lenovo S880.). All phone-side applications were set to collect data and upload results of measurement. For each experiment, volunteers are required to finish their queuing during the same time window. All experiments were conducted on weekends, and five time intervals were selected varying in the crowd density. The selected time interval is showed in Table 1.

Table 1 Time windows of the experiments

Time window	Times of experiments
9:40am–10:00am	4
1:20pm–1:40pm	4
4:40pm–5:00pm	5
6:40pm–7:00pm	4
9:40pm–10:00pm	3

In each experiment, CrowdQTE's server-side estimated the queue time of a certain supermarket based on the measurements uploaded by eight volunteers, while other four volunteers' measurements were set as test data to evaluate the estimation. The condition of queue time in a supermarket was labeled as one of the four levels as shown in Table 2.

Table 2 Queue time classification

Queue time/min	Classification
Less than 5	Green
Between 5 and 10	Yellow
Between 10 and 15	Orange
More than 15	Red

6.2 Queue time estimation accuracy

In the related work, the closest system to ours is the QueueSense [26]. QueueSense is a queuing recognition system on mobile phones to assist in a queue management system. QueueSense extracts features of queuing behaviors and classifies queueing via collaboration among people waiting in line. It measures the disparity of people in different lines using relative position changing rate and partitions different queues using a hierarchical clustering approach. In this section, we will use the approach adopted by QueueSense as the

baseline, and compare its performance to CrowdQTE in opportunistic sensing mode.

We repeat the process described in Section 6.1 by installing the baseline system QueueSense with the same setting. In each round of experiment, we vary the total number of participants in the same time window. The queue time estimation accuracy of CrowdQTE and QueueSense is shown in Fig. 6. We use the classification in Table 2 as the metric to judge if the estimation is accurate.

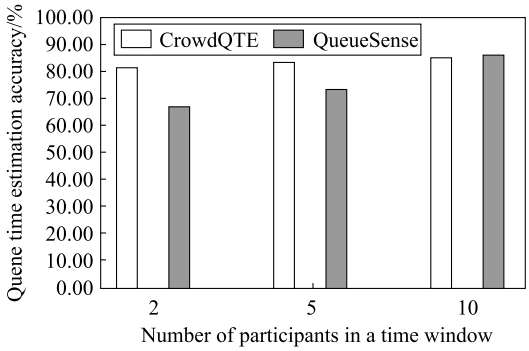


Fig. 6 Queue time estimation accuracy

From the results, we can see that when the number of participants is 2 and 5, our system outperforms the baseline system significantly. When the number of participants reaches to 10, their performance is almost the same (baseline system outperforms ours a little bit). Thus, compared to the baseline approach, the advantage of CrowdQTE is that it can achieve a good queue time estimation results even if there are only a few participants having installed the application in certain POIs. This characteristic is very important, because in many cases, especially when the system is online, we cannot expect a large number of registered users and assume there would be sufficient participants in each of the POIs.

The reason why our system outperformed QueueSense is the queue monitoring for CrowdQTE is that independent from the collaboration of participants, while the algorithms adopted by QueueSense rely on the behavior similarity analysis among different participants. Thus, when the number of participants is inadequate, the CrowdQTE outperforms QueueSense. In the future work, we will conduct more extensive evaluation when the number of participants is large (e.g., 20 or 60 participants).

6.3 Energy consumption

The backend running of CrowdQTE consumes the energy in participants' mobile phones, which mainly comes from the data collection (e.g., localization, accelerometer and microphone data) and queuing detection algorithms' execution. As energy consumption is crucial for mobile applications, we also evaluate it for CrowdQTE.

First, we deploy the phone-side application of CrowdQTE in five volunteers' mobile phones. At the same time, we install software to calculate the total energy consumption and the energy consumed by CrowdQTE every day, which lasts for a week. Second, we set six places on or nearby campus as the target POIs, including two supermarkets, two canteens, one hospital and one bank. The opportunistic sensing mode is adopted in supermarkets and canteens, while the participatory sensing is leveraged in the hospital and bank. Volunteers are required to pay a visit to each of the above POIs at least one time every day.

By collecting and analyzing the one-week energy consumption profiles of each volunteer, we calculate the proportion of the energy consumed by CrowdQTE in the total phone's consumption, which is shown in Fig. 7. From the results, we can see that the energy consumed by CrowdQTE accounts for 3.7%–8.1% of the total energy consumption.

