Urban Traffic Monitoring with the Help of Bus Riders

Pengfei Zhou, Shiqi Jiang and Mo Li

School of Computer Engineering, Nanyang Technological University
{pfzhou, sjiang004, limo}@ntu.edu.sg

***Abstract*—Real-time urban traffic conditions are critical to wide populations in the city and serve the needs of many transportation dependent applications. This paper presents our experience of building a participatory urban traffic monitoring system that exploits the power of bus riders' mobile phones. The system takes lightweight sensor hints and collects minimum set of cellular data from the bus riders' mobile phones. Based on such a participatory sensing framework, the system turns buses into dummy probes, monitors their travel statuses, and derives the instant traffic map of the city. Unlike previous works that rely on intrusive detection or full cooperation from "probe vehicles", our approach resorts to the crowd-participation of ordinary bus riders, who are the information source providers and major consumers of the final traffic output. The experiment results demonstrate the feasibility of such an approach achieving fine-grained traffic estimation with modest sensing and computation overhead at the crowd.**

I. Introduction

Real time urban traffic information is critical to wide populations living in the city. Comprehensive knowledge of instant urban traffic conditions contributes to commuter's better travel planning, improved urban transportation and commuting efficiency, reduced road congestion and waste emission, as well as other time and cost savings. For the past decades, increasing efforts have been put into exploring an accurate, efficient, and inexpensive way to instantly monitor the urban traffic conditions. Conventional methods rely on intrusive sensing, where people deploy infrastructural devices like magnetic loop detectors or traffic cameras at roadsides to actively detect traffic references. Installing such intrusive devices incurs substantial deployment and maintenance costs and can only provide limited observation at sparse positions. Recent studies resort to the GPS traces collected from "probe vehicles" like taxis or private cars to estimate the road traffic conditions [22], [25]. Such a passive probing method avoids cumbrous infrastructure deployment and enjoys flexible information extraction from the running probes in the city. Nevertheless, most of them largely rely on full cooperation from the probe vehicles and bear substantial cost in obtaining their location references.

In this paper, we describe our experience in building a participatory urban traffic monitoring system. Our system takes the operating buses as probe vehicles to sample the road traffic conditions. Instead of requesting any GPS traces from the transit agencies, we crowdsource the sensing jobs to public bus riders using their commodity off-the-shelf (COTS) mobile phones. The bus riders intelligently collect real time traffic sensing data on buses and anonymously upload the data. A backend server identifies and reorganizes the uploaded information from different buses, based on which the travel time and average vehicle speed at different road segments are estimated and a complete traffic map can be finally generated. We primarily rely on the cellular signatures together with the use of several lightweight sensing hints like audio and acceleration signals from the mobile phones rather than the energy expensive GPS data to derive the location references. Compared with existing approaches, the proposed traffic monitoring system provides a grassroots solution that solely relies on the collaborative efforts of the public. Built on the COTS mobile phones, our system obviates the need for special hardware extension and is energy friendly, which reduces operational overhead, encourages wide participation and expands the service coverage.

The full implementation of such a participatory monitoring system, however, entails substantial challenges which require practical solutions to cope with. First, tagging location with cellular references is non-trivial. The cellular signals themselves contain very rough location dependence. Precisely locating moving vehicles with only cellular signals may suffer from high localization error [16], [24] and huge overhead for complete war-driving [21]. In this work, we present a novel method which intellectually explores the invariant location and cellular attributes of bus stops so as to build a location mapping between the physical space and cellular space. Second, the crowdsourced sensing data are complicated and essentially carry error and noises. In this work, we carefully treat the sensing data on both mobile phones and the backend server. We do data cleaning at individual mobile phones and develop clustering and aggregation method on the backend server to process the joint data from all participants. We consolidate above techniques and implement a prototype system on Android phones and a laboratory server. During a 2-month experiment with 8 bus routes in a $\sim$25km^2 region in Singapore, we are able to collect the data input from 122 participants and derive instant traffic map of that region. The experimental results demonstrate high accuracy in mapping the traffic observations. The system power consumption is also carefully examined.

The rest of this paper is organized as follows. We introduce the background and motivation of urban traffic monitoring in §II. The methodology and detailed system design are presented in §III. The implementation and detailed evaluation results are reported in §IV. We introduce the related work in §V and conclude this paper and discuss possible future work in §VI.

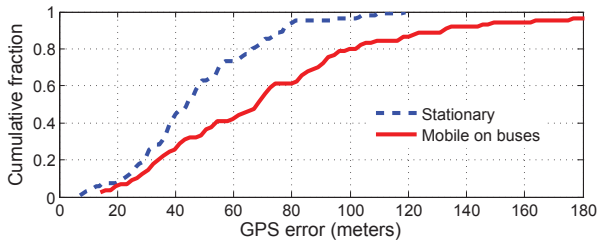


Fig. 1. Measurement of GPS localization errors in downtown Singapore.

II. Background and Motivation

Numerous approaches have been proposed to monitor the traffic conditions. Conventional traffic monitoring mainly relies on intrusive sensing infrastructures (e.g., the widely used magnetic loop detectors, roadside cameras and speed meters) to measure the spot speed of vehicles. The intrusive approach, however, suffers from two major drawbacks. First, due to the high implementation and maintenance cost, the systems are usually sparsely deployed and provide limited coverage. For example, a single loop detector costs \$900∼2000 depending on its type [19]. Second, measuring spot vehicle speeds at certain places may not accurately capture the travel delays along the whole road segments and introduce noises caused by traffic interruptions and congestions.

In order to overcome such drawbacks, many studies resort to using GPS traces from "probe vehicles" and measure the average travel time to derive the full traffic map. Modern public transport services cover most parts of the urban area [1], [2], [3], which provides readily available probe vehicles (e.g. numerous taxis) [18], [25]. Such an approach, however, is strongly dependent on the cooperation from the transit agencies and requires installation of real-time Automatic Vehicle Location (AVL) systems, which usually comes with substantial cost. With a fleet of thousands of vehicles, the installation of in-vehicle AVL systems incurs tens of millions of dollars [20].

