Technischer Bericht 2014/06

Quality-based Adaptive Positioning for Energy-Efficient Indoor Mapping

Patrick Baier, Damian Philipp,
Frank Dürr, Kurt Rothermel

Institut für Parallele und Verteilte Systeme
Universität Stuttgart

Universitätsstraße 38
70569 Stuttgart

November 2014

Quality-based Adaptive Positioning for Energy-Efficient Indoor Mapping

Patrick Baier
Universität Stuttgart
Stuttgart, Germany
patrick.baier@ipvs.uni-stuttgart.de

Damian Philipp
Universität Stuttgart
Stuttgart, Germany
damian.philipp@ipvs.uni-stuttgart.de

Frank Dürr
Universität Stuttgart
Stuttgart, Germany
frank.duerr@ipvs.uni-stuttgart.de

Kurt Rothermel
Universität Stuttgart
Stuttgart, Germany
kurt.rothermel@ipvs.uni-stuttgart.de

ABSTRACT

The availability of maps is one of the major prerequisites for deploying location-based services. However, only very few maps are publicly available for indoor environments. To overcome this problem, first approaches emerged to automatically derive indoor maps from odometry traces that pedestrian collected voluntarily. Although these approaches showed to be effective to automatically create indoor maps from mobility traces, they require energy-intensive positioning based on inertial navigation systems (INS) to collect traces of high quality, which impacts the users' willingness to participate in indoor mapping with his energy-constrained mobile device.

In this paper, we tackle this problem by providing an extension to automatic indoor mapping systems that lowers the energy consumption of the participating devices significantly. More precisely, we provide a framework enabling mobile devices to turn off INS while moving in areas that have been mapped already with high quality. In order to enable the dynamic re-start of INS—which requires an initial position and direction—when entering insufficiently mapped areas, we combine INS with low-energy WiFi-based position recovery. WiFi-based position recovery enables a coarse-grained position tracking while the inertial positioning is off and allows for bootstrapping INS by providing an initial position when needed.

Using our approach, we show that indoor models can be derived saving up to 25% of energy on the mobile devices without compromising on the quality of the derived maps.

Categories and Subject Descriptors

C.2.4 [**Distributed Systems**]: Distributed applications

General Terms

Algorithms

Keywords

Mobile Computing, energy-aware systems

1. INTRODUCTION

Location-based services (LBS), such as geo-social networks and navigation systems are well-established today. Although first LBS were focused on outdoor scenarios, indoor LBS are a new trend to offer services like indoor navigation or shopping assistants. However, while outdoor maps are readily accessible, only few indoor maps are available, which prevents the large-scale deployment of indoor LBS. The major reason for missing indoor maps is the labor-intensive task of indoor mapping, which typically requires manual interaction and, thus, is not scalable. As a result, only very few indoor LBS have been actually deployed so far.

Recently, first approaches showed the feasibility to automatically infer indoor maps from pedestrian mobility traces recorded by smartphones [1, 6, 11, 15, 22]. For instance, our MapGENIE approach [15] shows that it is indeed possible to derive indoor maps from noisy and drift-based pedestrian traces using an indoor grammar to support the mapping process.

All of these indoor mapping systems rely on the availability of indoor traces that are voluntarily provided by mobile users. Typically energy-intensive inertial navigation systems (INS) are used to provide high-quality traces for accurate mapping, which consume significant amounts of energy. Since users are sensitive to short battery lifetimes, this significantly impacts the willingness of users to participate.

In order to tackle this problem, we focus on the energy-efficient gathering of traces for automatic indoor mapping in this paper. The basic idea to save energy is to turn off

energy-intensive INS in areas that have been mapped accurately already and only turn on INS in areas that require further traces to generate accurate maps. For instance, a very busy area of a building is visited very often and, therefore, a lot of trace data is available as input to the mapping system. If that area is mapped with sufficient quality after a short time, collecting further trace data from that area does not improve the quality of the indoor model anymore. Hence, collecting further traces in that area would waste rare energy resources without adding benefit to the system.

