RideSense: Towards Ticketless Transportation

Rufeng Meng[†], David Wolfgang Grömling[‡§], Romit Roy Choudhury[‡], Srihari Nelakuditi[†]
[†]University of South Carolina [‡]University of Illinois at Urbana-Champaign [§]Technische Universität Darmstadt
mengr@email.sc.edu david_wolfgang.groemling@stud.tu-darmstadt.de croy@illinois.edu srihari@cse.sc.edu

Abstract—**Imagine a transportation system in which passengers simply get on/off without any explicit ticketing operation. Yet, the system tracks usage and charges passengers. Such a system will not only be more convenient and efficient, but also be more conducive for analytics, than existing systems. Towards that goal, we exploit the opportunity that people are carrying sensor-equipped smart devices, and their motion trajectories/patterns and experienced environment can be measured continuously. Assuming that vehicles are also equipped with such sensors (perhaps fixed devices or smart devices carried by drivers), the vehicles' motion and the experienced environment characteristics can also be recorded and uploaded to cloud. Under these assumptions, we hypothesize that the motion/environment sensed by a passenger's smart device correlates strongly with that of the vehicle she is traveling in and is distinct from that of other vehicles and/or other traces of the same vehicle. In this paper, we expand on this intuition and develop a system, called *RideSense*, that matches a passenger's sensor trace against the traces of buses in that area, to determine which bus, when she has taken and where she gets on/off. Our evaluation of *RideSense*, with 20+ hours of traces from 5 bus lines in our area, shows that it achieves an accuracy of 84∼98%, depending on the choice of sensors and their features, positions of the passengers' phones and the metrics of measurement. These results, while far from conclusive, offer confidence that ticketless public transportation may indeed be a possibility in smart cities of the future.**

I. Introduction

In current public transit system, there are two major ways for paying the fare. One is paying by cash. Passengers need to prepare and carry cash with them and pay the fare when they get on a bus. A more convenient way is IC-card based ticketing system. A passenger obtains a physical IC card from transit system operator and deposits fees in it. Later when she gets on a bus, she inserts or scans the IC card to pay. Besides, in recent years, we are witnessing the emergence of smart-device-based e-payment in public transit. For example, smartphone users in Japan use Felica [1]-embedded smartphones to take bus and subway. Apple Pay is now being used in London buses [2]. Both these are NFC-based solutions; passengers tap their phones (which contain NFC chip) on readers and go.

While the existing pricing, ticketing and billing process in public transportation systems generates significant revenue, it does not come free of financial and experiential hurdles. For instance, installing, upgrading, and maintaining the end-to-end accounting infrastructure (including ticketing kiosks, manned booths, ticket-checking gateways, card readers, etc.) require substantial investment. The core notion of buying tickets, at the right price, for the right destination, with the right monetary change, and just in time to catch the train, is often a source of frustration/anxiety, making the overall experience less seamless. Finally, ticketing often creates queues at purchase kiosks and at bus stations, because every passenger boarding the bus needs to pay for (or verify) her ticket. In sum, the process of gathering revenue in public transportation systems imposes operational burdens, both to the operators and the customers. Apart from the above drawbacks, in current bus systems, the operators at most know where each passenger gets on the bus, but they have no knowledge or record of where a passenger gets off, making it hard to perform analytics such as determining the occupancy of each bus at each road segment.

To eliminate the operational burdens from both customers and operators, and also provide an insight about the running of the traffic system, we propose a smart-device-based system, *RideSense*. Our core idea is simple and exploits the opportunity that people are carrying sensor-equipped smart devices everywhere, and their motion trajectories/patterns and the environment they experience can be measured continuously. Now, assuming that public vehicles can also be equipped with such sensors (perhaps installed explicitly or carried by the drivers), the vehicles' sensor data can also be recorded and uploaded to the cloud. Under these assumptions, we hypothesize that if Alice takes a bus, her sensor trace can be correlated against the sensor trace of the bus to precisely position her bus trip - which bus she takes, when she boards, where from and to she rides, etc. We envisage matching Alice's sensor data with the vehicle at fine granularities, including similar *pot hole jerks* that both Alice and the vehicle experienced, the *stops*, *turns* and the number of *lane changes*, precise times of *braking*, decreased *atmospheric pressure* due to increased altitude, etc. Fig. 1 illustrates *RideSense*.

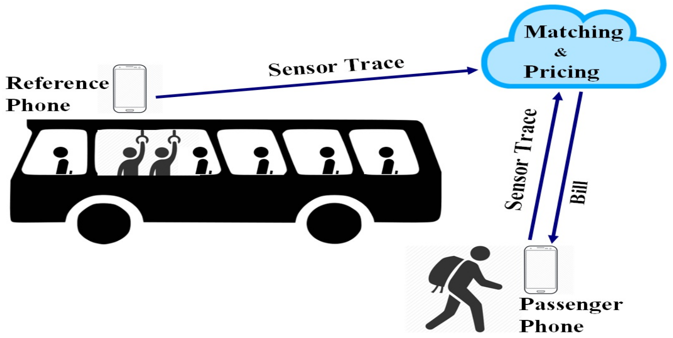


Fig. 1: An illustration of the *RideSense* system. Sensors on passenger's phone and on the bus collect the sensor data along the travel; later these sensor traces are uploaded to cloud for matching. The system could learn which bus line a passenger took, when she boarded, and also where the passenger has traveled, therefore the operator could bill the passenger.