In this paper, we leverage the public buses to probe the traffic conditions, but make use of bus riders' mobile phones, and fundamentally decomposes the individual "probing" tasks from the running buses. By doing so, our system does not rely on the cooperation of any transit agencies or even any particular probe vehicles. The crowd-participation of the bus riders themselves contribute the source sensing data and consume the final traffic output. We believe such an approach can be thus easily adopted to a wide range of different cities with slight modification. This work is fundamentally different from our previous work [27] on bus arrival prediction that the problem of traffic monitoring looks at holistic data collection and organization from unsorted bus trips rather than classifying and inspecting specific bus routes in [27]. In this paper, we focus on data fusion across bus riders' updates for traffic mapping and system scalability to support wider monitoring field. Instead of using GPS data, our system primarily relies on the energy-friendly and widely available sensing hints from the phones to intelligently track vehicles (e.g., cellular signals). No special hardware implementation or infrastructure devices are needed. The energy efficiency of sensing resources and automatic traffic data collection of the system brings negligible overhead to the bus riders and their phones, which as a result encourages user participation. There are some works [4], [20], [22], [25] that explore the GPS traces of COTS mobile phones to track vehicles or human movement. In this paper we do not employ GPS due to the following two major disadvantages. First, GPS suffers from big localization error in the downtown streets. To understand the magnitude of GPS tracking error, we perform a measurement study in downtown Singapore and summarize the GPS errors in Figure 1. We experiment with HTC sensation mobile phone and measure the GPS errors stationary or moving on buses by calculating the distance between the GPS position and the ground-truth position. The median errors are as high as 41m and 68m, respectively, and their 90th percentiles errors are 75m and 130m, respectively. Such big error is due to the complicated immediate surroundings in the downtown area, where the high buildings block the line-of-sight paths to GPS satellites and cause multipath problem. It is made worse when the phones are inside buses and the GPS signal is further attenuated. Similar results are also observed in other works [12], [20], [26], [17], which indicates that such a phenomenon is common for many cities across the world. Second, GPS device is known energy aggressive. We measure the energy consumption of the GPS receivers on Google Nexus One mobile phones with the Monsoon power monitor. Continuous GPS tracking incurs as high as 300mW energy consumption (details in §IV-D). Due to the limited battery capacity of COTS mobile phones, people usually turn off the GPS module to save power, which discourages user participation.

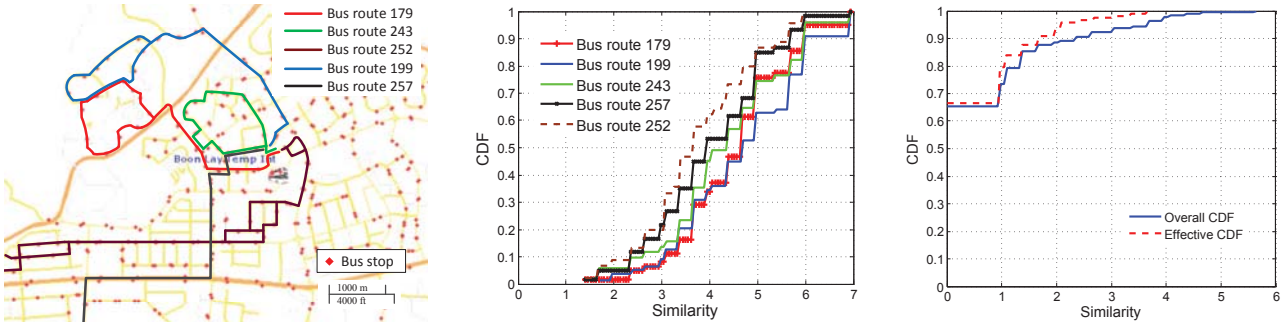
III. System Design

In this section, we first present the design methodology after which we detail the practical techniques in system design and implementation.

A. Design methodology

Bus routes have high coverage of the urban road systems. For example, the bus route coverage ratio is as high as 75% in London [2] and 79% in Singapore [1]. Figure 2(a) depicts the Jurong West area with a size of $\sim$25km^2 in Singapore, where $\sim$80% roads are covered by more than 20 bus routes. If we can track the moving trajectories of buses, we are then able to map down the probed traffic conditions in the region.

Solely using the cellular signals, however, is difficult to instantly track the buses. In urban area, the coverage of a typical cell tower is about 200∼900m, which cannot provide adequate location references. However, the inherent constraint of bus operation provides us a unique angle, i.e., buses strictly follow determined routes and stop at known bus stops. As Figure 2(a) depicts, more than 100 bus stops densely distribute in the region and separate the road systems into small road segments. The precise locations of the bus stops and detailed bus route operations are public information which is readily



(a) Measured bus routes.

(b) Similarity of the fingerprints collected at the same bus stop.

(c) Similarity of the fingerprints of different bus stops.

Fig. 2. Similarity measurement of bus stop fingerprints.

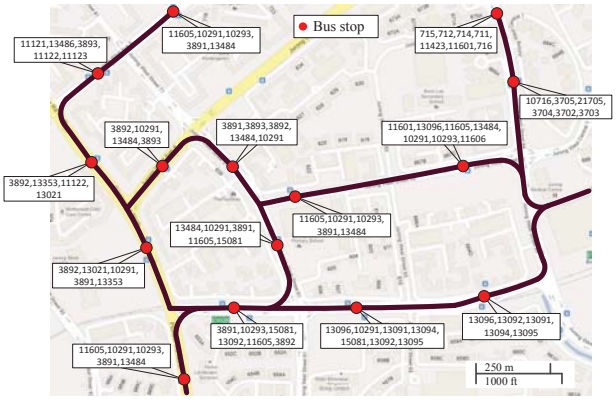


Fig. 3. An example area with the cellular fingerprints of 15 bus stops.

available on the web. Therefore, if we can track the stopping statuses of buses along the bus stops, we can naturally map their moving trajectories and derive the traffic estimations on the road segments in between bus stops. Assembling the traffic estimations of all segments gives us the whole traffic map. In order to implement such an idea, we consider fingerprinting the cellular signals at different bus stops and mapping them into the cellular space. Later we will be able to match those bus stops in cellular space according to the cellular fingerprints collected from the bus riders' mobile phones.

To understand the practical feasibility, we conduct a set of experimental studies to know how effectively the cellular signals can be used to distinguish different bus stops. We measure the cell tower signals at 86 bus stops of 5 bus routes (bus route 179, 199, 243, 252, and 257 in the region as shown in Figure 2(a)). We collect the cellular signals in two situations: when we stand at the bus stop and when we pass by the stops on a bus. Taking the time and weather factors into consideration, we collect the cell tower signals on days of different weather conditions and at different time of a day.

The mobile phone normally can capture the signals from multiple cell towers at one time, and chooses to connect to the one with the strongest signal strength. Typically there are 4$\sim$7 visible cell towers at each bus stop in our experiment. We order their cell IDs according to their Received Signal Strengths (RSS) and use such an ordered set to signature each bus stop in cellular space. Figure 3 depicts an example area where the cellular fingerprints of 15 bus stops are measured. For each bus stop, we collect the set of all visible cell towers and rank them in descending order of the RSS. The sets of cell IDs for different bus stops are highly different from each other. The bus stops well segment the road network.