In more detail, we propose an algorithm that monitors the mapping process and automatically detects sufficiently mapped areas according to a proposed quality metric, which is generic and applicable to existing mapping approaches. The locations of such accurately mapped areas are communicated to the devices, which turn off energy intensive INS while moving in accurately mapped areas and turn on INS again only when entering areas with low accuracy maps.

However, turning on INS on demand is not straightforward. INS is based on angular and linear acceleration values, which are integrated to calculate a motion vector that is added to an absolute (initial) position (dead reckoning). To turn on INS, we need to provide a current absolute position to enable dead reckoning. To tackle this problem, we combine INS with a low-energy WiFi-based position recovery approach. WiFi-based position recovery provides so-called anchor points as absolute reference points when turning INS on again. We show that the ability to disable and enable INS on demand helps to reduce the overall energy consumption of the mapping system significantly.

In summary, we make the following contributions: (1) We propose a quality metric to quantify the accuracy of areas of an indoor map. (2) We present a scheduling algorithm to decide when to turn INS on or off. (3) We propose a position recovery system that allows to resume INS positioning using WiFi anchor points, which are automatically collected during the mapping process with little energy overhead. (4) We evaluated our system with real-world traces and show that it helps to save up to 25% of energy while not compromising on the quality of the generated indoor maps.

The remainder of this paper is structured as follows: In Sec. 2 we discuss related work to our approach before we introduce the system model in Sec. 3. Next, Sec. 4 introduces the results of our energy comparison between INS and WiFi positioning. After giving an overview of our approach in Sec. 5, we introduce the quality metric in Sec. 6. Section 7 presents our WiFi-based position recovery system, before Sec. 8 explains the scheduler component that decides whether to turn off INS. Section 9 shows the evaluation results of our system, before Sec. 10 concludes this work.

2. RELATED WORK

In the literature, several approaches have been published proposing algorithms to automatically generate indoor floor plans from mobility traces collected by crowds of users [1, 6, 11, 15, 22]. Although these approaches use different algorithms to generate a floor plan, all of them require mobility traces as input. However, none of them considers the high energy consumption for the collection of traces by mobile devices, which might lower the user acceptance of these applications.

Although energy efficiency has not been targeted for indoor mapping so far, it has been a goal in other mobile sensing applications using different methods. One idea is the use of specialized hardware for sensor data processing [7, 18, 20]. Other approaches save energy by limiting spatio-temporal coverage and thus reducing the amount of sensor readings taken [3, 10] or by guiding users to most informative places to take sensor values [5]. In our previous work [16, 17], we showed how to use models on sensor data correlation to reduce the amount of information that has to be collected by mobile devices. Moreover, we designed algorithms for selective sensing by subsets of mobile devices for mobile target tracking [25] and opportunistic sensor data collection from stationary sensors by mobile devices [24]. Furthermore, in our previous work on outdoor map correction [2], mobile devices selectively disable their GPS when moving on roads that need no correction. However, none of these works deal with the specific challenges of an indoor environment, namely INS positioning.

In our approach, we combine low-energy WiFi positioning with high-quality INS positioning to selectively enable INS sensing using coarse WiFi reference positions (anchors). So far, there exist different approaches that proposed the combination of WiFi fingerprinting and INS to enhance indoor positioning. Noh et al. [13] propose an infrastructure-free positioning system using dead reckoning and WiFi beacons sent by peer devices for positioning. However, they require a known map to estimate the visibility of WiFi signals between devices. Other work [19, 21, 23] uses WiFi fingerprints to correct the error of drift-based INS traces. They identified landmark spots within the building that can be identified using WiFi and then corrected the direction drift of the INS traces to match the landmark. In our work, we use a similar approach but for a different goal. Instead of correcting a drift-based trace, we show how WiFi can be used to (re-)start INS without any available absolute position.