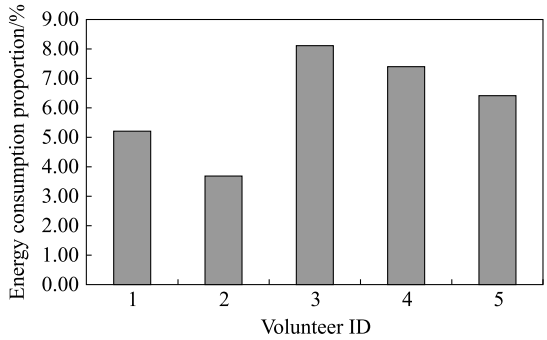


Fig. 7 The proportion of the energy consumed by CrowdQTE in total mobile phone's consumption

Note that each volunteer goes to at least six POIs every day in the experiment, the energy consumption of CrowdQTE is not significant compared to the total phone usage. Moreover, according to our life experience, the frequency we need to participate in a queue is commonly less than six times per day. Therefore, we can see that the energy consumption of CrowdQTE is acceptable in general cases, thus installing and running it will not influence the normal usage of a mobile phone.

6.4 Effectiveness of introducing ambient sound

In the experiment, we first gathered acoustic data in two supermarkets and extracted MFCC and MP features to be the benchmarks. For each supermarket, we collected the acoustic data within different distances from the cashier (less than 1m,

1m, 2m, 3m and 5m respectively). For each distance, data collection is conducted 30 times (each brand of phone collect 10 times), and the recognition recall is showed in Table 3. From the experiment we can see that the identification of the sound of POS machines and printers is closely related to the distance between the customers and the cashiers. Although it is difficult to recognize the special sound when people is far away from the cashier, it does not have negative impact on the system's performance. This is because people in the queue would finally move close to the cashier, so that the special sound will be detected when he/she gets closer to the cashier.

Table 3 Recognition recall of acoustic data of cashiers in supermarkets A and B

LOC	Data source	<1m	1m	2m	3m	5m
A	POS	86.7	83.3	50.0	6.7	0
	Printer	93.3	90.0	63.3	20.0	3.3
B	POS	90.0	90.0	60.0	13.3	0
	Printer	100	96.7	70.0	36.7	6.7

7 Conclusions and discussions

This paper presents CrowdQTE, a mobile crowdsensing system for providing real-time queue time information for different scenarios. In places where people wait in a line, accelerometer sensor data is collected to automatically detect the queueing behavior and calculate the queue time. In places where people do not wait in a line, the participants manually report the queuing status. We evaluate the performance of the system on commercially-available smartphones, and the results prove its effectiveness. In the future work, we plan to perform more extensive work as follows to make our system more applicable in real-world scenarios:

First, in order to incentivize our app, we plan to distribute fliers in the campus or through social networks (e.g., WeChat) in China. If there are a certain number of participants, we can retrieve readings daily from users regarding the wait-time of different POIs. With the accumulation of sensing data, CrowdQTE should be able to analyze the statistical queuing pattern in different POIs and time intervals, and predict the queue time in the near future by combining both current and historical information.

Second, there are more complex situations if the system is applied in the real-world settings. 1) As we discussed in Section 4.1, the time interval between two answers is a key parameter needs to be set for the participatory sensing mode. In the future work, we plan to conduct more extensive experiments in places such as hospitals and banks to study the adjustment of this parameter. 2) In the opportunistic sensing modes, movement status is judged by calculating the angle between the current sample vector and the adjacent sample vector, which is based on the assumption that the smartphone is in the pocket with a fixed position. However, in practice, people sometimes play smartphone in hand during the queue-waiting. Then the smartphone will move unavoidable and the judgement will be inaccurate. The possible solution is that the system continuously detects the usage of phone. If the phone is used, then the estimated queue time is labeled as invalid, which will not be used in the aggregation algorithm on the server side. 3) we utilize the ambient sounds collected by the microphone to distinguish the queuing from other similar behaviors. Nevertheless, we only take the supermarkets as use cases, which have obvious special sounds from POS machines and printers. Some other queuing place may not have the special sounds. Under such circumstances, we may have two possible solutions. The ideal one is to find additional contexts in such places and use them to distinguish the queuing from other similar behaviors. However, if we fail to find such contexts, then the system can adopt the participatory sensing mode, that is, asking the participant to record and report the queue time.

Third, the optimization of data collection frequency, including the localization, accelerometer and acoustic data, is important to reduce the energy consumption of the system. However, the optimization is relevant to personalized factors of a certain participant (e.g., mobility pattern and transportation mode). Therefore, in the future work, we try to deploy to a large number of users in real-world and collect adequate amount of historical data, and learn to optimize the localization frequency and conduct the evaluation more extensively.

Finally, like most mobile crowdsensing applications, location privacy is also an important issue that should be considered in real life deployment of CrowdQTE. To relieve participants' location privacy concerns, we plan to add an anonymization mechanism into CrowdQTE in future and allow the participants to upload data with anonymized IDs. Thus, the uploaded data cannot be linked to a particular user from the server side. Thanks to the advances in crowdsensing privacy-preserving incentive mechanisms, we still can pay these anonymized participants correct incentives [29].