We statistically analyze the similarities of the cell ID sets collected at the same bus stop (sef-similarity) in different runs. We use a matching algorithm (§III-C1) to calculate the similarities, where higher scores represent higher similarities. Figure 2(b) depicts the CDF of the self-similarity scores for all bus stops of the 5 routes. We see that the overall self-similarity score is very high. Generally 90% of the similarity scores are higher than 3 and more than 50% of the similarity scores are higher than 4, which demonstrates that the cell ID sets are adequately stable to signature bus stops.

We also analyze the similarities of the cell ID sets collected from different bus stops and plot the overall CDF of similarity scores in Figure 2(c). We see that for over 70% of the bus stops, their similarities are scored as 0 (no common cell IDs at all) and for over 90% of the bus stops, they have similarity scores lower than 2. We further examine the measurement data and find that most of those similarity scores higher than 3 are from the cell ID sets of two bus stops at opposite sides of the two-way roads. In terms of location reference, they can be treated as the same bus stop. Such treatment is proper and does not degrade our system because the uploaded trip is time-stamped, from which we can derive the bus moving direction and map the traffic estimation to the correct side of the road. We plot the effective CDF in Figure 2(c) with such treatment and we see that more than 94% bus stops have similarity scores lower than 2. The results suggest the feasibility of using cellular signal fingerprints to distinguish different bus stops.

We choose cell tower signals over other possible wireless signals to provide location inferences mainly due to the following considerations. First, mobile phones always maintain connections to nearby cell towers to support telephone calls and SMS service. The marginal energy consumption of collecting cell tower signals is negligible. Frequent scanning of other

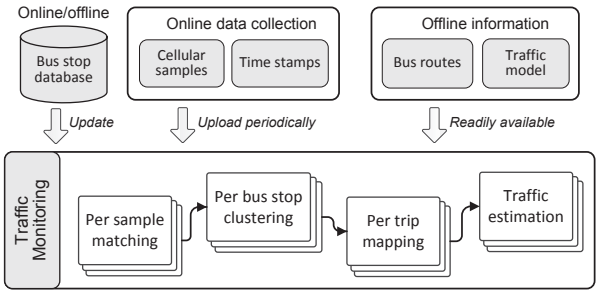


Fig. 4. System architecture and workflow.

wireless signals like WiFi, however, consumes much extra energy [21]. Second, cellular network has almost complete coverage of the entire city while other wireless signals usually suffer from poor signal availability in many outdoor areas. Third, the cellular signal sources are much more consistent over time than other wireless signal sources like WiFi hotspots, many of which are ad hoc and transient. The database built on cellular signals is thus more stable and easier to maintain.

Based on such a methodology, we can effectively segment the road network using bus stops and map down the traffic. Figure 4 sketches the system workflow, concerning two major components, i.e., online/offline data collection and trajectory mapping for traffic estimation. The bus stop data database can be updated in an online/offline manner. The real time trips of bus riders are uploaded periodically. The offline bus route and traffic model information is readily available. The backend server carefully maps down the real time trips to derive accurate traffic conditions. Each component will be elaborated in the following subsections.

B. Data collection

As depicted in Figure 4 (top), the major system input comprises three data sources.

Bus riders. The participatory bus riders serve as probes on buses. The online collection of their trips starts automatically when and only when users are detected on buses. We apply a beep detection approach similar with that in our previous work [27] to recognize the bus. Nowadays bus operators widely employ IC card systems where most bus riders pay the fees by tapping IC cards on the card readers which generate unique beep response, e.g., EZ-link [1] in Singapore, Oyster [2] in London, and MetroCard [3] in New York, etc. The beeps from card readers are always made of audio signals of specific frequencies (e.g., a combination of 1kHz and 3kHz audio signals in Singapore and 2.4kHz in London).

We apply the Goertzel algorithm [5] to perform beep detection instead of FFT used in [27] to extract specific frequencies (with prior knowledge of frequency components in the beep) rather than all frequencies which significantly saves energy. We measure and normallize the signal strength of several typical frequency bands. If the signal strength of the 1kHz and 3kHz frequency band obviously jumps (an empirical threshold of three standard deviation), we confirm the detection. We use the standard sliding window averaging with window size $w = 300ms$ to filter out the noises and increase the robustness.

Once detecting the beep, the mobile phone starts recording a trip. For each thereafter detected beep event, the mobile phone attaches a timestamp and the set of visible cell tower signals. The sensing data on the mobile phone thus record a sequence of timestamped cellular samples in the trip. The mobile phone concludes the current trip if no beep is detected for 10 minutes, and starts uploading another independent trip when new beeps are thereafter detected. We first primitively filter out the noisy beep detections (e.g., the rapid train stations use the same IC card readers) by thresholding the acceleration variance (measured by the acclerometer) to distinguish the people mobility pattern on rapid trains from taking buses. It is based on the observation that buses usually move with frequent acceleration, deceleration and turns, while rapid trains are operated more smoothly.

Bus stop database. There is a database storing cellular fingerprints of all bus stops which can be built online/offline. The server relies on this database to identify the bus stops for the uploaded cellular samples.

Bus routes and traffic model. The information of bus operational routes is readily available from bus operators and imposes constraints on how bus stops can be passed, which we will exploit for trajectory mapping. There have been available traffic models [10], [11], [18] giving the relationship between the travel speeds of public buses and ordinary automobiles, which we will adhere to for deriving the general traffic conditions from buses.

For all bus stops, we aggregate the bus stops located at the same location but different sides of the road as one, and record their relative locations in the bus operational routes. To yield the general traffic conditions from the bus speed estimation, we make use of a traffic model [10] which studies the relationship between public buses and ordinary automobiles. We implement the model to our data set and choose appropriate parameters according to our experiments (§III-D).

C. Trajectory mapping

As depicted in Figure 4 (bottom), we do trajectory mapping using bus stops as landmarks. We consider the received sequence of cellular samples of each independent trip, based on which the backend server identifies the passing by bus stops and maps the trip trajectory down. Three levels of mapping are done to refine the accuracy.

1) Per sample matching: With beep detection, the mobile phone not only recognizes buses but also indicates the arrival at bus stops because people only tap their IC cards at bus stops for paying transit fees and the card readers are usually disabled after buses move away from bus stops. Each uploaded cellular sample thus corresponds to a particular bus stop.

We match each cellular sample from one trip to a signature set stored in the fingerprint database and classify the sample into one bus stop. Many algorithms, like *k*-means clustering, have been used for fingerprint matching. In our system, the

TABLE I
BUS STOP MATCHING INSTANCE.

c_{upload}	1	2	3	4	5	Match	Gap	Mismatch	$\sum$
$c_{database}$	1	7	3		5				
Match	**1**	×	**3**	-	**5**	3	1	1	2.4
Score	+1	-0.3	+1	-0.3	+1				

cellular samples at bus stops may be collected under different conditions (e.g., on/off buses, different weather, etc.). While the cell tower RSS values may vary, their rank always preserves. Thus we use the modified Smith-Waterman algorithm [23] which focuses on the orders rather than the absolute RSS value to score the similarity of different sets.