Other approaches for indoor positioning use predeployed RFID-Tags [12] or Bluetooth Beacons [4, 14]. However, our approach does not rely on environments being prepared with specialized equipment, but instead exploits the already available deployments of WiFi access points.

3. SYSTEM MODEL

In the following, we introduce our system model. Figure 1 gives an overview of its components.

Our system consists of a server managing the floor plan and mobile devices providing mobility traces for automatically generating the floor plan on the server side.

We assume that each mobile device uses an *INS* to record odometry traces and sends them to the server using wireless communication (e.g., using WiFi, 3/4G). The INS calculates positions from an initial position using dead reckoning by adding motion vectors. These vectors are calculated from acceleration values sensed by accelerometers and gyroscopes. Therefore, recorded INS traces are subject to drift errors. However, we assume that these traces can be au-

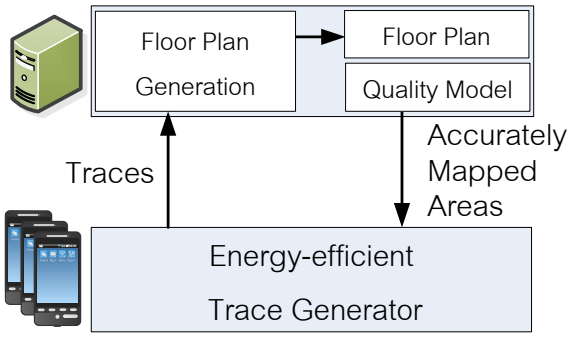

Figure 1: Components of the system model.

tomatically aligned and corrected as shown in our previous work [15]. Moreover, each device has a *WiFi* interface to scan for signal strength (RSSI) values of nearby WiFi access points. As shown later, we use these RSSI values to calculate coarse-grained positions called anchor points to provide initial positions to INS. Due to the method of integrating acceleration values, INS is susceptible to fast movements, which might occur when the participant uses the phone while walking. Making INS robust to such movements superimposing accelerations from walking is an active research area and beyond the scope of this paper. Here, we assume that we can identify phones which can be used for INS positioning. For instance, we will not use phones whose screen is on, since this typically happens when the participant uses (and thereby moves) the phone.

The server executes a *floor plan generation component* that automatically generates a floor plan from a set of recorded traces. Our system for energy-efficient trace collection can work with any concrete mapping algorithm generating floor plans from traces. To evaluate our system, we used our MapGENIE algorithm [15].

A *floor plan* consists of spatial objects, namely hallways and rooms. Each room is accessible from at least one hallway. We assume individual hallway objects to be sections of the buildings hallway network that are free of intersections. Hence, intersections of the hallway network are formed when the ends of two or more hallway objects meet. As every hallway network can be projected into this definition, this does not impact the generality of our approach. Furthermore, we define an assignment of rooms to hallways based on accessibility, i.e., every room is assigned to exactly one hallway from which it is accessible in an arbitrary constant fashion.

In order to control the process of energy-efficient trace collection, the floor plan is partitioned into *areas* by an operator, depending on the requirements of the trace generator. Areas are defined such that they only contain whole floor plan objects, e.g., do not cut a room in half. The generated floor plan is given as input to a *quality model* quantifying the areas of the floor plan according to a quality metric. For the purpose of energy-efficient trace collection, a binary classification of areas is sufficient, classifying areas as either accurately mapped (AM-areas) or inaccurately mapped (IM-areas). In Sec. 6, we will present a concrete quality model for our approach. In general, area partitioning as well as the classification algorithm depend on the concrete mapping algorithm. To evaluate our system, we define an area to be comprised of a single hallway and all rooms accessible from that hallway, i.e., every hallway forms its own area. AM-areas and IM-areas are communicated back to the devices, which locally store this information to control the energy-efficient trace collection process.

