MapCorrect: Automatic Correction and Validation of Road Maps Using Public Sensing

Patrick Baier, Harald Weinschrott, Frank Dürr, Kurt Rothermel
Institute of Parallel and Distributed Systems
Universität Stuttgart
70569 Stuttgart, Germany
Email: {baier, weinschrott, duerr, rothermel}@ipvs.uni-stuttgart.de

***Abstract*—With the increasing proliferation of small and cheap GPS receivers, a new way of generating road maps could be witnessed over the last few years. Participatory mapping approaches like OpenStreetMap introduced a way to generate road maps collaboratively from scratch. Moreover, automatic mapping algorithms were proposed, which automatically infer road maps from a set of given GPS traces. Nevertheless, one of the main problems of these maps is their unknown quality in terms of accuracy, which makes them unreliable and, therefore, not applicable for the use in critical scenarios.**

To address this issue, we propose MapCorrect: An automatic map correction and validation system. MapCorrect automatically collects GPS traces from people's mobile devices to correct a given road map and validate it by identifying those parts of the map that are accurately mapped with respect to some user provided quality requirements. Since fixing a GPS position is a battery draining operation, the collection of GPS data raises concerns about the energy consumption of the participating mobile devices. We tackle this issue by introducing an optimized sensing mechanism that gives the mobile devices notifications indicating those parts of the map that are considered as sufficiently mapped and, therefore, require no further GPS data for their validation. Furthermore, we show by simulation that using this approach up to 50% of the mobile phones' energy can be saved while not impairing the effectiveness of the map correction and validation process at all.

***Index Terms*—Wireless sensor networks, Mobile Computing, Energy-aware systems**

I. Introduction

Within the last few years mobile phones evolved from special purpose devices to versatile computing platforms. Hence, most of them nowadays combine the ability of conducting phone calls with the possibility to transfer data packets and to make use of the built-in sensors like a GPS sensor or a camera. By utilizing these built-in sensors, sensor data can be collected without the requisite for a prior sensor deployment, as they already reside on mobile phones that are carried by people. This approach was described in various papers entitling it with different names, e.g. "Public Urban Sensing" [1], "Mobile Phone Sensing" [2] or "Mobiscopes" [3]. We refer to this new form of sensor data collecting in the following as *Public Sensing*. Due to the inherent mobility of the sensors, new scientific challenges arise in Public Sensing that go beyond the research field of traditional sensor networks.

One particular scenario, in which sensor data from different spatial locations can be utilized, is the generation of road maps. Manual approaches like OpenStreetMap [4] show the feasibility of generating useful maps out of GPS traces that were gathered and manually integrated into a global map by volunteers. In contrast to that approach, there also exists work dealing with the automatic generation of road maps out of raw GPS traces (e.g. [5], [6]). The drawback of both approaches is that the quality of the resulting road map can hardly be assessed. For instance, the accuracy of the road points of the map heavily depends on the accuracy of the GPS sensors that were used for gathering the required traces. Furthermore, the geometry of a road map might change over time due to road construction works. Therefore, to make sure that a given road map is accurate and up-to-date, some form of map validation and correction has to be done. Given the fact that most GPS sensors indicate an estimation about the accuracy of a position fix, a set of recently recorded GPS traces can be utilized to validate and correct the geometry of an existing road map by comparing the coordinates and accuracy values of these traces with the given map.

While manually collecting the needed GPS traces would cause a huge effort and thus cannot be considered as a suitable solution, an approach that is based on the aforementioned Public Sensing works without any manual effort by automatically utilizing GPS sensors that are already deployed on common mobile phones. Since reading sensor values and uploading the sensed data are battery draining operations the user will only accept the sensing system if it does not significantly decrease the battery life of his device. Having these constraints in mind, we present MapCorrect: A map correction and validation system that uses a Public Sensing approach to extract and process GPS traces from common mobile phones while minimizing the battery consumption of the involved devices.

Given an initial road map, MapCorrect matches GPS traces that were automatically recorded by the participating mobile phones to this map. Using the accuracy and position information provided with each GPS fix, MapCorrect refines the geometry of the road map by using map interpolation methods and marks a road segment as validated if it is sufficiently verified by GPS traces. Addressing the efficiency matter, MapCorrect sends the set of validated road segments to the mobile devices, which can turn off their GPS devices as long as they are located on a validated road segment.

Published in Proceedings of the 36th IEEE Conference on Local Computer Networks (LCN 2011), pp.58-66.
Bonn, Germany. October 2011.
© IEEE 2011
http://dx.doi.org/10.1109/LCN.2011.6115522

In detail, the contributions of our work are as follows:

1) We introduce quality metrics that quantify the accuracy of a given road map without requiring knowledge about the ground truth data.
2) We introduce MapCorrect, a system that automatically gathers GPS traces from mobile devices to refine the geometry of a given road map and to mark those parts of the map as validated that are accurately mapped with regard to some user given accuracy thresholds.
3) We present an optimized sensing approach that reduces the number of GPS fixes the mobile devices have to perform for collecting the necessary GPS traces.
4) We show by simulation that using the optimized sensing approach MapCorrect saves up to 50% of the mobile devices' energy compared to a naive approach, in which every mobile device continuously senses its position. Furthermore, we show that the speed and the quality of the map correction and validation process do not suffer from the optimized sensing approach.

The rest of the paper is organized as follows. In Section II we first give an overview of related work concerning map manipulation algorithms and efficient mobile sensor data acquisition. In Section III we introduce our system model, before we present in Section VI the basic concepts of MapCorrect dealing with the general process of map validation and correction. In Section V we address the efficiency of our approach by introducing an optimized sensing concept. In Section VI we show the simulation results we obtained by evaluating the system, before Section VII concludes this work.