The backend server arranges the cell tower IDs in the set in the descending order of each cell tower's RSS. We denote the cell tower set of a cellular sample $e(x)$ as $c_{upload} = \{u_1, u_2, \cdots, u_l\}$ ordered by the RSS of the l cell towers. (i.e., $s_i \geq s_j, 1 \leq i \leq j$), where u_i and s_i denote the cell IDs of cell tower i and its RSS, respectively. We denote the cellular fingerprint of a given bus stop $b(y)$ in the database as $c_{database} = \{d_1, d_2, \cdots, d_q\}$ also ordered by the RSS, where q is the set length. We match $e(x)$ and $b(y)$ by comparing the similarity of the two sets. c_{upload} typically has a different length with $c_{database}$. For each cell tower set, Smith-Waterman algorithm compares the segments of all possible lengths to find out the optimal alignment with one cellular fingerprint in the database, and uses a scoring system to weigh the value of *match*, *mismatch* and *gap*.

We modify the Smith-Waterman algorithm settings to adapt to our system. The performance of such a matching algorithm is mainly determined by two factors, the set length and the penalty cost for gaps and mismatches. In our system, the set length of the fingerprint for each bus stop (the number of cell towers) is about $4 \sim 7$ which is sufficient for the matching algorithm. We vary the value of mismatch penalty cost from -0.1 to -0.9 and simulate the matching accuracy. Choosing -0.3 as the penalty cost gives the best result. Table I shows an illustrative example where the uploaded cell tower set is $c_{upload} = \langle 1, 2, 3, 4, 5\rangle$ and compared with $c_{database} = \langle 1, 7, 3, 5\rangle$. The matching algorithm scores 2.4 by aggregating 3 matches, 1 gap and 1 mismatch. We denote $Sim(e(x), b(y))$ as the similarity score of one cellular sample $e(x)$ and that of an actual bus stop $b(y)$. The server selects the bus stop with the highest similarity score from bus stop candidates. To filter out noisy reports, we only keep the cellular sample when its highest matching score with the candidate bus stops exceeds a threshold γ. We set $\gamma = 2$ according to the measurement results in Figure 1(b). All cellular samples whose highest similarity score is lower than 2 are discarded without further processing. If there are more than one matched bus stop, the one with a larger number of common cell IDs is selected. We denote $M(e(x), b(y)) = 1$ if the matching result of $e(x)$ is $b(y)$, and $M(e(x), b(y)) = 0$ otherwise.

2) Per bus stop clustering: When a bus arrives at a bus stop, there are usually a number of passengers boarding and alighting giving multiple beeps, and multiple cellular samples are taken. They will be received in a time sequence at the server, allowing us information redundancy for better reliability in identifying the correct bus stop. We co-cluster the individual cellular samples according to their matched bus stops and timestamps, and identify the bus stop for closely clustered reports with more confidence.

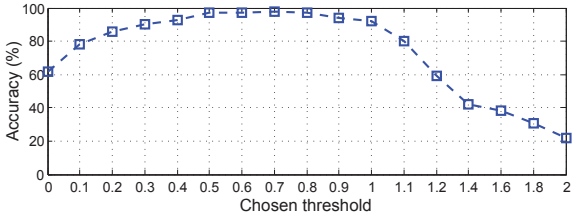


Fig. 5. Setting threshold ε.

For a sequence of m cellular samples $E = \{e_1, e_2, \ldots, e_m\}$ with timestamps $T = \{t_1, t_2, \ldots, t_m\}$, their matching results are a set of corresponding bus stops $\{b_1, b_2, \ldots, b_m\}$ with their similarity scores $\{s_1, s_2, \ldots, s_m\}$. In our co-clustering algorithm, we denote the maximum possible similarity score as s_0 and the maximum possible time interval between two cellular samples for the same bus stop as t_0. In our system implementation, parameters s_0 and t_0 are set to 7 and 30 secs, respectively. For two samples e_i and e_j, we weigh their matching relationship as

$$L(e_i, e_j) = \begin{cases} \frac{s_0 - |s_j - s_i|}{s_0}, & \text{if } b_i = b_j \\ 0, & \text{otherwise.} \end{cases}$$

Adding the timestamp information, we put e_i and e_j into the same cluster if

$$\frac{t_0 - |t_j - t_i|}{t_0} + L(e_i, e_j) > \varepsilon, \tag{1}$$

i.e., two samples are classified into the same cluster if they are detected close in time and have similar matching results. We use a threshold parameter ε to verdict their relationship. We vary ε from 0 to 2 with step length 0.1 and test the clustering accuracy according to an experiment trial with bus route 243. The result is plotted in Figure 5. If the threshold is too small, cellular samples from different bus stops might be treated as one cluster while if it is too big, the samples at the same bus stop might be classified into different clusters. Nevertheless, we find the result reasonably tolerates threshold selection. We can get satisfactory clustering accuracy with $\varepsilon = 0.3 \sim 1$. In our later system implementation, we choose $\varepsilon = 0.6$.

Figure 6 shows an example with a sequence of cellular samples collected from one trip. The backend server clusters the cellular samples into 2 clusters corresponding to bus stop i and j and extracts the arrival time and departing time at each bus stop, which will later be used to estimate the bus traveling speed (in §III-D).

After the cellular sample clustering, we get a sequence of n clusters $\{C_1, C_2, \ldots, C_n\}$. Each cluster C_i corresponds to a bus stop. As the cellular samples in the same cluster may

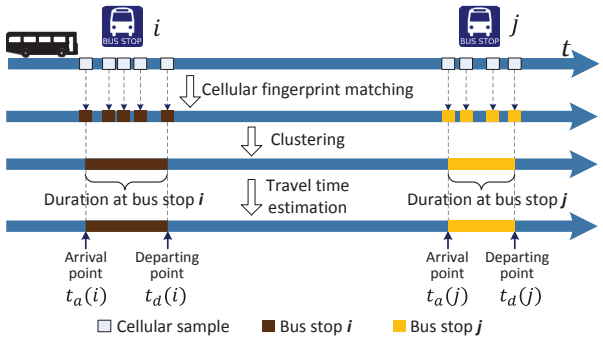


Fig. 6. Bus stop clustering of an example trip.