To provide mobility traces for the floor plan generation, each mobile device runs the *energy-efficient trace generator*, which is the main contribution of this work. The goal of this component is to provide traces with minimum energy-overhead for mobile devices. To this end, it should only record traces in IM-areas with insufficient quality according to the quality model. These traces have to be recorded using INS to have sufficient accuracy for mapping. In AM-areas, which have been accurately mapped already, INS should be turned off to save energy.

4. ENERGY EFFICIENCY OF INS AND WIFI POSITIONING

The basic idea of our approach is to turn off energy-intensive INS positioning when the mobile device enters AM-areas and to turn it on again before entering IM-areas. In order to decide when the device enters an IM-area, it still needs a position while moving in AM-areas. Since INS is turned off in AM-areas, we need another method to track the device and provide an initial position to INS when it should be turned on again. To this end, we use WiFi-based position recovery through RSSI fingerprinting.

Obviously, in order to increase energy efficiency with this approach, scanning for WiFi fingerprints must consume less energy than running the INS. To backup this assumption, we evaluated the energy consumption of INS and WiFi scanning with an experiment: We implemented both positioning methods on a Samsung Galaxy Nexus i9250 and attached a measurement device between the battery and the device to measure the device's power drain. Figure 2a shows the energy consumption with INS implemented according to [8]. Figure 2b shows the results for an algorithm performing a WiFi scan every 10 seconds.

Having a look at the power over time for INS, we see that running the INS keeps the device constantly active. This can be attributed to the fact that the device has to continuously sample its sensor and analyze the sensor data. Therefore, the CPU cannot go into IDLE mode.

In contrast, although the peak power consumption at the times when a WiFi scan is performed (Fig. 2b) is higher than using INS (Fig. 2a), the device is able to power down to the energy-saving IDLE mode between two scans using WiFi. As a result, the WiFi-based positioning is much less energy consuming than INS when using a sample rate of 10 s between two scans. On average the INS consumes 0.45 W, while the WiFi system only consumes 0.16 W.

This study shows that it is worth turning off INS positioning and using WiFi fingerprinting in areas that are sufficiently mapped.

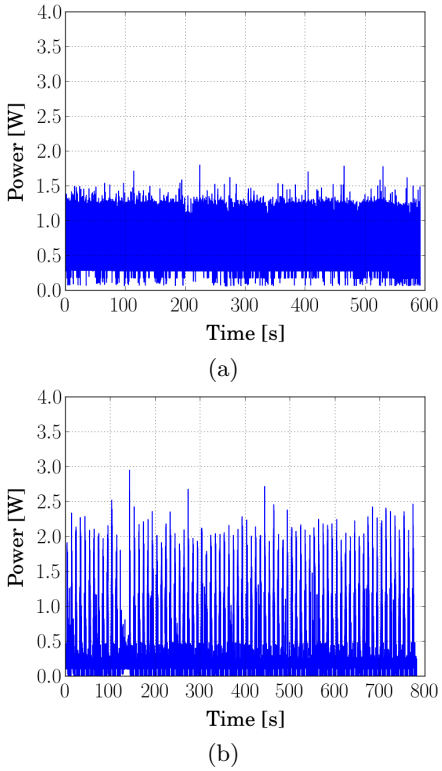

Figure 2: Energy consumption of the mobile device over time: (a) Device runs INS. (b) Device performs WiFi scan every 10 s

5. OVERVIEW

We now present an overview of our approach which is implemented by the trace generator component. Figure 3 shows the components of the trace generator. As mentioned before, a device receives the areas that are considered as accurately mapped (AM-areas) from the server. Using this information, the *Scheduler* decides, whether to turn INS on or off. When the device is going from an IM-area into an AM-area, INS and trace uploading is turned off, and the device continues to track its position using WiFi.