In a *RideSense*-enabled bus, the reference device collects sensor data all the time when the bus is on duty. When the bus is off duty, the sensor traces are uploaded into cloud for matching. Passengers run a *RideSense* app in their smart devices and could take buses freely without any explicit interaction with any ticketing/billing facilities when they get on/off. *RideSense* utilizes the built-in sensors to record the motion and environment information (e.g. atmospheric pressure) along the passenger's trip. After the passenger gets off the bus, the app finds an opportunity (for example, when the user is at home with WiFi connected and the phone is not busy) to upload the recorded sensor trace into cloud.

The cloud matches a passenger's sensor trace with reference trace to find out: *i)* **Which** bus line the passenger has taken. *ii)* **Where** the passenger boarded and disembarked, i.e. the source and destination stations of a passenger's trip. *iii)* **When** the passenger took the bus. We use *Which/Where/When* to refer to these three aspects in the rest of this paper.

Designed correctly, *RideSense* can bill the users accurately afterwards. It depends on how well our intuition holds, i.e. one sensor trace from the reference device in bus, the other from the passenger's smart device, should correlate strongly. As a preliminary check, we collected motion data from three passengers' phones each on a different bus. We gathered motion data from each bus too and matched them against the passengers' motion data. Fig. 2 shows that the highest correlation values are along the diagonal indicating strong correlation between motion of a passenger and her bus. Though this is a toy experiment, it does affirm the core intuition.

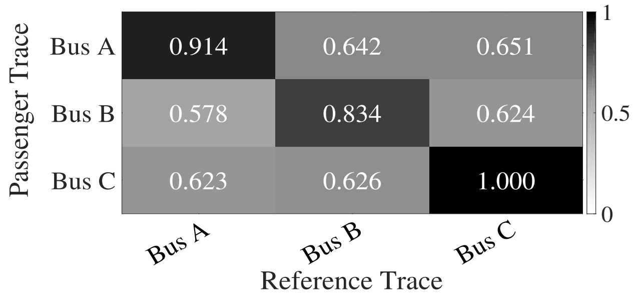


Fig. 2: The sensor trace collected by a passenger phone shows higher correlation with the corresponding bus trace.

Apart from billing the users correctly, perhaps more importantly, the overall operation of the public transit system can become far more seamless. For example, users can board buses from any of the doors and conventional ticket verification facilities are no longer needed, which reduces traffic backlogs at the stations, and administrators can attain entire programmability on pricing and billing. Perhaps other opportunities will arise, given the disruptions in the transportation industry with Uber-like services becoming popular. Finally, the data from the vehicles and users can be amenable to valuable analytics, offering insights into city planning, human mobility models, traffic control, pricing, etc. Of course, realizing such a vision will entail a variety of challenges, including sensor data processing, mechanisms to thwart cheating, location privacy for users, appropriate user interfaces, policies, etc. The tangible outcome of this research is expected to be a convincing, data-driven argument on the viability/practicality of this vision.

Currently, we target *RideSense* at bus system and assume that each bus will be fitted with a device (e.g. smartphone) for sensing motion and environment. We focus on implementing and evaluating the algorithms for matching passengers' sensor traces against the traces of the buses in that region, to study how accurately *RideSense* could tell the Which/Where/When information about passengers' bus trips. Considering that passengers' sensor traces are their tickets, we need to be concerned about the potential for cheating and tampering. Also, users' normal usage of the smart devices should not adversely affect the *RideSense* accuracy. While there are several such important issues that need to be tackled prior to deployment of *RideSense*, in this work, we assume a non-malicious passenger and conduct a study to assess the feasibility of *RideSense*.

Before we proceed with system design, it is important to clarify a question one may have: Why not match the GPS data from the user's phone with that of buses in that region? A major limitation of this GPS-based approach is that users are unlikely to always turn on the power-hungry GPS. Furthermore, if another vehicle also takes the same route between "source" and "destination" at that time, GPS-based approach can not correctly place the passenger on the right bus. The error of GPS readings in urban area (where tall buildings and/or tunnels exist) could also make the approach infeasible. On the other hand, *RideSense* can offer better overall performance using cheaper sensors (such as accelerometer, gyroscope, and barometer) on a smart device.

II. System Design

We now present the design of *RideSense* system. As mentioned earlier, *RideSense* app in passengers' smart devices collect readings from accelerometer, gyroscope and barometer. We refer to this sensor data as passenger trace. A smart device plugged in each bus records readings from GPS too in addition to the above set of sensors. We refer to this sensor data as reference bus trace. Considering that GPS coordinates of bus stations could be obtained in advance, we can extract the reference bus trace for each segment (see Fig. 3 for terminology). The objective of *RideSense* is to identify the sequence of bus segments whose sensor trace matches closely with the passenger sensor trace.

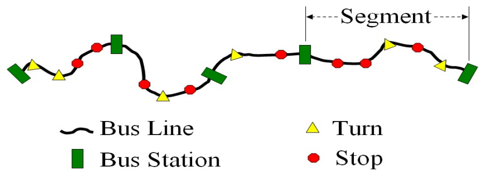


Fig. 3: A bus line is made up of segments, which are delimited by stations. Stations are the regulated places where the bus stops and allows passengers to get on/off. When a bus is moving on the road, it could experience many stops and turns.