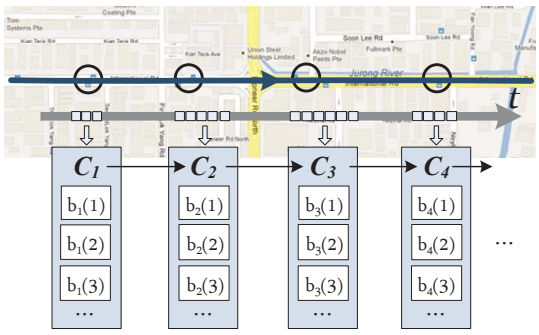


Fig. 7. Bus stop identification with a cluster sequence.

match different bus stops due to noises, at this moment each cluster C_i is associated with a pool of potential candidate bus stops as Figure 7 shows (though our experiments suggest that for most clusters there is only one candidate in the pool).

3) Per trip mapping: The operation of bus routes largely constrains the possible combinations and sequences the bus stops can be visited. Considering such constraints, we can further filter out the impossible bus stop candidates and map each cluster of cellular samples to a sole bus stop. As shown in Figure 7, a sequence of n cellular sample clusters are outputted from the previous step. In each cluster $C_k (k = 1, 2, \ldots, n)$, there are a total number of E_k samples $\{e_k(1), e_k(2), \ldots, e_k(E_k)\}$, and B_k potential candidate bus stops $\{b_k(1), b_k(2), \ldots, b_k(B_k)\}$. Each candidate bus stop $b_k(i)$ is assigned a probability $p_k(i) = \frac{\sum_{j=1}^{E_k} M(e_k(j), b_k(i))}{E_k}$ and an average similarity $\overline{s}_k(i) = \frac{\sum_{j=1}^{E_k} [M(e_k(j), b_k(i)) \cdot Sim(e_k(j), b_k(i))]}{\sum_{j=1}^{E_k} M(e_k(j), b_k(i))}$.

Our goal is to find out a segment from one bus route or all possible concatenation of multiple bus routes that best matches the current trip and successively derive the most "correct" bus stop for each sample cluster. For two actual bus stops x and y, we denote their relationship as $R(x, y) = 1$ if y is "behind" x in some bus route, which means that buses might arrive at y after passing by x, and $R(x, y) = 0$ if $x = y$, and $R(x, y) = -1$ for the rest. Since there are probably more than 1 bus stop candidates for some clusters, we can get a set of all possible bus stop sequences $S = \{S_1, S_2, \ldots, S_N\}$, where $N = \prod_{k=1}^{n} B_k$. Each S_j is comprised of a sequence of n bus stops $\{b_1(a_j(1)), b_2(a_j(2)), \ldots, b_n(a_j(n))\}$. We use maximum likelihood estimation to find the best matching sequence

$$S^* = \arg \max_{S_{j:1\sim N}} \{p_1(a_j(1)) \cdot \overline{s}_1(a_j(1)) + \sum_{i=2}^{n} [p_i(a_j(i)) \cdot \overline{s}_i(a_j(i)) \cdot R(b_{i-1}(a_j(i-1)), b_i(a_j(i)))]\}. \quad (2)$$

In Equation (2), we weigh a sequence S_j using both the probabilities $p_i(a_j(i))$ and average similarities $\overline{s}_i(a_j(i))$. The best matching sequence S^* finally maps down the trajectory of the trip, and determines the "most likely" bus stop for each cellular sample cluster on the trajectory.

D. Traffic estimation

Based on the mapping result, the backend server estimates the average travel speed on road segments divided by bus stops. As illustrated in Figure 6, for each uploaded trip, we are able to identify the passing by bus stops and extract the arrival time and departing time of each bus stop. We denote the arrival time at bus stop i as $t_a(i)$ and the departing time as $t_d(i)$. The travel time between i and j is thus estimated as $t_{ij} = t_a(j) - t_d(i)$. In practice, the bus may not stop at one particular bus stop if there is no bus rider boarding or alighting, and thus the information at the bus stop is missing. In such cases, our method automatically treats the combined two adjacent segments as one.

The bus travel time may not directly yield the general traffic conditions. Public buses have more frequent stops and usually adhere to more strict speed limits. The relationship between the transit buses and general traffic conditions has been studied in transportation domain [18], [10], [11]. We use a linear traffic model similar with that in [10] to estimate ATT from BTT:

$$\text{ATT} = a + b\text{BTT}, \quad (3)$$

where $a = \frac{road\ length}{free\ travel\ speed}$, representing the average travel time of an automobile when there is little or no traffic, and b represents the effect of traffic congestion (as measured by the running time of buses) on ATT. The value of b can be determined using linear regression and our experimental measurement suggests that the value of b lies within a narrow range $[0.13, 0.18]$ for most road segments. For simplicity, we select $b = 0.15$ for all road segments.

When we consider the trip reports from massive mobile phones, for each road segment, there are typically more than one speed estimation. In our system, we use a Bayesian method [18] to combine the initial estimation with new data input. We denote the variance of the historic mean speed $\bar{v}_0$ as σ_0^2 and the variance of new mean speed $\bar{v}$ as σ^2. Then the updated speed estimation is normal with mean speed $\bar{v}_{new}$ and variance σ_{new}^2

$$\bar{v}_{new} = \frac{\frac{\bar{v}_0}{\sigma_0^2} + \frac{\bar{v}}{\sigma^2}}{\frac{1}{\sigma_0^2} + \frac{1}{\sigma^2}}, \quad \sigma_{new}^2 = \frac{1}{\frac{1}{\sigma_0^2} + \frac{1}{\sigma^2}}. \quad (4)$$

The updating procedure follows Equation (4) and produces sequential travel speed estimations. The updating procedure

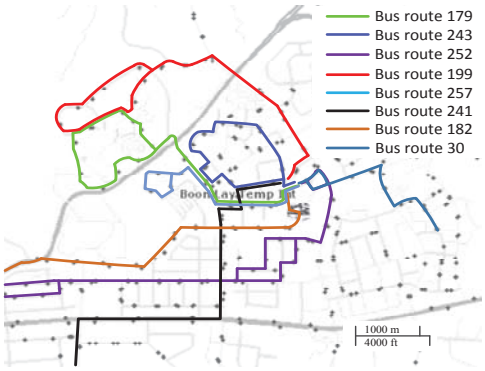


Fig. 8. 8 concerned bus routes in the $\sim 25km^2$ implementation area.

uses the inverse of the estimation variance to weigh the historic estimation and the updated estimations. The travel speed estimation on each road segment is updated with a period of T. In our experimental implementation, we set $T = 15$ mins.

IV. Implementation and Evaluation

We implement a prototype system on the Android platform with different types of mobile phones, and experiment with real data from two resources, manual collection and participatory collection, over a 2-month period. In this section, we introduce the experiment methodology and detailed results. we first present our experiment environment details and methodology (§IV-A). We test and evaluate the performance of bus stop identification (§IV-B) and show the traffic estimation results during the experiment period (§IV-C) and compare the estimation results with the official traffic data from the transit angency. The system overhead is also carefully investigated (§IV-D).