Acknowledgements This work was mainly funded by the National Natural Science Foundation of China (Grant No. 61572048), Research Fund from China Electric Power Research Institute (JS71-16-005), and Microsoft Collaboration Research Grant. Besides, the work was partially supported by the Fundamental Research Funds for the Central Universities (106112015CD-

JXY180001), Open Research Fund Program of Shenzhen Key Laboratory of Spatial Smart Sensing and Services (Shenzhen University, China), and Chongqing Basic and Frontier Research Program (cstc2015jcyjA00016).

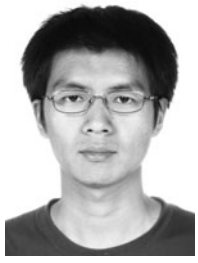
References

1. Kong D, Gray D, Tao H. Counting pedestrians in crowds using viewpoint invariant training. In: Proceedings of British Machine Vision Conference. 2005
2. Lin S F, Chen J Y, Chao H X. Estimation of number of people in crowded scenes using perspective transformation. IEEE Transactions on Systems, Man, and Cybernetics-Part A: Systems and Humans, 2001, 31(6): 645–654
3. Reisman P, Mano O, Avidan S, Shashua A. Crowd detection in video sequences. In: Proceedings of IEEE Intelligent Vehicles Symposium. 2004, 66–71
4. Huang X Y, Li L Y, Sim T. Stereo-based human head detection from crowd scenes. In: Proceedings of International Conference on Image Processing. 2004, 1353–1356
5. Mckenna S J, Jabri S, Duric Z, Rosenfeld A, Wechsler H. Tracking groups of people. Computer Vision and Image Understanding, 2000, 80(1): 42–56
6. Bauer D, Ray M, Seer S. Simple sensors used for measuring service times and counting pedestrians. Transportation Research Record: Journal of the Transportation Research Board, 2011, (2214): 77–84
7. Bullock D, Haseman R, Wasson J, Spitler R. Automated measurement of wait times at airport security. Transportation Research Record: Journal of the Transportation Research Board, 2010, (2177): 60–68
8. Wang Y, Yang J, Chen Y Y, Liu H B, Gruteser M, Martin R P. Tracking human queues using single-point signal monitoring. In: Proceedings of the 12th ACM Annual International Conference on Mobile Systems, Applications, and Services. 2014, 42–54
9. Ganti R K, Ye F, Lei H. Mobile crowdsensing: current state and future challenges. IEEE Communications Magazine, 2011, 49(11): 32–39
10. Guo B, Wang Z, Yu Z W, Wang Y, Yen N Y, Huang R H, Zhou X S. Mobile crowd sensing and computing: the review of an emerging human-powered sensing paradigm. ACM Computing Surveys, 2015, 48(1): 7
11. Koukoumidis E, Peh L S, Martonosi M R. SignalGuru: leveraging mobile phones for collaborative traffic signal schedule advisory. In: Proceedings of the 9th ACM International Conference on Mobile Systems, Applications, and Services. 2011, 127–140
12. Xu C R, Li S G, Liu G, Zhang Y Y, Miluzzo E, Chen Y F, Li J, Firner B. Crowd++: unsupervised speaker count with smartphones. In: Proceedings of the 2013 ACM International Joint Conference on Pervasive and Ubiquitous Computing. 2013, 43–52
13. Guo B, Chen H H, Yu Z W. FlierMeet: a mobile crowdsensing system for cross-space public information reposting, tagging, and sharing. IEEE Transactions on Mobile Computing, 2015, 14(10): 2020–2033
14. Rana R K, Chou C T, Kanhere S S, Bulusu N, Hu W. Ear-phone: an end-to-end participatory urban noise mapping system. In: Proceedings of the 9th ACM/IEEE International Conference on Information Processing in Sensor Networks. 2010, 105–116
15. Zheng Y, Liu F, Hsieh H P. U-Air: when urban air quality inference meets big data. In: Proceedings of the 19th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. 2013, 1436–1444
16. Zhou P F, Zheng Y Q, Li M. How long to wait? predicting bus arrival time with mobile phone based participatory sensing. In: Proceedings of the 10th ACM International Conference on Mobile Systems, Applications, and Services. 2012, 379–392
17. Lane N D, Miluzzo E, Lu H, Peebles D, Choudhury T, Campbell A T. A survey of mobile phone sensing. IEEE Communications magazine, 2010, 48(9): 140–150
18. Campbell A T, Eisenman S B, Lane N D, Miluzzo E, Peterson R A, Lu H, Zheng X, Musolesi M, Fodor K, Ahn G S. The rise of people-centric sensing. IEEE Internet Computing, 2008, 12(4): 12–21
19. Burke J, Estrin D, Hansen M, Parker A, Ramanathan N, Reddy S, Srivastava M B. Participatory sensing. Center for Embedded Network Sensing, 2006
20. Ma H D, Zhao D, Yuan P Y. Opportunities in mobile crowd sensing. IEEE Communications Magazine, 2014, 52(8), 29–35
21. Zhang D Q, Wang L Y, Xiong H Y, Guo, B. 4W1H in mobile crowd sensing. IEEE Communications Magazine, 2014, 52(8): 42–48
22. Chon Y H, Lane N D, Li F, Cha H J, Zhao F. Automatically characterizing places with opportunistic crowdsensing using smartphones. In: Proceedings of the 2012 ACM Conference on Ubiquitous Computing. 2012, 481–490
23. Faulkner M, Olson N, Chandy R, Krause J, Chandy K M, Krause A. The next big one: detecting earthquakes and other rare events from community-based sensors. In: Proceedings of the 10th IEEE International Conference on Information Processing in Sensor Networks. 2011, 13–24
24. Bao X, Choudhury R R. MoVi: mobile phone based video highlights via collaborative sensing. In: Proceedings of the 8th ACM Iinternational Conference on Mobile Systems, Applications, and Services. 2010, 357–370
25. Bulut M F, Yilmaz Y S, Demirbas M, Ferhatosmanoglu N, Ferhatosmanoglu H. Lineking: crowdsourced line wait-time estimation using smartphones. In: Proceedings of International Conference on Mobile Computing, Applications, and Services. 2013, 205–224
26. Li Q, Han Q, Cheng X Z, Sun L M, QueueSense: collaborative recognition of queuing behavior on mobile phones. IEEE Transactions on Mobile Computing, 2016, 15(1):60–73
27. Hossan M A, Memon S, Gregory M A. A novel approach for MFCC features extraction. In: Proceedings of the 4th IEEE International Conference on Signal Processing and Communication Systems. 2010
28. Chu S, Narayanan S, Kuo C J. Environmental sound recognition using MP-based features. In: Proceedings of IEEE International Conference on Acoustics, Speech and Signal Processing. 2008, 1–4
29. Li Q H, Cao G H. Providing privacy-aware incentives in mobile sensing systems. IEEE Transactions on Mobile Computing, 2016, 9770(5): 76–84