Note that passenger trace may include data from other daily activities, as the user may have the app on even when she is not traveling on the bus. We need to make sure a passenger's smart device will only upload the sensor trace related to her travel in bus system. Although *RideSense* could require passengers to explicitly turn on the app only when they get on a bus and turn off immediately after they get off a bus, this might sacrifice some convenience and people might forget to turn on or off. To provide the best user experience, user only needs to turn on the app once and does nothing else. So the system needs to distinguish a user's bus traveling from all other daily activities. Lots of solutions [3]–[8] exist which could distinguish vehicle transportation from other daily activities through smartphone

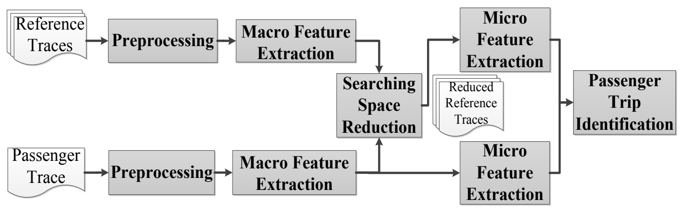


Fig. 4: *RideSense* extracts macro features from traces after preprocessing. Based on macro features, the searching space is reduced for each passenger trace. The system uses micro features to do trace matching to identify a passenger's trip.

sensors. Even for vehicle transportation, there are smartphone-based solutions [5] [7] to distinguish bus from car. We borrow the result from the existing approaches on distinguishing bus traveling from other activities.

Given a passenger trace, *RideSense* only knows it starts and ends at stations, but does not know the number of stations/segments in-between. To find the corresponding reference bus trace, *RideSense* needs to compare all possible combinations of consecutive segments from all bus lines. Considering the number of bus lines and passengers in a city, enormous number of reference and passenger traces are produced everyday, *RideSense* needs to search a huge space to match a given passenger trace. Fig. 4 shows the pipeline of the matching algorithm of *RideSense* in cloud. The system extracts macro features and micro features from both reference and passenger traces. Macro features are the basic properties of the trace, such as turn, stop information, which are used to filter out reference traces and reduce the searching space for each passenger trace. Micro features are the fine-grained characteristics of the motion and atmospheric pressure information of the travel, which are used for trace matching once the searching space is reduced. We elaborate on the pipeline steps below.

A. Data Preprocessing

The data collected with smartphone's built-in sensors, i.e. accelerometer, gyroscope and barometer, is very noisy. Before extracting features from these sensor data, *RideSense* goes through a preprocessing stage to clean the data, which contains two steps: smoothness and interpolation.

We run an exponential moving average on the raw sensor data to remove noise. Due to the low-quality of smartphone sensors and also the limitation of operation system, the collected sensor data does not have fixed intervals between samples. We use cubic spline interpolation [9] to interpolate the sample data and then resample to obtain sample data with needed sample rate.

B. Searching Space Reduction

Searching space reduction is carried out based on macro features. Macro features define the basic properties of a bus line and its segments. To reduce the searching space of reference traces for passenger traces, *RideSense* needs to extract macro features both from passenger traces and reference traces. By comparing the macro features of both sides, the reference traces which show irrelevant properties to the passenger traces could be eliminated from the candidate set.

In *RideSense*, following macro features are extracted and used: cellular network information, duration of travel, turn information, altitude information. We present the details of each macro feature, its usage and the way to extract in following paragraphs.

a) Cellular Network: Cellular network contains lots of cellular towers, which are distributed to provide good coverage. Each tower provides service to a limited area. A bus usually travels through many cells of the network, which are covered by different towers. Every cellular tower has a unique ID, which could be detected by the phone. By using the cellular network ID, *RideSense* could limit the reference traces for a passenger trace to a certain area, the searching space could be significantly reduced.

b) Duration of Travel: Both the reference traces and passenger traces are formed by segments which are delimited by stations. In the reference trace, by knowing the GPS coordinates of each bus station, we could easily figure out the duration at each station and also the duration between two consecutive stations. In passenger trace, *RideSense* does not have GPS information, so it doesn't know how many stations/segments a trace covers. Fortunately, the places where a passenger gets on and gets off are stations inherently. Furthermore, the segments and stations covered by a passenger trace are continuous. Therefore, the end-to-end duration of a passenger trace should correspond to the duration from one station to another station (including the inbetween segments and other stations), although *RideSense* does not know exactly which stations they are.

To find out the corresponding reference sub-trace for a passenger trace (a passenger usually only travels a part of a bus line), *RideSense* only needs to search sub-traces which start and terminate at stations. It does not need to search the reference traces in the middle of a segment. For a passenger trace, if the duration of a sub-trace in reference space is far beyond the duration of the passenger trace, it could not be the reference trace corresponding to the passenger trace.

From motion's perspective, every trace is formed by stop and move. So an end-to-end duration of a trip is composed of many stop-durations and move-durations. Different bus lines and even different traces of the same bus line exhibit different stops and moves. Thereby the proportion of stop and move durations within an end-to-end duration could also be macro features for a trace.

In reference trace, the stop and move could easily be identified by the speed measured by GPS. While in passenger trace, no GPS speed information is available. We developed a stop detector to identify the stop and move. After running the stop detector, we summarize the stop and move proportion of a passenger trace as macro features.

Stop Detection: The stop detection in *RideSense* is based on accelerometer and gyroscope readings. We run a sliding window with 1-second length and 0.5-second step size on the sensor data to extract features (as listed in Table. I) for stop detection. These features are selected experimentally. We trained a random forest which contains 50 decision trees for stop detection.

TABLE I: Features for Stop Detection

Sensor	Feature
Accelerometer	*Time Domain*: (Magnitude of Linear Acceleration) Mean, Median, Variance, Standard Deviation
	Frequency Domain: (Magnitude of Linear Acceleration) Maximal amplitude, Energy, Mean coefficient magnitude,Root Mean Square of bucket 0∼20 Hz
Gyroscope	*Time Domain*: (Magnitude of Gyroscope Readings) Mean, Median, Variance, Standard Deviation
	Frequency Domain: (Magnitude of Gyroscope Readings) Maximal amplitude, Energy, Mean coefficient magnitude, Root Mean Square of bucket 0∼20 Hz

c) Turn: Each bus line and its segments have different numbers of turns. A turn could be detected by continuously integrating the gyroscope readings over a window. We experimentally select 6 seconds as the window length for a turn. If it is a turn, the bus should be able to complete a continuous movement within the window and experience large enough turn degrees. Although gyroscope suffers from the problem of drifting [10], which usually leads to problems with a long duration. In a 6-second window, it is reliable to tell it is a turn or not from the integrated gyroscope readings.