A. Experiment methodology

Mobile phones. We develop the data collection app with Android phones in Android OS 4.0.3 with API version 15. We do controlled experiments mainly with three types of mobile phones, i.e., HTC Sensation XE, HTC Desire S, and Google Nexus 4 and 5. All types of mobile phones can easily support the light computation and sensing overhead in our app. As our system is independent of platforms, we believe that the proposed method can be easily implanted to other OS and hardware platforms, such as Apple iOS and Windows Phone.

Backend server. We implement a backend server in Java for our experiments. It is running on the DELL Precision WorkStation T3500 with 6GB memory and Intel Xeon(R) CPU W3565 @ 3.20GHz(4 CPUs). It provides database update and receives data from the participants.

Experiment environment. The public bus transit system serves millions of bus riders every day covering most parts of Singapore [27]. It is commercially operated by two major transit companies, SBS Transit [6] and SMRT Corporation [15]. IC card systems are widely used for paying transit fees. Multiple card readers are installed on each SBS or SMRT bus for collecting fees. Figure 8 depicts the experiment region in Singapore. Public transit buses periodically run on more than 20 bus routes covering most of the roads in this $7km \times 4km$ area. Our experiment primarily concerns 8 bus routes, i.e, bus route 179, 199, 241, 243, 252, 257, 182 and partial part of route 30. These 8 bus routes cover a major portion of the road system in this area. We did extensive experiments to study our system design feasibility and evaluate the system performance. The performance of each system component was carefully examined. The experiments started from Jan. 2013, ended in Mar. 2014, and took more than 2 months in total.

TABLE II
Bus stop identification accuracy.

Route	total	errors	error rate	1 stop error	2 stops error
182	121	8	6.61%	5	2
30	58	3	3.45%	3	0
241	80	6	7.5%	5	1
199	93	5	5.38%	4	1

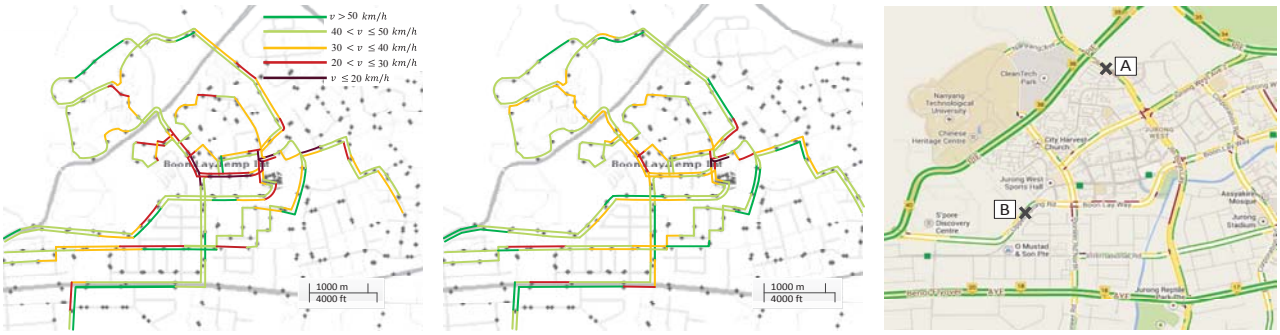
Data collection. The data used in our system contain two parts: the fingerprint database of bus stops and the real time sensing data from the participatory mobile phones. In our experiments, there are two data resources.

We manually collect the cellular fingerprints of the bus stops on the 8 bus routes. For each bus stop, multiple cellular samples are primitively collected and the sample with the highest similarity with the rest samples is chosen as the fingerprint and stored in the database. The cellular data used in §III-A and the evaluation data for bus stop identification used in §IV-B are also manually collected. The Land Transport Authority (LTA) [7] of Singapore also provides us their traffic data measured from the AVL reports of over 10,000 moving taxis, which we take as ground truth in the experiment.

During the experiment, 122 participants, mainly undergraduate students and volunteers, contribute real time bus information to the backend server. The data collection app is installed in each participant's mobile phone and uploads the sensory data through WiFi or 3G network. In the first month, we receive limited data from the participatory bus riders due to their small number. The data concentrate on frequent taken bus routes. In order to comprehensively evaluate the system performance with wider participation, we encourage (with vouchers) the participants to intensively take buses for 9 days in total and provide richer traffic data for our system. The experiment results of both the sparse and intensive data collection stages are shown in §IV-C.

B. Bus stop identification

The accuracy of bus stop identification for the 8 experimental bus routes is shown in Table II. In order to evaluate our bus stop identification algorithm, we take buses to collect the cellular signals at bus stops for 8 rounds. The cellular signals in one of the 8 runs are used as the fingerprints stored in the database. The rest 7 runs of data are used to identify the bus stops. In Table II, we summarize the statistical results of the bus stop identification error for 4 bus routes. The results of the other 4 bus routes are similar. The bus identification error is



(a) 8:30AM. (b) 15:00PM. (c) Google Maps.

Fig. 9. 2 snapshots of the traffic map at different time of a day and the Google Maps traffic.

smaller than 8% for all the 4 bus routes. Bus route 241 has the highest identification error and it has 13 effective bus stops. In this experiment, we analyze 80 cellular sets collected from its bus stops and 6 of them are mis-identified. In the mis-identified cases, the results of 6 cases are 1 bus stop away from the actual bus stop and only 1 case is 2 bus stops away. The mis-identification cases have little impact on the system performance. The high accuracy of bus stop identification guarantees the accuracy of travel speed estimation.

C. Traffic estimation

In this section, we report our experiment results from the participatory sensing data, and compare the results with the official data provided by LTA.

Figure 9 depicts 2 snapshots of the traffic maps at different time points (8:30AM and 15:00PM) on an experiment day when we encouraged most participants to intensively take buses. We show the travel speed of automobiles in 5 levels as shown in Figure 9(a). The average moving speed is mainly 30-50 km/h. Traffic conditions in the studied area are spatially different. For example, for the traffic condition at 8:30AM shown in Figure 9(a), the highest moving speed is higher than 50 km/h (left and bottom sides) while, in contrary, the lowest speed is as low as 20 km/h (middle side). Meanwhile, the distribution of the traffic at the 2 time points are also different. The overall moving speed at 15:00PM (Figure 9(b)) is relatively higher. There are few road segments at 15:00PM with travel speed lower than 20 km/h. The low-speed road segments at 8:30AM are close to each other in 2 main roads in the middle of Figure 9(a) where there are routine bus shuttles between a university and a rapid train station every 15 minutes every morning. As Figure 9 shows, although we only concern 8 bus routes, the coverage for the roads in the area is higher than 50% and they cover most major roads. The coverage ratio is much higher than the Google Maps traffic for the area (Figure 9(c)). We believe that we can obtain more comprehensive traffic conditions if more bus routes are concerned.