INS should be turned on again before entering an AM-area. For making the decision, whether the device is entering an IM-area and for resuming INS, current device positions are required. To this end, WiFi position recovery is used. We implemented WiFi position recovery based on so-called *anchor points* to provide coarse positions with small energy overhead. Anchor points are points with known position and RSSI fingerprint. Devices map their position to the closest anchor point based on the measured RSSI to get a coarse position. The data for establishing these anchor points is learned on-the-fly, as we will see later on. Hence, we do not need any prior knowledge about WiFi signal strengths to run the position recovery.

When the scheduler decides to resume INS positioning, the INS needs to be provided with an absolute starting point. In our architecture, the *INS Initialization* component allows for dynamically resuming INS positioning using WiFi anchor points. In order to precisely know when a device is actually

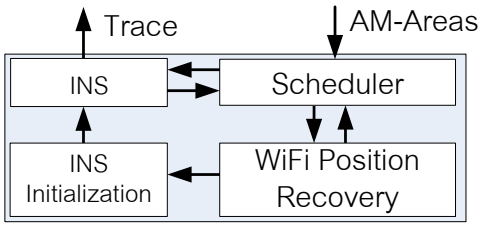

Figure 3: Architectural overview of the energy-efficient trace generator.

at an anchor point, we also utilize acceleration data after the INS is turned on again to identify sharp turns at corners (hallway intersections and turns). Since we also strategically place anchor points at each corner, we can precisely align the trace when the device reaches a corner making a turn.

The scheduler also uses anchor points to determine whether the device is going to enter an IM-area. However, the decision when to turn on INS again is more complicated than the decision whether to turn it off. As described, in order to turn on INS, we need an initial position, which is only accurate when the device crosses an anchor point. Therefore, we have to make sure that the device crosses an anchor point *before* entering an IM-area, to have a chance to turn INS on in time. The scheduler uses mobility prediction to predict whether a device will enter an IM-area in the near future, to start INS before it actually enters the IM-area.

Recorded trace data is periodically uploaded to the infrastructure. The upload period is in the range of several minutes to ensure the freshness of the indoor model on the server. After receiving a trace, the server calculates a floor plan in the *floor plan generation* process using all received traces and updates the quality model with a fresh set of AM-areas and sends them to the devices.

6. QUALITY MODEL

The goal of our quality model is to label areas of the floor plan as AM-areas or IM-areas, thus, informing the scheduler whether additional traces should be recorded for an area. Intuitively, an area is accurately mapped if no more information can be gained from traces. Theoretically, this is the case when an area is completely filled with floor plan objects (hallways and rooms) with no empty space left in between. However, due to inaccessible spaces, e.g., maintenance access rooms with practically no visits or furniture blocking access to walls, this leads to blank spots that cannot be covered by traces. From traces alone it is in general undecidable whether there is no information for an area due to missing traces (no user traveled there although it would have been possible) or due to an obstacle (no user can physically travel there). Therefore, it is impossible to define the perfect quality metric based on traces alone. Hence, such a metric has to be based on heuristics. In the following, we define a generic quality metric applicable to all trace-based algorithms.

We define the quality of an area A (AM-/IM-area) based on a threshold $\hat{T}^{area}$ of the relative spatial coverage Q_A of A by accurately mapped objects (AM-objects). For instance, if the spatial area of all accurately mapped objects

in A is more than 90%, and $\hat{T}^{area} = 0.8$, then A is accurately mapped. Whether an object is accurately mapped (AM) or inaccurately mapped (IM) is defined by the number of observations, i.e., how many times this object was observed in the underlying trace data. This metric ensures that we do not consider A to be an AM-area if there is a very large number of observations for a small part of A (e.g., few rooms with many visits) and few observations for the remainder. It also tolerates a certain amount of inaccessible space per area without having to accept a lot of inaccurately mapped objects (IM-objects). The rationale behind this metric is based on the following observations about mapping approaches and real user mobility.