Turns have different degrees and directions. To compare turn degree accurately, the passenger phone and reference phone must be aligned accurately. Reference phone could be fixed in a bus, but passenger phone has unlimited flexibility, which makes it less reliable in turn degree comparison.

When a bus makes a turn, it could have one of the two directions: left and right. A bus turns around its Z-axis (as shown in Fig. 5 (b)). The Z-axis of bus aligns with the Z-axis of the earth coordinate system (as shown in Fig. 5 (a)).

While the passenger and reference phones could be in any attitude, their coordinate systems do not align with the earth coordinate system (as shown in Fig. 5 (c)). In order to measure the turn around the earth's Z-axis, the smartphone's Z-axis and the Z-axis of bus (i.e. Z-axis in earth coordinate system) must be aligned.

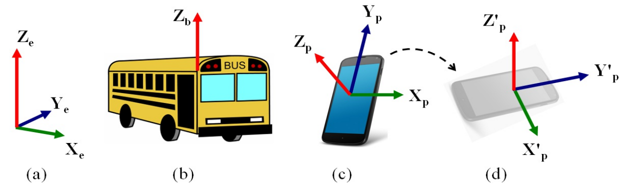


Fig. 5: (a) Earth coordinate system. Smartphone's coordinate system (c) usually does not align with bus coordinate system (a, b). To learn the turn direction, their Z axes must be aligned. *RideSense* utilizes the accelerometer readings when the phone is in static status to measure the relative attitude between smartphone and the earth coordinate system, and then a rotation is applied to align smartphone's Z-axis with the Z-axis in earth coordinate system (d).

To align the passenger phone's Z-axis with bus, *RideSense* needs to do a rotation on the smartphone's gyroscope readings, which could eliminate the relative difference between the two Z axes. The relative difference could be learned from accelerometer when the passenger phone is in a static status. If a phone is static, what the accelerometer measures is only the gravity. The gravity is distributed on its 3 axes, which tells the attitude of the phone in the earth coordinate system.

For reference trace, it is easy to find the static status from the GPS information, where the GPS speed is shown as 0. It is more difficult to identify the static status in passenger trace, which does not include GPS information. Although we could use stop detector to identify stops, which is unlikely 100% accurate in recognizing all stops. To reduce the impact of error propagation caused by stop detector, *RideSense* uses the accelerometer readings in a short period when the stop detector reports stop continuously for 3 times. Once 3 consecutive stops are reported, it is safer to conclude that the passenger phone is static, then *RideSense* learns the distribution of acceleration (i.e. gravity) on the 3 axes to obtain the attitude of the passenger phone.

From the attitude value, *RideSense* derives a rotation matrix, which is used to rotate the Z-axis in smartphone's sensor readings into the Z-axis of the earth coordinate system. Therefore, the system could measure the rotation around earth's Z-axis, i.e. how the bus is turning.

To tell the turn direction, *RideSense* only needs to look at the gyroscope readings on Z-axis. If the gyroscope reading on Z-axis is positive, the bus is turning left, otherwise it is turning right. We use a voting mechanism within the turn window to decide whether the gyroscope reading is positive or not.

In *RideSense*, we use the number of turns and turn directions as macro features.

d) Altitude: *RideSense* collects barometer readings which correspond to the altitude of the road. But barometer is easily affected by climate factors, such as temperature, humidity. It could only reflect relative relationship in altitude, e.g. place A is higher than place B. *RideSense* utilizes barometer to eliminate candidate reference traces for a given passenger trace. For instance, if the barometer reading at source station of a passenger trace is higher (with a threshold) than the destination station, the reference sub-trace which has opposite relationship in barometer readings between its source and destination stations could be ignored.

After reducing the searching space based on the macro features, the remaining reference traces are further compared with the passenger traces based on micro features, which is explained in next section.

C. Passenger Trip Identification

When a passenger takes a bus, the passenger phone should experience same or similar motion and environment as the reference device in the bus. *RideSense* characterizes the motion through micro features extracted from motion sensor data. Besides, the system also extracts micro features from the barometer readings which represent the atmospheric pressure along the trip. By comparing a sequence of micro features of passenger trace with a sequence of micro features extracted from reference trace, *RideSense* finds out the reference trace which corresponds to a passenger's travel.

When the reference phone and passenger phone are in static status, the trace data does not contain meaningful and comparable motion characteristics, which also wastes resource on computation. Therefore *RideSense* extracts micro features only from the sensor readings when the phone is experiencing movement.

The system extracts micro features from accelerometer, gyroscope and barometer both in time and frequency domain. In feature extraction, a 1-second sliding window is applied on sensor data, which moves every 0.5 seconds. The sequence of micro features of reference traces are delimited by stations. *RideSense* Z-normalizes the micro features and uses Dynamic Time Warping to do matching. In this system, normalized DTW distance is used to represent the similarity between traces. The reference trace which achieves minimal normalized DTW distance to the passenger trace is regarded as the corresponding reference trace.