We compare our traffic estimation results with the official traffic data that we acquire from LTA [7]. We pick 2 typical road segments (A and B, as shown in Figure (c)) for comparison. Figure 10 compares the travel speed estimation

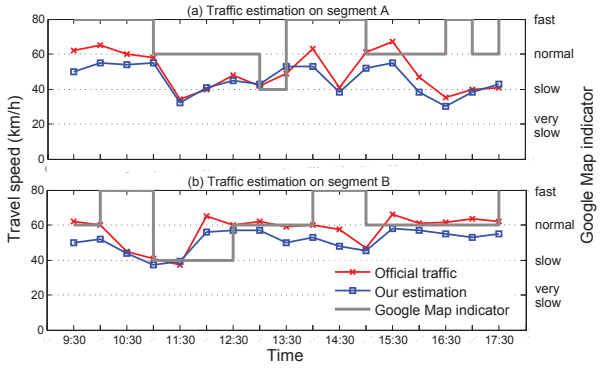


Fig. 10. Traffic estimation compared with official traffic and Google Map indicator.

of automobiles (v_A) from our system, the travel speed (v_T) from official traffic data, and the traffic conditions indicated from Google Maps on the two road segments for the time period from 9:30AM to 17:30PM on another experiment day. 17 values are plotted for v_A and v_T respectively, each of which is an average speed for a 15-minute window. The traffic conditions from Google Maps are not the exact travel speeds but 4 traffic levels indicated as *very slow*, *slow*, *normal*, and *fast*. As depicted in Figure 10, Google Maps only provide rough traffic levels which are not fine-grained in time and may not accurately reflect the instant traffic conditions. v_A and v_T are more sensitive to the traffic variation.

When we compare v_A and v_T on the two segments, they are not always perfectly matching with each other but exhibit interesting relationships. The speed estimates of v_A are divided into 3 groups, i.e., low speed ($<$ 45 km/h), medium speed (40$\sim$50 km/h) and high speed ($>$50 km/h). When the travel speed is low, v_A perfectly matches v_T. When the travel speed is high, there is usually a gap between v_A and v_T. This is probably because v_A is the general traffic estimation derived from the bus speeds, which are usually capped by lower speed limits, while taxis on the other hand usually travel more aggressively and yield a higher v_T when the traffic is light. Nevertheless, we clearly observe that v_A follows the variation pattern of v_T for most of the time.

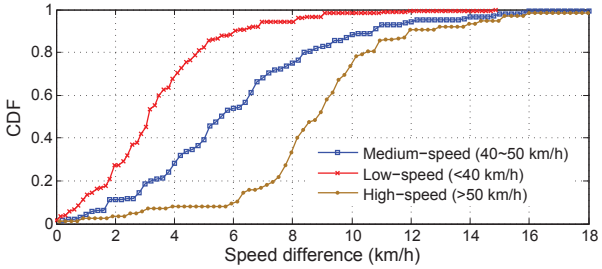


Fig. 11. Speed difference Δv between our traffic estimation v_A and the official traffic data v_T.

In Figure 11, we statistically summarize the available estimation results during the 2-month experiment, and plot the CDF of the speed difference Δv between v_T and v_A for all road segments and time durations when both are available. We categorize the comparison cases into 3 types, i.e., high-speed traffics ($v_A > 50km/h$), medium-speed traffics ($40 \leq v_A \leq 50km/h$) and low-speed traffics ($v_A < 40km/h$) and plot them separately. The majority of the studied cases are of medium-speed traffics. As depicted in Figure 11, Δv is the lowest (mostly about $3 \sim 5$) for low-speed traffics and the highest (mostly about $8 \sim 12$) for high-speed traffics. For medium-speed traffics, Δv is more disperse across $0 \sim 12$. The results suggest that the estimated traffic speed from our system generally provides a good measure of the traffic conditions. It is particularly indicative for heavy traffics and congestions that usually lead to low road speed.

D. System overhead

The computation complexity of the algorithms on mobile phones is bounded by the Goertzel algorithm used for the frequency extraction. The complexity of Goertzel algorithm is $O(K_g NM)$ and that of FFT is $O(K_f N \log N)$, where K_g and K_f are the "cost of operation per unit" of the two algorithms, respectively. M is the number of measured frequencies and N is the sampling values. When the number of calculated terms M is smaller than $\log N$, the advantage of the Goertzel algorithm is obvious. As FFT code is comparatively more complex, the factor K_f is often much larger than K_g[5]. We perform bus detection with microphone at the sampling rate of 8kHz. By using the Goertzel algorithm instead of FFT, the power consumption of the data collection app is reduced by 60mW.

We use Monsoon power monitor to measure the power consumption of two types of mobile phones (HTC Sensation and Nexus One) under different sensor settings and summarize the results in Table III. For each setting, we record the consumed energy over a period of 10 mins. The mobile phone screen is switched off during the measurement. The relative standard deviation is also shown in the parentheses. We measure the power consumption when no sensors are activated as a baseline case. The power consumption of sampling cellular signals is negligible for smartphones. We then measure the power consumption of GPS tracking at a sampling rate of 0.05Hz, which is already considered very low for vehicle tracking [22]. The average power consumption as high as 304mW for HTC and 333mW for Nexus One. The overall power consumption of our data collection app is 182mW for HTC and 196mW for Nexus One but it can be as high as ∼450mW if we use GPS instead of cellular data to track the trips of bus riders.

TABLE III
POWER CONSUMPTION COMPARISON (IN MW).

Sensor settings	HTC Sensation	Nexus One
No sensors	71(6)	84(5)
Cellular 1Hz	72(6)	85(8)
GPS	304(32)	333(41)
Cellular+Mic(Goertzel)	182(20)	196(22)
GPS+Mic(Goertzel)	447(45)	443(57)

V. RELATED WORK

Traffic estimation. In transportation domain, some operational systems have been developed to measure the traffic conditions using AVL system [10], [11], [18]. Chakroborty *et al.* [10] study the possibility of using transit vehicles as probes to predict automobile travel time. Pu *et al.* [18] propose to use bus travel information to infer general vehicle traffic conditions. Coifman *et al.* [11] use the transit fleet AVL data to find all trips that use any portion of a prespecified freeway segment. Researchers from computer science domain leverage various location references to build up traffic monitoring systems. VTrack [22] explores the GPS and WiFi references using COTS mobile phones to tag the traffic estimations. Aslam *et al.* [8] conduct a case study demonstrating that it is possible to accurately infer traffic volume through GPS data traces from a taxi fleet. Some works rely on heavy deployment of static infrastructures providing very limited spots of observations. For example, Kyun Queue [19] is a sensor network system for real time traffic queue monitoring with static sensor nodes deployed on roadsides. Our work primarily differs from existing works. We rely on the bus network to estimate the traffic conditions but fundamentally decompose the "probing" tasks from the running buses. We encourage participatory efforts from bus riders and derive the traffic map without cooperation of any transit agencies.