II. Related Work

Since our work addresses problems that arise from two different research areas, we discuss them in the following separately. First, we look at algorithms that perform some kind of *automatic map manipulation*. Second, we focus on the problem of *efficient data acquisition from mobile sensors*, which is not covered by the aforementioned mapping algorithms.

A. Automatic Map Manipulation

One of the first approaches that automatically constructs a road map from scratch was proposed by Morris et al [7]. In the first step of their approach, they construct an initial graph from a set of given GPS traces. This is done by splitting the traces at their intersection points into segments, which constitute the edges of that graph. To merge edges that are duplicate representations of the same physical road, a sequence of graphical reduction steps is executed on that graph. As a result, a distinct road structure is obtained in the end. Until now, several other map generation algorithms were proposed, all using different theoretical approaches to merge a set of ambiguous GPS traces into a distinct representation of a road map (e.g. [5], [6], [8], [9]). Bruntrup et al. [6], for instance, use an incremental map generation approach, which starts with a single GPS trace and consecutively merges similar traces with the existing one to infer the road geometry.

Besides these general mapping algorithms, other work focuses on more specific challenges like inferring intersections from a set of overlapping GPS traces [10] or detecting obstacles by analyzing the position and communication history of mobile nodes [11]. In contrast to the aforementioned algorithms, which have the goal to infer the road structure, there are some approaches that assume the road structure as given and only focus on surveying its current condition. For instance, the Pothole Patrol system [12] tries to detect road anomalies by analyzing acceleration data of cars. The Nericell system [13] follows a similar approach but, in addition, interprets data from audio sensors to estimate traffic conditions. The work that comes closest to our approach in terms of map manipulation is the one presented by Rogers et al. [14]. They use GPS traces to refine an already existing road map and try to predict vehicle lanes for a given road.

Our approach differs from those mentioned in this section due to the fact that none of them provide explicit quality metrics that can be used for checking the accuracy of a given road map with respect to some user given quality requirements. Furthermore, all of these approaches rely on GPS traces that were collected by mobile devices, but none of them considers the process of efficiently acquiring this data. In the next section we have a closer look on existing work dealing with that issue.

B. Efficient Mobile Sensor Data Acquisition

Originating from the field of traditional stationary sensor networks, the research field of effective data acquisition from common mobile devices (Public Sensing) has gained increasing momentum over the last few years. While most of the initial papers described the major visions and challenges of this new research field (e.g. [1], [3], [15]), gradually more concrete solutions arise that deal with the efficient collection of sensor data from mobile devices. The scope of these solution ranges from the cooperative distribution of sensing tasks (e.g. [16]) to efficient data acquisition algorithms, on which we will focus in the following.

Mobile sensors are deployed on mobile devices, which come with limited resources. Hence, one of the main objectives of the efficient data acquisition from mobile sensors is to reduce the mobile devices' overall energy consumption. For instance, Eisenman et. al. [17] reduce the energy consumption of mobile nodes by adapting the communication range of a node to an optimal size considering parameters like time constraints. Another way of reducing the energy consumption is the proactive scheduling of sensor readings, in order to avoid redundant data acquisition from multiple sensors. Most of the existing work on this topic presents scheduling algorithms for stationary sensor networks, which do not consider sensor movement (e.g. [18], [19]). In our previous work ([20]–[22]), we presented mobile sensor scheduling algorithms that tackle the mobility issue. By using position and movement information of the mobile devices as input to the sensor scheduling, we showed that the number of devices that redundantly sense a predefined point of interest can be significantly reduced.

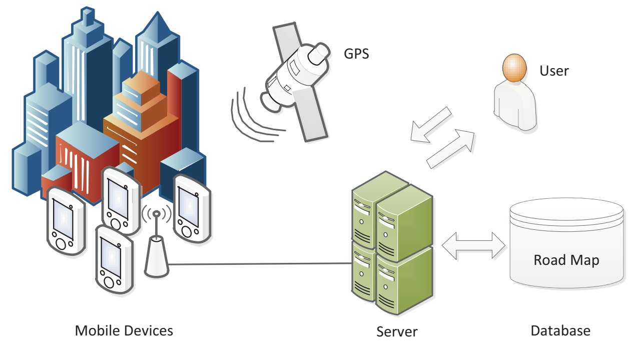

Fig. 1. System Model

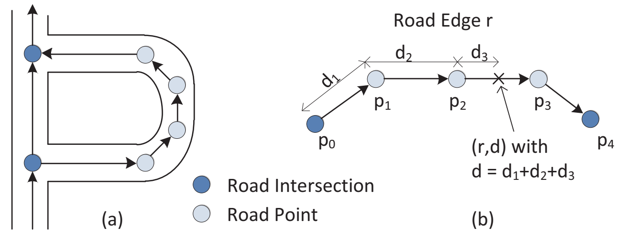

Fig. 2. Example Road Graph

This work differs from the work mentioned before since it is the first work that tackles the problem of verifying a road map using Public Sensing. This addresses new problems that were not considered in the approaches mentioned before. Since the scheduling of the GPS sensors depends on the dynamically changing set of validated road segments, we have to face the question of a dynamic sensor scheduling. Moreover, we have to make sure that the information that is necessary for the dynamic scheduling is communicated in an efficient manner.

III. System Model

Next, we present our system model and assumptions. The main parts of our system consist of a central server and a set of mobile devices. A mobile device comes with a GPS sensor and communicates with the central server via a cellular mobile network like UMTS (see Figure 1). We assume that the server knows the set of mobile devices that are available for the data collection due to a registration mechanism like the one proposed in [23]. Mobile devices are carried by people, thus the server has no influence on their movement direction or speed. The movement of the devices follows an underlying road structure. While the central server comes with unlimited processing power and power supply, the energy of a mobile devices is limited by the device's battery. Therefore, battery draining operations like GPS position fixes and mobile communication are critical and have to be minimized.