To learn the effectiveness of the features, we collected experimental sensor traces along some randomly selected bus lines in our area, and then we carried out the study with each single feature on the experimental traces. Base on the experiment result and also taking the cost of computation into consideration, following (micro) features are finally used in *RideSense* for identifying detailed which/where/when information of passenger trips:

TABLE II: Micro Features for Passenger Trip Identification

Sensor	Feature (all based on the magnitude, time domain)
Motion Sensor	Median, Root Mean Square of Linear Acceleration; Log Energy of Gyroscope Readings
Barometer	Mean, Median, Variance, Standard Deviation, Range, Mean Crossing Rate, Mean Absolute Deviation, Skew, Root Mean Square, Signal Magnitude Area

III. Evaluation

We evaluate *RideSense* with the public transit in our university area to study how well the system could identify the bus line/shift (*which/when*) and source/destination stations (*where*), corresponding to a passenger's bus trip. Before presenting the detailed results, we summarize our findings below.

- *RideSense* can identify *which/when/where* about a passenger's travel with an overall accuracy of 85% using motion sensors and an accuracy of 91% based on barometer.
- It can determine the number of segments a passenger has travelled (fare for the bus-riding may depend on this) correctly in 96% and 99% instances with motion sensors and barometer respectively.
- The more segments a passenger travels, the higher the accuracy of *RideSense* in identifying her trip.
- Motion-sensor-based *RideSense* performs better when a passenger has the phone in his pocket than the phone in hand, whereas barometer-based *RideSense* is irrelevant to phone positions.

A. Data Collection

We recruited two volunteers for data collection. The volunteers traveled on 5 bus lines, which have 5 to 8 bus stations (correspondingly, 4 to 7 segments). For each bus line, they rode 3 times. The volunteers spent 20+ hours and collected more than 30G sensor data. The data collection was carried out in different time of different days, including rush hours and other time, and also experienced different weather conditions.

Each time, the volunteers use 3 phones. One acts as reference phone, the other two phones act as passenger phones. The phones are synchronized before the volunteers get on the bus, which allows us to get the ground truth for evaluation. After the volunteers get on the bus, the one carrying the reference phone sits close to the driver. The reference phone is fixed on the seat close to the driver. The two passenger phones are carried by the other volunteer who randomly selects available seat to sit. One of two passenger phones is in pants pocket, the other is in hand.

The reference phone collects sensor data from GPS, cellular network component (cellular network ID), accelerometer, gyroscope and barometer. Before the bus starts to move from the first station, the volunteer taking care of the reference phone marks a "start" on the reference phone; and then after the bus stops at the final station, this volunteer marks a "stop" on the reference phone. The start and stop markers in the reference trace tells the ground truth of the beginning and end of the bus trip. In the middle of the travel, the reference phone is not touched to avoid involving any motion from human.

The phone in the pants pocket of the passenger also collects the same sensor data as the reference phone. But the GPS information is only used for ground truth, it is not used in trace matching.

The phone in passenger's hand also collects same sensors as reference phone. Same as the pants pocket-phone, the GPS information here is only used as ground truth, which is not used in trace matching. The phone in hand is used for two purposes. First, the passenger marks the bus stations on this phone, which provides further ground truth about the stations of the bus line when combined with the reference trace, because we don't have detailed GPS location information about the bus stations in advance. Second, typing on the phone to mark the ground truth of stations will introduce the motion of the passenger, which allows us to mimic normal user operations on smartphone, although the operations are not intensive.

The collected data are processed and then used in evaluating the performance of *RideSense*.

B. Performance

To be a practical and useful system, *RideSense* should be able to achieve two goals: 1) It searches a candidate space as efficient as possible for a given passenger trace. 2) It identifies passenger trip with high accuracy. In this section, we verify our design and study the performance of *RideSense*.

In current implementation, we set relatively conservative conditions for macro-feature based searching space reduction. Fig. 6 shows the effectiveness of macro features in reducing searching space. On average, *RideSense* filters 91% of the candidate reference traces for each passenger trace before it goes into the micro-feature-based matching stage.

Next, we study whether *RideSense* could accurately identify: (i) *Which* bus line a passenger has taken; (ii) *When*, i.e., the shift of the bus she rides; (iii) *Where* she has traveled, i.e. the source and destination stations.

From the perspective of applications, *RideSense* could be used to bill a passenger. It could also help bus system operators analyze the traffic flows and the travel patterns of passengers.

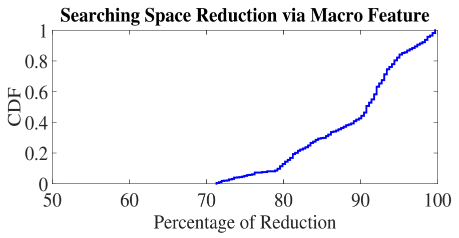


Fig. 6: Based on macro features, on average 91% candidate reference traces are filtered for each passenger trace.

For billing, knowing *which/where* is enough in most cases. Besides, we also want to find out how correctly the system will charge passengers and in what scenario, to what extent a passenger might be charged less or more than needed.

For traffic analysis, we consider the following fine-grained to coarse-grained measurements: i) *which/where/when*: It tells detailed travel information about each passenger, which helps the operators get fine-grained knowledge about the traffic system and travel pattern of each passenger. ii) *which/when*: It tells the bus line/shift a passenger has traveled, which could help the operator learn about the load on each bus line in a particular duration. iii) *which*: It enables the transit operator to be aware of the overall usage of its bus lines.