Tracking and localization. Many approaches for vehicle tracking and localization have been developed recently [14], [20], [21], [22]. Jiang *et al.* [14] quantify the relative proximity among physical objects and humans using "proximity zone", and compare the proximity zones created by various wireless signals. Thiagarajan *et al.* [20] present a crowd-sourced alternative to conventional transit tracking system using GPS, WiFi and accelerometer sensors on smartphones. CTrack [21] presents energy-efficient trajectory mapping using celltower fingerprints and utilize various sensors on mobile phones to improve the mapping accuracy. This work primarily differs from those works in its goal of urban traffic mapping. Our approach does not necessarily rely on tracking or locating individual bus routes. Accurate bus stop identification and mapping suffices to project the traffic estimation down to the

map. The system scalability and deployment overhead is a key design consideration under the crowdsourcing framework.

VI. Conclusions and future work

This paper presents the design and evaluation of a participatory urban traffic monitoring system. We leverage public bus networks to estimate the traffic conditions, and explore bus route constraints and bus stop references to generate the traffic map. Our system relies on participatory efforts from bus riders and utilizes lightweight sensing resources from COTS mobile phones. We comprehensively evaluate the system with a prototype system for 2 months in total in Singapore. The implementation and experiment results demonstrate our system effectively monitors the traffic conditions with amiable overhead. Due to its low cost and independence on any third-party cooperation, our system can be easily adopted to other urban areas with slight modifications.

Future work includes deriving the overall traffic of a region from the bus covered road segments. There have been some existing models in transportation domain [9], [13], which can be applied with our data feed. How to encourage bus riders' participation for consistent and good performance is important. At the initial stage, we may encourage the bus drivers to install our app to bootstrap the system. Meanwhile we are putting up the app on Google Play for possible experimental studies in other areas and expect more user involvement.

References

[1] Bus transport in Singapore. http://en.wikipedia.org/wiki/Bus_transport_in_Singapore.

[2] Bus Transport for London. http://www.tfl.gov.uk/.

[3] New York city bus system. http://www.ny.com/transportation/buses/.

[4] Waze traffic and navigation. http://www.waze.com/.

[5] Goertzel algorithm. http://en.wikipedia.org/wiki/Goertzel_algorithm.

[6] SBS Transit in Singapore. http://www.sbstransit.com.sg/.

[7] Land Transport Authority in Singapore. http://www.lta.gov.sg/content/ltaweb/en.html.

[8] J. Aslam, S. Lim, X. Pan, and D. Rus. City-scale traffic estimation from a roving sensor network. In *ACM SenSys*, pages 141–154, 2012.

[9] A. Bejan, R. Gibbens, D. Evans, A. Beresford, J. Bacon, and A. Friday. Statistical modelling and analysis of sparse bus probe data in urban areas. In *IEEE ITSC*, pages 1256 –1263, 2010.

[10] P. Chakroborty and S. Kikuchi. Using bus travel time data to estimate travel times on urban corridors. *Transportation Research Record*, 1870:18–25, 2004.

[11] B. Coifman and S. Kim. Using transit vehicles to measure freeway traffic conditions. In *AATT*, pages 90–101, 2006.

[12] W. Hedgecock, M. Maroti, J. Sallai, P. Volgyesi, and A. Ledeczi. High-accuracy differential tracking of low-cost gps receivers. In *ACM MobiSys*, pages 221–234, 2013.

[13] R. Herring, A. Hofleitner, and A. Abbeel, P.and Bayen. Estimating arterial traffic conditions using sparse probe data. In *IEEE ITSC*, pages 929 –936, 2010.

[14] X. Jiang, C.-J. M. Liang, K. Chen, B. Zhang, J. Hsu, J. Liu, B. Cao, and F. Zhao. Design and evaluation of a wireless magnetic-based proximity detection platform for indoor applications. In *ACM/IEEE IPSN*, pages 221–232, 2012.

[15] F. Li, Y. Yu, H. Lin, and W. Min. Public bus arrival time prediction based on traffic information management system. In *IEEE SOLI*, pages 336–341, 2011.

[16] K. Lin, A. Kansal, D. Lymberopoulos, and F. Zhao. Energy-accuracy trade-off for continuous mobile device location. In *ACM MobiSys*, pages 285–298, 2010.

[17] D. Maier and A. Kleiner. Improved gps sensor model for mobile robots in urban terrain. In *IEEE Robotics and Automation (ICRA)*, pages 4385–4390, 2010.

[18] W. Pu, J. Lin, and L. Long. Real-time estimation of urban street segment travel time using buses as speed probes. *Transportation Research Record*, 2129:81–89, 2009.

[19] R. Sen, A. Maurya, B. Raman, R. Mehta, R. Kalyanaraman, N. Vankadhara, S. Roy, and P. Sharma. Kyun queue: A sensor network system to monitor road traffic queues. In *ACM SenSys*, pages 127–140, 2012.

[20] A. Thiagarajan, J. Biagioni, T. Gerlich, and J. Eriksson. Cooperative transit tracking using smart-phones. In *ACM SenSys*, pages 85–98, 2010.

[21] A. Thiagarajan, L. Ravindranath, H. Balakrishnan, S. Madden, and L. Girod. Accurate, low-energy trajectory mapping for mobile devices. In *USENIX NSDI*, 2011.

[22] A. Thiagarajan, L. Ravindranath, K. LaCurts, S. Madden, H. Balakrishnan, S. Toledo, and J. Eriksson. Vtrack: accurate, energy-aware road traffic delay estimation using mobile phones. In *ACM SenSys*, pages 85–98, 2009.

[23] M. S. Waterman and T. F. Smith. Identification of common molecular subsequences. *Journal of Molecular Biology*, 147:195–197, 1981.

[24] J. Yang, A. Varshavsky, H. Liu, Y. Chen, and M. Gruteser. Accuracy characterization of cell tower localization. In *ACM Ubicomp*, pages 223–226, 2010.

[25] J. Yoon, B. Noble, and M. Liu. Surface street traffic estimation. In *ACM MobiSys*, pages 220–232, 2007.

[26] J. Zheng, Y. Wang, and N. L. Nihan. Tracking vehicles with gps: Is it a feasible solution? In *the 13th Annual Meeting of ITS*, 2002.

[27] P. Zhou, Y. Zheng, and M. Li. How long to wait?: predicting bus arrival time with mobile phone based participatory sensing. In *ACM MobiSys*, pages 379–392, 2012.