A mobile device obtains with each GPS fix location coordinates and an error value that indicates the accuracy of these coordinates. As proposed in [24], this error value is expressed by the standard deviation σ of the obtained position. By taking into account the relative geometry of the satellites used for the GPS fix, the Horizontal Dilution of Precision (HDOP) can be derived. With the help of this value and the User Equivalent Range Error (σ_{UERE}), σ can be calculated by [25]:

$$\sigma = HDOP \cdot \sigma_{UERE}$$

Hence, a smaller value for σ implies a higher probability that the true position of the mobile device is close to the coordinates obtained by the GPS fix. We assume that all GPS sensors can derive the HDOP and the σ_{UERE} value and, therefore, can calculate the standard deviation σ of a GPS position. We refer to this value as $p_{gps}.\sigma$.

The road map that is to be verified is stored in a database and can be accessed by the server. We assume that the mobile devices also know this map, otherwise the server could just distribute it to the mobile devices. The map is given as a road graph that consists of edges and vertices. The vertices represent road intersections and terminal points of a road. They are interconnected by polygonal road edges, which represent the roads. Each road edge consists of a set of connected road points, which reflect the geometry of the road (see Figure 2a). A road point p has an assigned error value $p.e$, which specifies the accuracy of the road point in analogy to the error value of a position fix $p_{gps}.\sigma$. Therefore, the position of a road point with a low error value can be considered as relative accurate with respect to the real shape of the road. By assigning each road edge a distinct number and defining its first road point as its starting point, we can refer to a particular point on a road by (r, d), in which r indicates the road edge and d the distance from its starting point to the location of the point following the road's course (see Figure 2b). Additionally, we define the subsection of a road edge starting at road point p_i and ending at road point p_j with $i < j$ as the road segment $\langle p_i, p_j\rangle = (p_i, p_{i+1}, ..., p_{j-1}, p_j)$.

IV. Map Validation Mechanism

To explain the basic concepts of MapCorrect, we present in this section the basic idea of map correction and validation. First, we introduce the quality metrics that are needed for identifying accurately mapped road segments before we show how the system is initialized with a given map validation query. Subsequently, we show how GPS traces are processed to refine and validate a road map.

A. Quality Metrics

Besides correcting the geometry of given road map, the goal of MapCorrect is to provide the user with a set of road segments that can be considered validated in terms of accuracy. Therefore, we have to define a way to measure the quality of a given road segment and to decide in which case we consider a road segment as validated. Since these estimations are highly subjective, we subsequently introduce quality requirements,

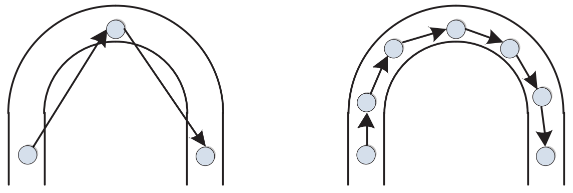

Fig. 3. Approximation of a curve

which are specified by the user and included as part of the map validation query.

The output of the sensing process of the mobile devices are GPS traces, which are processed by the central server. These GPS traces are used to enrich the existing road graph with additional road points, whereas the error value $p.e$ of a new road point is derived from the positioning error $p_{gps}.\sigma$. More details of the individual processing steps are presented in the next section. By evaluating the spatial distribution and the accuracy of the road points of a road segment, we decide if that segment can be considered as validated. Hence, we introduce two user-given parameters, which define the maximum distance d_{max} and the maximum error e_{max} of a set of successive road points.

Consider a road segment $\langle p_i, p_j \rangle$ of the road graph. To mark this road segment as validated it must fulfill the two following criteria, in which $d(p_x, p_y)$ indicates the length of the edge between the road points p_x and p_y:

$$\forall k \in \{i, ..., j-1\} : d(p_k, p_{k+1}) \leq d_{max} \quad (1)$$

$$\forall k \in \{i, ..., j\} : p_k.e \leq e_{max} \quad (2)$$

Both requirements aim at the accuracy of the shape of the road segment. Equation (1) ensures that the maximum distance between each pair of successive road points is not larger than the given threshold value d_{max}. Obviously, complex shapes like curves can only be approximated accurately, if the road points are close enough (see Figure 3). Equation (2) ensures that the error value of all road points in $\langle p_i, p_j \rangle$ does not exceed the maximum error threshold e_{max}. Thus, it makes sure that the road points are sufficiently mapped in terms of position accuracy.

B. Initialization

The systems starts with a map validation query from a user. This query includes the afore defined quality metrics and the map that should be validated. The bounding box of this map defines the area-of-interest from which the system gathers GPS traces for map validation. For the following processing steps, we initialize all road points of the road graph with an initial error value. If no accuracy information for the initial road map is known, we initialize its road points with a worst-case error value. Otherwise, the system can start with already known accuracy values for the road points.

To start with, the server sends an initialization message to all known mobile devices that are located in the area-of-interest. After having received such a message, all participating mobile devices start continuously recording GPS positions every t_{gps} seconds. Each time a mobile device has recorded a predefined number of positions it sends the recorded GPS trace to the server.

C. Processing GPS Traces

Having received a GPS trace from a mobile device, the server starts comparing it with the road graph. In the following we assume that every GPS trace can be matched with a corresponding road of the road graph. If that is not the case the trace could either have been recorded on a road that is not part of the given map or the mobile device is currently not located on a road. Since we focus on map validation, we filter out GPS traces that cannot be matched to a road segment. Nevertheless, one of the map generation algorithm described in Section II could be applied to these traces to enrich the road graph with additional roads.