Jiangtao Wang received his PhD degree in Peking University (PKU), China in 2015. He is currently a postdoc researcher in Institute of Software, School of Electronics Engineering and Computer Science, PKU. His research interest includes ubiquitous computing, urban data analytics and software engineering.

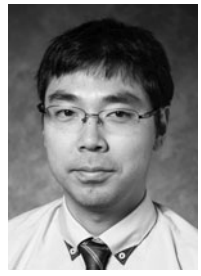
Yasha Wang received his PhD degree in Northeastern University, China in 2003. He is a professor and associate director of National Research & Engineering Center of Software Engineering in Peking University, China. He has published more than 50 papers in prestigious conferences and journals, such as ICWS, UbiComp, ICSP, etc. As a technical leader and manager, he has accomplished several key national projects on software engineering and smart cities. Cooperating with major smart-city solution providing companies, he carried out a lot of research work which has been widely adopted in more than 20 cities in China. His research interest includes urban data analytics, ubiquitous computing, software reuse, and online software development environment.

Daqing Zhang is a professor at Peking University, China and Télécom SudParis, France. He obtained his PhD from the University of Rome "La Sapienza," Italy in 1996. He served as the General or Program Chair for more than ten international conferences. He is an associate editor for ACM Transactions on Intelligent Systems and Technology, IEEE Transactions on Big Data, and others. His research interests include context-aware computing, urban computing,mobile computing, and so on.



Leye Wang obtained his PhD from Institut Mines-Télécom/Télécom SudParis and Université Pierre et Marie Curie, France in 2016. He received his MS and BS in computer science from Peking University, China. His research interests include mobile crowdsensing, social networks, and intelligent transportation systems.

Chao Chen is an associate professor at College of Computer Science, Chongqing University, China. He obtained his PhD degree from Pierre and Marie Curie University, France in 2014. His research interests include pervasive computing, urban logistics, data mining from large-scale taxi data, and big data analytics for smart cities.



Jae Woong Lee is an assistant professor in the School of Computer Science and Mathematics, University of Central Missouri, USA. He received the PhD degree from the Department of Computer and Information Science and Engineering, University of Florida, USA. His research focuses on modeling and simulation of human activities and sensor-based smart spaces, which especially advances assistive and intelligent systems for health cares. His research interests include human-centric environments, mobile health, data analytics and data science. He is currently researching future technologies equipped for smart cities and health informatics.

Yuanduo He received his bachelor degree in Peking University (PKU), China in 2014. He is currently an PhD student in the Institute of Software, School of Electronics Engineering and Computer Science, PKU. His research interests include ubiquitous computing, mobile computing, and data mining.