Therefore, *RideSense* is evaluated from four aspects, i.e. *which/where/when*, *which/where*, *which/when*, *which*. The accuracy of these measurements is defined by equations (1)∼(4).

$$Accuracy_{Which/Where/When} = \frac{\sum L_P = L_R \wedge BS_P = BS_R \wedge SS_P = SS_R \wedge DS_P = DS_R}{\# MatchingPairs} \quad (1)$$

$$Accuracy_{Which/Where} = \frac{\sum L_P = L_R \wedge SS_P = SS_R \wedge DS_P = DS_R}{\# MatchingPairs} \quad (2)$$

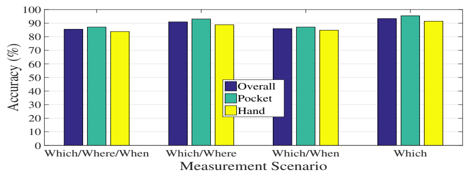
$$Accuracy_{Which/When} = \frac{\sum L_P = L_R \wedge BS_P = BS_R}{\# MatchingPairs} \quad (3)$$

$$Accuracy_{Which} = \frac{\sum L_P = L_R}{\# MatchingPairs} \quad (4)$$

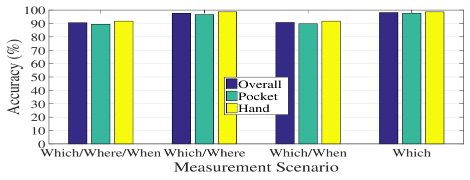
where L_P (L_R) is the id of bus line of passenger P (reference R); BS_P (BS_R) is the id of bus shift; SS_P (SS_R) and DS_P (DS_R) stand for source and destination segments.

The area we collected data is not completely flat, which exhibits barometer-friendly characteristics. Buses go through several slopes, which have rakes ranging from 5 to 30 degrees. Barometer is capturing the characteristics of the terrain of certain area, hence its performance may be less generalizable than motion sensors. Therefore, we evaluated *RideSense* based on micro features from motion sensors and barometer separately. The overall accuracy is shown in Fig. 7.

According to these results, when matching is based on motion sensors, the system achieves overall accuracy of 85% for *which/where/when*, 91% for *which/where*, 86% for *which/when* and *which* is identified with an overall accuracy to 93%. Barometer related results show that, in our area, if the *RideSense* uses barometer-based information, it could achieve much better performance than the situation with motion sensors. Its overall accuracy goes to 91% in the fine-grained measurement (i.e. *which/where/when*). In coarse-grained measurement, the accuracy is even higher, up to 98%.



(a) Motion Sensors

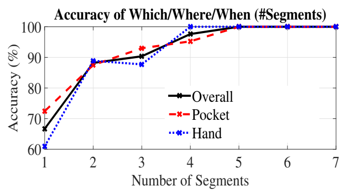


(b) Barometer

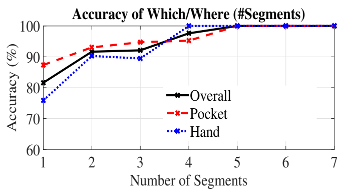
Fig. 7: Accuracy of (a) motion-sensor and (b) barometer based *RideSense* in identifying *which/where/when*, *which/where*, *which/when*, and *which*, with phones in hand or pocket.

When the evaluation is broken down into different positions of passenger phones, it is evident that with motion-sensor based *RideSense*, the position of the passenger phone affects the performance. In all the measurement, the passenger trace from the pocket phone could achieve higher correlation with the corresponding reference trace than the correlation between the trace from the hand phone and the reference phone. The reason behind this is that the motion sensor data collected by the phone in hand is polluted by the compensatory motion of hand, which weakens its correlation with the reference trace. On the contrary, if it is barometer-based, the positions of the passenger phone do not play a role and the accuracy between different phone positions does not show significant difference.

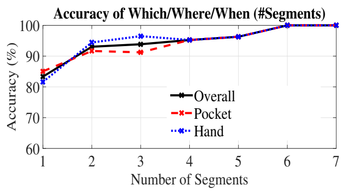
A passenger trip might cover different numbers of segments of each bus line. In the bus lines we collected, the numbers of segments range from 4 to 7. We derive sensor traces with all possible numbers of segments a passenger might travel, i.e. 1∼7 segments each trip. Then, we study the matching accuracy between passenger and reference traces based on different numbers of segments. The result is shown in Fig. 8. Due to the limitation of space, we only show the result for *which/where/when* and *which/where*. The accuracy of *which/when* and *which* also follow a similar trend.