Given a GPS trace, we use one of the existing road fusion algorithms (e.g. [26], [7]) to merge this trace with an existing road. Therefore, we briefly sketch Roth's approach [26] in the following. In the first step, the points of a given GPS trace are matched to a road r_i and the road points of r_i are matched to the given GPS trace. The details of this matching can be found in [26]. We will focus in the following on an example in which a GPS point is matched to a road point. The processing of a road point that is matched to a GPS trace is performed in an analogous way.

Given that the GPS position p_{gps} is matched to the position (r_i, d) on road r_i, a new road point p_m at position (r_i, d) is generated (see Figure 4-a). The error value $p_m.e$ of this point is calculated by interpolating the error value of its predecessor and successor road point on the road edge, in relation to its distance to these points:

$$p_m.e = \frac{p_{m-1}.e \cdot d(p_m, p_{m+1}) + p_{m+1}.e \cdot d(p_m, p_{m-1})}{d(p_m, p_{m+1}) + d(p_m, p_{m-1})} \quad (3)$$

In the next step, the GPS position p_{gps} and the road point p_m are used to correct the road geometry by calculating a new road point p_{new} (see Figure 4-b). To obtain the coordinates of this point, a weighted interpolation is performed that is based on the standard deviation of p_{gps} and the error value of p_m:

$$p_{new}.x = \frac{p_m.x \cdot p_{gps}.\sigma + p_{gps}.x \cdot p_m.e}{(p_m.e + p_{gps}.\sigma)} \quad (4)$$

$$p_{new}.y = \frac{p_m.y \cdot p_{gps}.\sigma + p_{gps}.y \cdot p_m.e}{(p_m.e + p_{gps}.\sigma)} \quad (5)$$

Furthermore, the error value of the new point is set by using Bayes conditional probabilities applied to the Gaussian distribution [14]:

$$p_{new}.e = \frac{p_m.e \cdot p_{gps}.\sigma}{(p_m.e + p_{gps}.\sigma)} \quad (6)$$

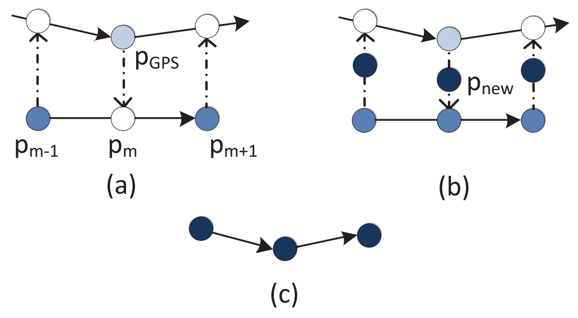

Fig. 4. Processing GPS Trace

It can be easily derived from this formula that the obtained value of $p_{new}.e$ decreases in comparison to $p_m.e$ and, therefore, the confidence in the new position is higher. Having calculated the coordinates and the accuracy value for p_{new}, we remove p_m from the road edge and insert p_{new} at its corresponding position (see Figure 4-c). Since merging a GPS trace with a road results in the creation of additional road points and lower error values for the involved road points, the likelihood that a road segment fulfills the user given quality requirements increases with each processed GPS trace.

Having merged a given GPS trace with the road graph as described above, the server checks for all changed road edges if there are road segments on these edges which fulfill the quality criteria defined in section IV-A. This can be done for each road edge in linear time, starting with the first road point p_0 and increasing the sequence until one of the conditions is violated at point p_i. In this case the server marks the segment $\langle p_0, p_{i-1} \rangle$ as validated and continues checking with p_i as a new starting point. As a result, the server can deduce a set of validated segments for each road edge. This set can now be used to realize an optimized sensing approach that is explained in the next section. Besides the refined road map, this set also constitutes the final output of the system, which indicates the user which parts of the maps are validated.

V. Optimized Sensing

Having introduced the basic concepts of the map correction and validation system, we address in the following the efficiency of the sensing process. For this reason, we introduce an optimized sensing approach that reduces the number of GPS fixes a mobile devices has to execute. The basic idea behind this approach is to suppress GPS fixes that are not needed for the map validation process. This is achieved by restricting sensing to road segments that have not been validated yet.

In order to let the mobile devices know on which road segments GPS fixes can be suppressed, we introduce update messages that are sent from the server to the mobile devices containing the list of already validated road segments. In the following we first show how to suppress GPS fixes by introducing the concept of selective sensing. Subsequently, we present an efficient way for disseminating the required update messages from the server to the mobile devices.

```
Require: $p_{gps}, t_{gps}, V, v_{avg}$
1:  $t_{sup} \leftarrow t_{gps}$
2:  $p_m \leftarrow$ MAP-MATCHING$(p_{gps})$
3:  $seg \leftarrow$ GET-SEGMENT$(p_m)$
4:  if $seg \notin V$ then
5:      return $t_{sup}$
6:  else
7:      $d_{min} \leftarrow min(seg.end - p_m, p_m - seg.start)$
8:      $t_{sup} \leftarrow d_{min}/v_{avg}$
9:      $t_{sup} \leftarrow max(t_{sup}, t_{gps})$
10:     return $t_{sup}$
11: end if
```

Fig. 5. Selective Sensing Algorithm

A. Selective Sensing

Assuming that a mobile device knows all already validated road segments, it can save energy by suppressing its GPS positioning as long as it is located on one of these segments. To detect this, a mobile device matches each obtained GPS position to the nearest point on the road graph and then checks if this position lies on a validated road segment. If this is the case, the device can postpone its next GPS fix until the earliest possible time when it might leave the segment. Having disabled the GPS sensor, the problem is that a mobile device cannot detect whether it is still on a validated segment or not. Therefore, it has to determine a time span that estimates the minimum time it will stay on that segment to schedule its next GPS fix. To get an appropriate approximation for that time span, we calculate it by using the algorithm shown in Figure 5. The algorithm is executed after each position fix and takes as input the current GPS position p_{gps}, the standard positioning interval t_{gps}, the set of validated road segments V and the average speed v_{avg} of the mobile device that executes this algorithm. Since the current speed can be obtained from a GPS fix, a mobile device can determine v_{avg} by averaging the speed values of all fixes performed so far. The output of the algorithm is the time interval t_{sup} that determines the time until the next position fix is performed.

The algorithm starts by identifying the road segment seg to which the GPS position p_{gps} is matched (lines 2–3) and then checks if that segment is already validated (line 4). If that is not the case, the algorithm returns t_{sup} (line 5) that is set to the standard positioning interval t_{gps} (line 1). Otherwise, the distance d_{min} from the closest end of the segment seg to the matched point p_m is calculated (line 7). Given the resulting distance, we use linear dead-reckoning to calculate the minimum time span t_{sup} in which the mobile device might reach the end of the segment seg if it moves at average speed (line 8). Finally, we require t_{sup} to be at least as big as the standard positioning interval t_{gps} and therefore only return the larger value (line 9). By considering the average speed v_{avg} of the mobile device and not taking into account its positioning

```
Require: $p_{gps}, c_o, S_c, t_c$
1: $c_n \leftarrow$ GET-CELL$(p_{gps})$
2: if $c_o \neq c_n$ then
3:     $S_u \leftarrow$ GET-UPDATE$(c_n, t_c(c_n))$
4:     $S_c(c_n) \leftarrow S_c(c_n) \cup S_u$
5:     $c_o \leftarrow c_n$
6: end if
```

Fig. 6. Query Algorithm - Running on Mobile Devices

error, we derive an optimistic estimation for t_{sup}. This may result in a too large t_{sup} if the mobile device moves at a higher speed then v_{avg}. On the other hand, it results in a more appropriate estimation of t_{sup} if the device's speed is v_{avg} or lower, and therefore results in a more effective GPS suppression. After having calculated t_{sup}, the mobile device schedules its next GPS fix at the end of this time span.

B. Update Protocol

In order to perform the selective sensing algorithm described in the previous section, the mobile nodes need to know the current set of validated road segments. Therefore, we introduce a protocol, that ensures an efficient communication between the server and the mobile nodes for updating this set. The simplest way to do this is to broadcast the set of validated segments to all mobile devices in the area-of-interest. Using this approach, all mobile devices have an up-to-date view on all validated segments. If the server marks new segments as validated, it only has to send these segments to update the devices' view. However, receiving the broadcast messages consumes energy at the mobile devices. Therefore, updates messages should only be sent to devices for which the information is actually relevant. Hence, we utilize a more scalable solution that follows a location-aware approach.

The basic idea is to use the information about the current position of a device to inform it only about the set of validated segments that are in its current vicinity. For this reason, we divide the area-of-interest into a grid structure. The coordinates of the grid cells are included in the initialization message and thus are known to all devices. A mobile device associates every grid cell c_i with a timestamp $t_c(c_i)$ and a set of validated segments $S_c(c_i)$. The timestamp $t_c(c_i)$ indicates the last time the device has been located in cell c_i and $S_c(c_i)$ contains the validated segments that are located in c_i and were up-to-date when the device left the cell at time $t_c(c_i)$. Based on this, a mobile device runs the query algorithm shown in Figure 6, right before it executes the selective sensing algorithm presented in the last section. In the first part of the algorithm, the device determines its current grid cell and checks if the cell has changed since the last position fix (lines 1–2). If that is the case, the device sends a query for validated road segments to the server containing the new cell and the timestamp associated with that cell (line 3). Having received an update from the server the mobile device updates its local set of validated segments for that cell (line 4).

```
Require: $c_n, t_n$
1: $S_n \leftarrow$ GET-SEGMENTS-IN-CELL$(c_n)$
2: $S_{out} \leftarrow \emptyset$
3: for all $vs \in S_n$ do
4:     if $t_s(vs) > t_n$ then
5:         $S_{out} \leftarrow S_{out} \cup vs$
6:     end if
7: end for
8: return $S_{out}$
```

Fig. 7. Update Algorithm - Running on Server

To supply the mobile devices with the appropriate set of validated segments, the server has to keep track of all changes it applies to the set of validated segments. Therefore, it associates each validated segment vs with a timestamp $t_s(vs)$ indicating the time when the server marked the segment as validated. If the server now receives a query from a mobile device, it performs the algorithm shown in Figure 7 to return a set of validated segments to the device. Given the required cell c_n and the corresponding timestamp t_n from the mobile device, the server determines the set of all validated segments that are located in c_n (line 1). In the next steps it initializes an output set and adds all validated segments to this output set that were marked as validated after time t_n (lines 2–7).

Both algorithms ensures that a mobile device that changes a cell gets an up-to-date view on all currently validated segments that are located in its new cell. However, the mobile devices only receive an update from the server when they change a cell. Because mobile devices can stay inside a cell for an unknown time, the server also has to proactively distribute information about newly validated segments to the mobile devices. Since the server knows from the query messages in which cell each mobile device is currently located, it can send a set of newly validated segments selectively to those mobile devices which are located in the cells that are affected by this change.

VI. Evaluation

In the evaluation of our concept we conducted extensive simulations to check the effectiveness and efficiency of Map-Correct. Since we utilize for the map correction step existing map fusion algorithms, the resulting accuracy of the road edges heavily depend on the respective algorithm. Therefore, we focus the evaluation on the efficiency of the system and the effectiveness of the map validation process.

A. Simulation Setup

To provide a realistic simulation scenario we evaluated our approach with the network simulator ns-2 using mobility traces from the trace file generator UDelModels [27]. UDelModels uses a given road map to generate mobility traces that are based on well-founded statistical data models. For the generation of the necessary pedestrian traces we used partitions of the road graphs of Chicago and Dallas. The same graphs were taken as input in the map validation query. Since the evaluation for these two maps led to the same results, we only discuss the outcome of the simulations for the Chicago map in the following.