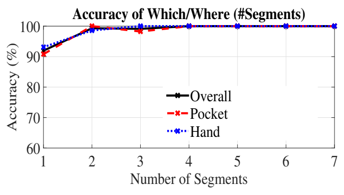
(a) Motion Sensors



(b) Motion Sensors

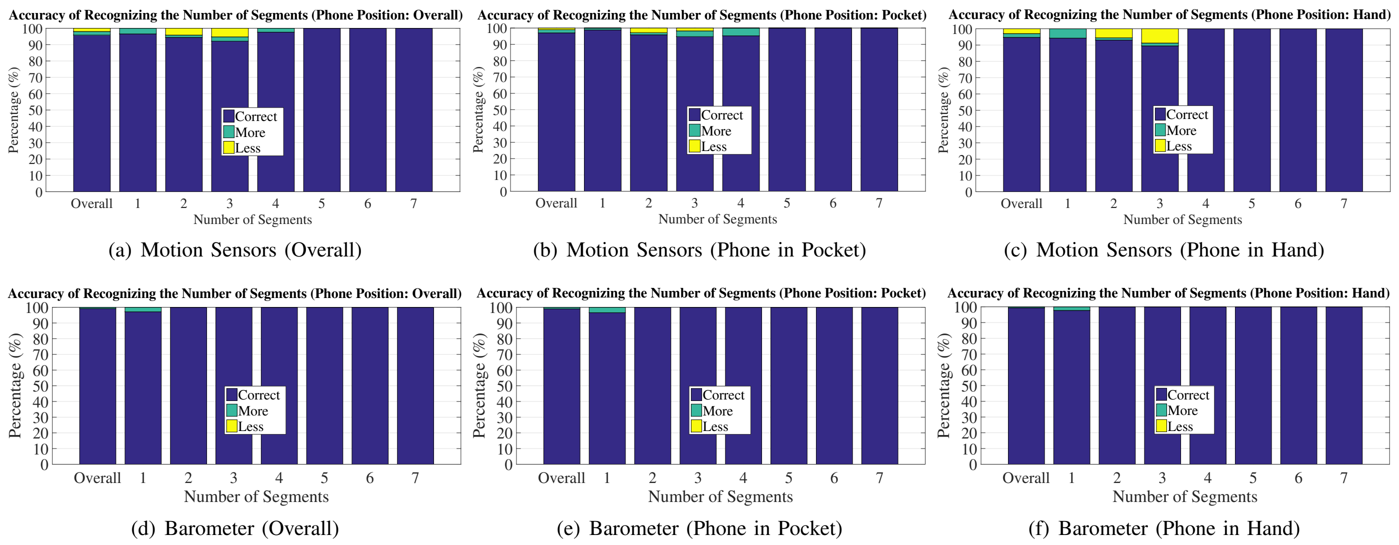


(c) Barometer



(d) Barometer

Fig. 8: The accuracy of motion-sensor-based *RideSense* in identifying which/where/when (a) and which/where (b) when a passenger travels different numbers of segments. (c)(d) The accuracy of *RideSense* when barometer is used.



(a) Motion Sensors (Overall) (b) Motion Sensors (Phone in Pocket) (c) Motion Sensors (Phone in Hand)

(d) Barometer (Overall) (e) Barometer (Phone in Pocket) (f) Barometer (Phone in Hand)

Fig. 9: (a∼c) The accuracy of motion-sensor-based *RideSense* in recognizing the numbers of segments a passenger has traveled, which decides whether the passenger will be charged correctly, or charged more, charged less. (a) shows the overall accuracy; (b) shows the accuracy when the passenger phone is in pocket; (c) shows the accuracy when the passenger phone is in hand. (d∼f) Performance of barometer-based *RideSense* in charging passengers.

Overall, with motion sensors or barometer, the accuracy increases along with the increasing numbers of segments. In other words, when a passenger travels more segments on a bus line, the system can identify her ride more reliably.

Consistent with the result shown in Fig. 7, in our area, barometer-based *RideSense* achieves higher accuracy than motion-sensor-based system. When motion sensors are used, the position of the passenger phone matters. The trace from the pocket phone could be used to recognize the numbers of segments a passenger traveled more accurately. When barometer is used, *RideSense* performance is not affected by the position of the phone or the motion of the passenger.

In many countries/cities, transportation fare is proportionate to the number of segments a passenger travels. After matching passenger trace with reference trace, if *RideSense* concludes a longer bus-riding than a passenger's actual trip, the passenger might be charged more than what she should pay. Conversely, if a shorter bus-riding is concluded for a passenger trace, the passenger might pay less than she should. Fig. 9 shows how accurately *RideSense* identifies the numbers of segments a passenger has traveled. When motion sensors are used, overall, a passenger will be charged correctly in 96% instances. In 2% cases, *RideSense* might charge a passenger more than she should pay, and in another 2% cases, the system charges passenger less. When a passenger takes a longer trip, more likely she will be charged correctly. Also, phone in pocket is more friendly to the billing system than the phone in hand. When barometer is used, the overall accuracy climbs to 99% and the performance is agnostic to the position of phone.

To summarize, our preliminary study shows that *RideSense*, while not yet accurate enough to be an alternative ticketless system, holds promise. By comparing the performance of motion-sensor-based matching and barometer-based matching in *RideSense*, we find that, in our area, by using barometer-based feature, the system could outperform motion-sensor-based method. We admit that the performance of barometer-based *RideSense* is correlated with the terrain in our area. For real deployment, we could combine motion sensor with barometer to build a hybrid *RideSense*, which utilizes the characteristics exhibited both in motion and terrain to achieve best performance.

IV. Limitations and Discussion

Needless to say, this paper is a small step towards the broader vision; substantial work remains as discussed here.

User Behavior: In our experiment, only minor user motion is introduced (i.e. the user is tapping on the phone in hand for marking the ground truth) in the sensing process of the passenger phones. In reality, situation might be much more complex. For instance, a passenger might be playing game with her smartphone during her bus trip. A malicious user might deliberately shake and move the phone to continuously and intensively distort the motion trace. These user behavior could pollute the sensor trace of the passenger's bus-riding, which thereby hinders the correlation with the reference trace. To cope with these situations, we are exploring factorization algorithms to separate vehicle motion from human motion, given that they have some statistically distinct properties. We leave this to future work.

Bandwidth and Energy Considerations: We have also assumed that the uploaded data is only from the segments during which the passenger is in the bus – in reality, this is non-trivial. A classifier that recognizes that a person has boarded a bus will need to run continuously, imposing an energy burden. Even if continuous sensing and classification can be solved efficiently, not all the data during a vehicular trip may need to be uploaded. Identifying the most discriminating data segments, and uploading them at opportune times, will reduce the upload bandwidth from millions of users. A more careful treatment is needed for a complete system.

Additional Opportunities: While challenges are many, opportunities exist too. Data from multiple passengers in the same bus should exhibit similarities that could be useful in improving the system's accuracy, and in thwarting misbehavior. The post-paid model possible with *RideSense* could also enable various pricing schemes and discounts. For instance, those that traveled in a crowded bus, or were delayed by accidents, could be compensated through a discount applied to their fare later, ultimately incentivizing public transportation. Moreover, in Uber-like business, by correlating the sensor traces from driver's phone and passenger's phone, service could be improved in case GPS signal is lost or corrupted.

In current implementation and evaluation, we fixed the reference phone on a seat close to the driver and tried passenger phones in pocket and hand. As a next step, we plan to try both the reference phone and passenger phone in different positions, and collect data with more bus lines. We will also carry out the study in different areas to see how the system will perform in different traffic situations and terrain.