Throughout the simulations we used the following parameter settings: We assumed a requested maximum point distance of $d_{max} = 3$ meters and a maximum error of $e_{max} = 3$ meters. The error value of a GPS fix $p_{gps}.\sigma$ was modeled according to a normal distribution with $\mu = 0$ and $\sigma = 10$ meters. The standard positioning interval was set to $t_{gps} = 3$ seconds. We simulated 2000 mobile nodes moving at pedestrian speed while the simulation time was set to one day (5am till 23pm). To measure the amount of energy that is consumed by each operation on a mobile device, we use a commonly used energy model ([28], [20]) that can be found in Table I.

Operation	Energy [mJ]
GPS Position Fix	75
GPRS Send (1000 Bit)	80
GPRS Receive (1000 Bit)	40

TABLE I
ENERGY MODEL

For the evaluation of our system we compare the results of three different approaches that reflect the different concepts we developed for MapCorrect. At first, we look at a *naive approach* in which all mobile devices continuously record GPS positions without using the presented optimized sensing approach. In the second approach we use the optimized sensing approach by performing selective sensing and broadcasting the set of validated segments from the server to all mobile devices in the area-of-interest. We refer to this as the *broadcast approach*. In the last case, we look again at the concept of selective sensing, but use the grid-based update protocol for distributing the validated segment information to the mobile devices. We refer to this as the *grid-based approach*.

B. Effectiveness of the Map Validation

At first, we have a look at the effectiveness of the three approaches regarding the map validation process. To assess the progress of the map validation we have a look at the road segments that fulfill the quality metrics defined in Section IV-A after a certain time and thus can be considered as validated. The results of the naive approach can be seen as an upper bound for the effectiveness of the system. Since mobile devices perform a continuous sensing in that approach, the server has as much information as possible for the map validation.

Figure 8 shows the cumulated length of all validated segments over time. We see that there is almost no difference between the three approaches at any point in time. As a consequence, we can derive two facts:

1) In the end, there is no difference in the total length of the validated segments. This implies that the two approaches that use optimized sensing (broadcast and grid-based approach) validate the same amount of segments as the naive approach.

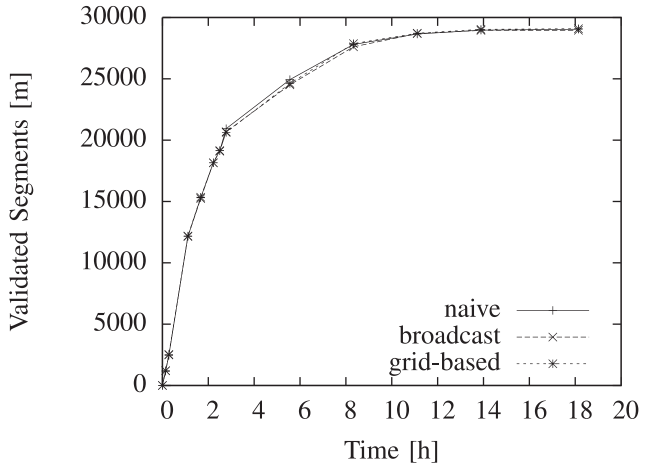

Fig. 8. Cumulated length of validated segments

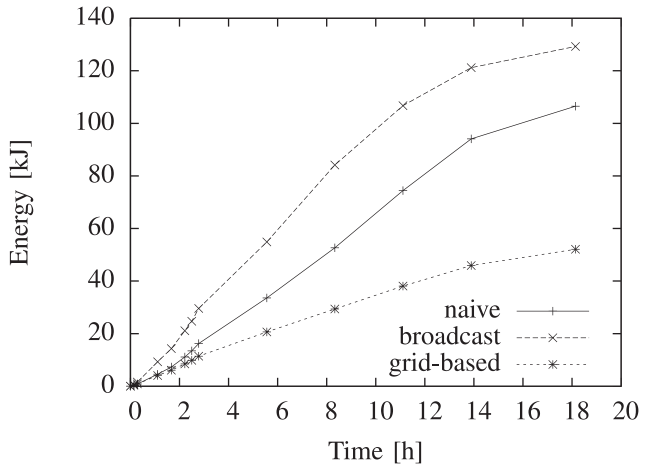

Fig. 9. Total amount of energy consumed

2) There is no temporal gap between the different approaches at any point in time. Hence, the speed of the validation is the same in all three approaches.

We can therefore conclude that optimized sensing does not decrease the effectiveness of the map validation process at all.

C. Energy Efficiency

Next, we have a look at the amount of energy that is consumed by the mobile devices. For this reason, we have a look at the cumulated amount of energy that is consumed by all mobile devices over time. We can see from Figure 9 that the energy consumption of the broadcast approach is the highest followed by the naive approach. The least energy is consumed by the grid-based approach, which saves up to 50% compared to the naive one. Since the energy consumption constitutes from different operations, we have a closer look at the single operations that are performed in each approach to better understand these results.

Figure 10 shows the sum of all GPS fixes over all mobile devices that were performed in the system. We can see that

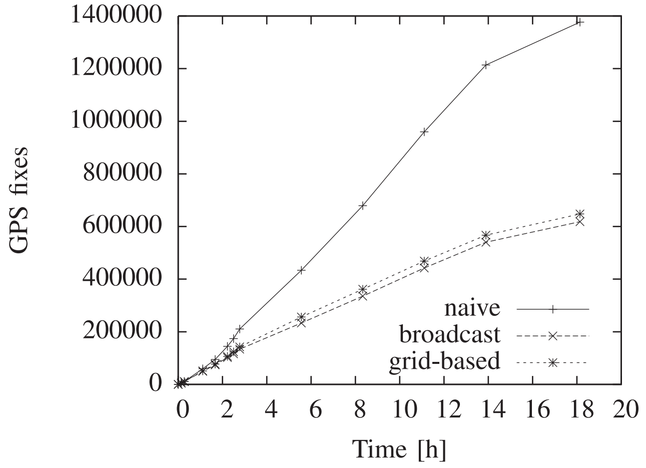

Fig. 10. Total amount of GPS fixes performed

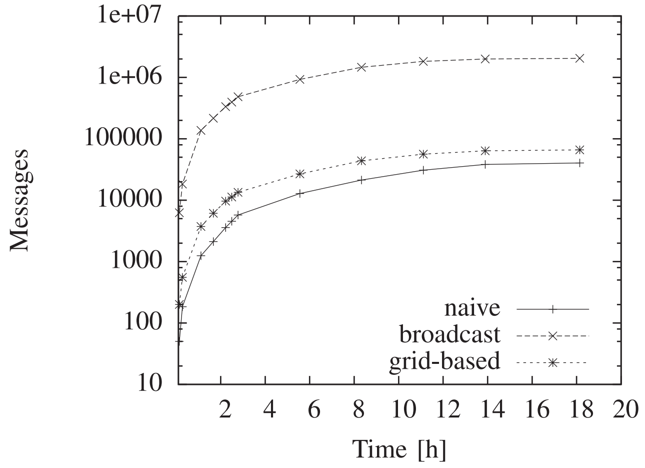

Fig. 11. Total amount of messages sent and received

the approaches that use optimized sensing need 55% less GPS fixes than the naive approach. Moreover, we notice a small gap between the broadcast and the grid-based approach. This stems from the fact that in the broadcast approach a mobile device knows all validated segments from the whole area-of-interest. However, in the grid-based approach a mobile device knows only the validated segments that are located in the device's current cell. As a result, a mobile device can perform a more effective suppression of GPS fixes in the broadcast approach if a validated segment ranges over more than one cell.

Considering the fact that all approaches showed the same effectiveness regarding the speed and the final output, we can conclude that all the additional GPS fixes that were performed in the naive approach were redundant and brought no additional value to the validation process.

At last, we have a look at the total sum of messages that were sent and received by all mobile devices over time. This sum includes the messages used for uploading the GPS traces from the mobile devices to the server and all the messages that are needed in the update protocol. From Figure 11 we can see that the broadcast-based approach by far requires the most number of messages. From that, we can explain why it performs so badly in the overall energy consumption. The grid-based approach needs only a fraction of the messages of the broadcast approach, while the naive approach needs the fewest number of messages since it does not employ update messages.

Comparing the Figures 10 and 11 we can explain the total energy consumption shown in Figure 9 as follows: On the one hand the broadcast and the grid-based approach save up to 50% of energy for GPS fixes over the naive approach. On the other hand the naive approach is the less energy consuming approach considering the number of messages. Hence, the number of messages in the broadcast approach is too high to outperform the naive approach. In contrast, the grid-based approach needs far fewer messages and can therefore compensate the effort for additional messages by requiring far less GPS fixes than the naive approach. To conclude the evaluation we can state that a grid-based approach saves up to 50% of energy compared to a naive map validation approach, whereas the broadcast approach does not pay off in terms of energy consumption. We also showed that although less energy is consumed and less GPS fixes are performed, the quality and the speed of the validation process do not decrease.

VII. Conclusion

In this paper we presented an automatic approach to refine and validate the geometry of an existing roadmap. We proposed a Public Sensing based data acquisition approach that utilizes the already deployed GPS sensors on people's mobile devices for automatically gathering GPS traces and use them for correcting and validating a given road map. Moreover, we introduced an optimized sensing approach, which drastically decreases the amount of energy that is consumed by the mobile devices that take part in the data collection. Although the number of performed GPS fixes decreases, we showed that this approach does not impair the effectiveness of the map validation process.

In future work we want to address further issues concerning Public Sensing based mapping. Besides the validation approach we have presented in this paper, we will focus on an effective map consistency check, in which mobile devices check in predefined time intervals if the already validated geometry of a road map is still valid. This requires a quite different sensor scheduling, since the systems needs to assign the mobile devices sensing tasks, instead of the proactive sensing that the mobile devices perform in MapCorrect. Furthermore, we want to include fairness aspects in the sensor scheduling decisions, which aims at an uniform load distribution among the mobile devices.

Acknowledgments

The work described in this paper was supported by the German Research Foundation (DFG) within the Collaborative Research Center 627 (Nexus).

References

[1] D. Cuff, M. Hansen, and J. Kang, "Urban sensing: out of the woods," *Commun. ACM*, vol. 51, no. 3, pp. 24–33, 2008.

[2] N. Lane, E. Miluzzo, H. Lu, D. Peebles, T. Choudhury, and A. Campbell, "A survey of mobile phone sensing," *Communications Magazine, IEEE*, vol. 48, no. 9, pp. 140–150, September 2010.

[3] T. Abdelzaher, Y. Anokwa, P. Boda, J. Burke, D. Estrin, L. Guibas, A. Kansal, S. Madden, and J. Reich, "Mobiscopes for human spaces," *IEEE Pervasive Computing*, vol. 6, no. 2, pp. 20–29, 2007.

[4] M. Haklay and P. Weber, "Openstreetmap: User-generated street maps," *IEEE Pervasive Computing*, vol. 7, no. 4, pp. 12–18, 2008.

[5] L. Cao and J. Krumm, "From gps traces to a routable road map," in *Proceedings of the 17th ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems*, ser. GIS '09. New York, NY, USA: ACM, 2009, pp. 3–12.