V. Related Work

Smartphone sensors have been used in traffic/transportation related research to detect transportation mode, study driver's behavior, localize vehicles, and even identify travel information. Here we list several works which are close to our solution.

Trellis [11] is a WiFi-based solution for transit analytics. It utilizes the WiFi infrastructure in buses to detect the WiFi-enabled mobile device carried by passengers. This system could be used to identify popular routes, occupancy of buses, etc. Compared with our solution, with Trellis, WiFi in passengers' devices must be enabled. Due to the characteristics of wireless signal, the passengers in other parallel-moving vehicles could be erroneously counted as the passengers in current bus. While with *RideSense*, even two vehicles are moving in parallel, they still exhibit difference in motion (for example, the acceleration/deceleration in speeding up/down, time duration of stops, etc), which make the two vehicles potentially distinguishable.

VTrack [12] and CTrack [13] use smartphone to collect GPS, WiFi and GSM information along a user's trip to figure out the road segment and trajectory a user travels. By aggregating the travel information, these two systems learn about the traffic situation and estimate travel time to help on route planning.

Authors in [14] utilize bluetooth to scan passenger's phone to identify passenger's trip on a bus. It treats first detection of a phone as the source station and if unreachable for a certain time, it concludes the passenger has got off. This solution cannot reliably isolate its detection within a bus, people in a neighboring bus might also be detected. It also incurs a privacy problem, as anyone can detect the passengers.

The works [15], [16] and [17] also perform sensor-based trace matching, similar to our system. [15] uses GPS and barometer to collect altitude related information and matches a passenger's partial GPS/barometer trace with pre-constructed reference trace to find out the bus line of travel. [16] uses GPS, WiFi and accelerometer data to determine bus-riding and matches routes based on the shape of the road in horizontal plane. [17] recognizes bus line and estimates arrival time based on matching cell tower ID sequence. This work relies on audio processing to detect the audio signal of IC card reader in order to tell whether a user is in public transit system. Compared with these works, our approach uses only energy-efficient sensors and provides *which/when/where* information about a passenger's travel, enables detailed traffic analysis.

VI. Conclusion

We envision smart transportation of the future wherein purchasing tickets is not necessary. By exporting sensor data from passengers' smart devices and public vehicles, it should be possible to detect how a passenger utilizes public transportation, and bill her appropriately at the end of a day or a month. From an operational point of view, such a system could eliminate the entire ticketing and maintenance infrastructure (ticket dispensers/sellers, gate checks, card readers, etc.). From a user's perspective, she could experience a seamless hop-on/hop-off experience, without worrying about correct pricing, cash amount, losing the ticket, etc. While realizing this vision indeed poses logistical hurdles, *RideSense* is a first step towards initiating the process and asking the right questions. We believe there is adequate promise to engage into a serious research effort.

References

[1] "Felica," http://www.sony.net/Products/felica/business/data/FeliCa_E.pdf.

[2] "Apple pay - transport for london," https://tfl.gov.uk/fares-and-payments/contactless/other-methods-of-contactless-payment/apple-pay.

[3] S. Wang, C. Chen, and J. Ma, "Accelerometer based transportation mode recognition on mobile phones." in *APWCS*, 2010.

[4] L. Bedogni, M. Di Felice, and L. Bononi, "By train or by car? detecting the user's motion type through smartphone sensors data," in *Wireless Days (WD)*. IFIP, 2012.

[5] S. Hemminki, P. Nurmi, and S. Tarkoma, "Accelerometer-based transportation mode detection on smartphones," in *Sensys*, 2013.

[6] K. Sankaran, M. Zhu, X. F. Guo, A. L. Ananda, M. C. Chan, and L.-S. Peh, "Using mobile phone barometer for low-power transportation context detection," in *Sensys*, 2014.

[7] D. Shin, D. Aliaga, B. Tunçer, S. M. Arisona, S. Kim, D. Zünd, and G. Schmitt, "Urban sensing: Using smartphones for transportation mode classification," *Computers, Environment and Urban Systems*, vol. 53, 2015.

[8] H. Xia, Y. Qiao, J. Jian, and Y. Chang, "Using smart phone sensors to detect transportation modes," *Sensors*, vol. 14, 2014.

[9] C. De Boor, "A practical guide to splines," *Mathematics of Computation*, 1978.

[10] O. J. Woodman, "An introduction to inertial navigation," *University of Cambridge, Computer Laboratory, Tech. Rep. UCAMCL-TR-696*, 2007.

[11] L. Kang, B. Qi, and S. Banerjee, "A wireless-based approach for transit analytics," in *HotMobile*, 2016.

[12] 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 *Sensys*, 2009.

[13] A. Thiagarajan, L. Ravindranath, H. Balakrishnan, S. Madden, L. Girod *et al.*, "Accurate, low-energy trajectory mapping for mobile devices." in *NSDI*, 2011.

[14] V. Kostakos, T. Camacho, and C. Mantero, "Towards proximity-based passenger sensing on public transport buses," *Personal and ubiquitous computing*, vol. 17, 2013.

[15] S. Chatterjee, J. K. Nurminen, and M. Siekkinen, "Low cost positioning by matching altitude readings with crowd-sourced route data," in *MoMM*, 2012.

[16] A. Thiagarajan, J. Biagioni, T. Gerlich, and J. Eriksson, "Cooperative transit tracking using smart-phones," in *Sensys*, 2010.

[17] P. Zhou, Y. Zheng, and M. Li, "How long to wait?: predicting bus arrival time with mobile phone based participatory sensing," in *MobiSys*, 2012.