[6] R. Bruntrup, S. Edelkamp, S. Jabbar, and B. Scholz, "Incremental map generation with gps traces," in *Proceedings of the 8th International IEEE Conference on Intelligent Transportation Systems*, 2005.

[7] S. Morris, A. Morris, and K. Barnard, "Digital trail libraries," in *JCDL '04: Proceedings of the 4th ACM/IEEE-CS joint conference on Digital libraries*. New York, NY, USA: ACM, 2004, pp. 63–71.

[8] D. Chen, L. Guibas, J. Hershberger, and J. Sun, "Road network reconstruction for organizing paths," in *Proceedings of the 21st ACM-SIAM Symposium on Discrete Algorithms*, 2010.

[9] J. J. Davies, A. R. Beresford, and A. Hopper, "Scalable, distributed, real-time map generation," *IEEE Pervasive Computing*, vol. 5, pp. 47–54, 2006.

[10] A. Fathi and J. Krumm, "Detecting road intersections from gps traces," in *Geographic Information Science*, ser. Lecture Notes in Computer Science, S. Fabrikant, T. Reichenbacher, M. van Kreveld, and C. Schlieder, Eds. Springer Berlin / Heidelberg, 2010, vol. 6292, pp. 56–69.

[11] S. Minamimoto, S. Fujii, H. Yamaguchi, and T. Higashino, "Local map generation using position and communication history of mobile nodes," in *Pervasive Computing and Communications (PerCom), 2010 IEEE International Conference on*, April 2010, pp. 2–10.

[12] J. Eriksson, L. Girod, B. Hull, R. Newton, S. Madden, and H. Balakrishnan, "The pothole patrol: using a mobile sensor network for road surface monitoring," in *Proceeding of the 6th international conference on Mobile systems, applications, and services*, ser. MobiSys '08. New York, NY, USA: ACM, 2008, pp. 29–39.

[13] P. Mohan, V. N. Padmanabhan, and R. Ramjee, "Nericell: rich monitoring of road and traffic conditions using mobile smartphones," in *SenSys '08: Proceedings of the 6th ACM conference on Embedded network sensor systems*. New York, NY, USA: ACM, 2008, pp. 323–336.

[14] S. Rogers, P. Langley, and C. Wilson, "Mining gps data to augment road models," in *KDD '99: Proceedings of the fifth ACM SIGKDD international conference on Knowledge discovery and data mining*. New York, NY, USA: ACM, 1999, pp. 104–113.

[15] A. T. Campbell, S. B. Eisenman, N. D. Lane, E. Miluzzo, R. A. Peterson, H. Lu, X. Zheng, M. Musolesi, K. Fodor, and G.-S. Ahn, "The rise of people-centric sensing," *IEEE Internet Computing*, vol. 12, pp. 12–21, 2008.

[16] H. Lu, N. D. Lane, S. B. Eisenman, and A. T. Campbell, "Bubble-sensing: Binding sensing tasks to the physical world," *Pervasive and Mobile Computing*, vol. 6, no. 1, pp. 58–71, 2010.

[17] S. B. Eisenman, H. Lu, and A. T. Campbell, "Halo: Managing node rendezvous in opportunistic sensor networks," in *DCOSS*, 2010, pp. 273–287.

[18] D. Golovin, M. Faulkner, and A. Krause, "Online distributed sensor selection," in *Proceedings of the 9th ACM/IEEE International Conference on Information Processing in Sensor Networks*, ser. IPSN '10. New York, NY, USA: ACM, 2010, pp. 220–231.

[19] F. Meng, H. Wang, G. Wei, and Z. Fan, "Energy-efficient and coverage-specific node scheduling for wireless sensor networks," in *Proceedings of the 13th ACM international conference on Modeling, analysis, and simulation of wireless and mobile systems*, ser. MSWIM '10. New York, NY, USA: ACM, 2010, pp. 368–375.

[20] H. Weinschrott, F. Dürr, and K. Rothermel, "Streamshaper: Coordination algorithms for participatory mobile urban sensing." in *MASS'10*, 2010, pp. 195–204.

[21] H. Weinschrott, J. Weisser, F. Durr, and K. Rothermel, "Participatory sensing algorithms for mobile object discovery in urban areas," in *Pervasive Computing and Communications (PerCom), 2011 IEEE International Conference on*, march 2011, pp. 128–135.

[22] D. Philipp, F. Dürr, and K. Rothermel, "A sensor network abstraction for flexible public sensing systems," in *Proceedings of the 8th IEEE International Conference on Mobile Ad-hoc and Sensor Systems*, October 2011.

[23] T. Farrell, K. Rothermel, and R. Cheng, "Processing continuous range queries with spatiotemporal tolerance," *Mobile Computing, IEEE Transactions on*, vol. 10, no. 3, pp. 320–334, March 2011.

[24] "Navstar gps: Global positioning system standard positioning service performance standard," United States Department of Defense, Tech. Rep., 2008.

[25] R. Lange, H. Weinschrott, L. Geiger, A. Blessing, F. Dürr, K. Rothermel, and H. Schütze, "On a generic uncertainty model for position information," in *Proceedings of the 1st International Workshop on Quality of Context*. Springer, Juni 2009, Workshop-Beitrag, pp. 1–12.

[26] J. Roth, "Extracting line string features from gps logs," *Schriftenreihe der Georg-Simon-Ohm-Hochschule Nuernberg*, 2008.

[27] J. Kim, V. Sridhara, and S. Bohacek, "Realistic mobility simulation of urban mesh networks," *Ad Hoc Netw.*, vol. 7, pp. 411–430, March 2009.

[28] H. Weinschrott, F. Dürr, and K. Rothermel, "Efficient capturing of environmental data with mobile rfid readers," in *Mobile Data Management*, 2009, pp. 41–51.