All mapping approaches based on traces require users to cover the complete space of an object to determine its correct size. As it is unlikely that any single user will cover a complete object with a single trace—e.g., covering the complete length *and* width of a hallway or room—we require multiple observations of each object. However, with an increasing number of observations already available, the chance to discover a new feature of the object diminishes. We therefore define a floor plan object o to be an AM-object when the number of observations $|o|$ is above a certain threshold: $|o| \geq \hat{T}^{type(o)}, type(o) \in \{room, hall\}$. Note that we use different threshold values for $\hat{T}^{room}$ and $\hat{T}^{hall}$, as movement in hallways is typically less constrained than in rooms and thus fewer traces are sufficient to cover the full width of a hallway.

However, only basing the area quality metric on the relative spatial coverage by AM-objects will often lead to suboptimal results due to the fact that large popular objects will dominate small and/or unpopular objects. In particular, hallways often cover large parts of an area and are visited by many users. Therefore, after a large popular hallway has been accurately mapped—which due to its popularity typically happens quickly—, no more rooms would be detected since the area is well-covered by AM-objects. To avoid this problem, we use a two-phase approach as shown in Fig. 4: First, we ensure in Phase 1 that all hallways have been detected (l. 1–2). Then in Phase 2 we focus on the mapping of rooms only (l. 4 ff.).

Note that as mentioned above it is undecidable from traces alone whether we have detected all objects (including all hallways). However, for the specific case of hallways, topological constraints typically apply that make the problem decidable for hallways. Hallways are typically connected to other objects (further hallways or rooms) or an outer wall on both sides. Thus, as long as there is empty space at either side of the hallway (we say that the hallway is *not terminated*), it is not accurately mapped since its coverage is still unknown, and the hallway is marked as IM-area (l. 1). If we have detected both ends of a hallway and it received enough traces, it is considered to be accurately mapped. If every hallway is accurately mapped, we start Phase 2 and focus from there on the mapping of rooms based on the relative coverage of the area by AM-objects (l. 4). Since at this point we know that hallways have been accurately mapped already, we subtract their area from the total area under consideration.

1: **if** $\exists h_i \in A : |h_i| < \hat{T}^{hall} \vee h_i$ not terminated **then**
2: **return** IM-Area ▷ Phase 1
3: **else**
4: $Q_A \leftarrow \frac{\sum \{geoArea(r_i) || r_i | \geq \hat{T}^{room}\}}{geoArea(A) - \sum \{geoArea(h_j)\}}$ ▷ Phase 2
5: **if** $Q_A \geq \hat{T}^{area}$ **then return** AM-Area
6: **else return** IM-Area

Figure 4: Algorithm for computing the quality of an area A. $geoArea()$ defines the area covered by an object, $|o|$ the number of observations of o. h_i denotes a hallway, and r_i denotes a room.

Note that all threshold values $\hat{T}^{\{room,hall,area\}}$ are parameters of the quality model that need to be set by an operator. $\hat{T}^{hall}$ and $\hat{T}^{room}$ must be adjusted depending on the floor plan generation component, e.g., to account for the varying degree of accuracy of different mapping algorithms. $\hat{T}^{area}$ can be adjusted according to the desired level of accuracy of the indoor model and thus defines an energy/accuracy-tradeoff. $\hat{T}^{area} = 1.0$ would only classify areas as AM-areas, when every detail about them is known. From our evaluations, we could conclude that choosing $\hat{T}^{area} \leq 0.17$ allows significant energy savings without compromising on the quality of the derived floor plan.

7. WIFI POSITION RECOVERY

The goal of WiFi position recovery is to provide the scheduler with a coarse device position when the INS is off. Using this position, the scheduler can detect when a device is about to leave an AM-area and start the INS initialization.

Our WiFi position recovery approach is based on the concept of location anchors. The idea of anchor-based positioning is to supply each device with a list of geographically distributed anchor points including their geographic position and RSSI values measured at these anchor points during anchor point setup. A device can estimate its position by identifying the closest anchor point by performing a WiFi scan and selecting the anchor point whose RSSI fingerprint is closest to the RSSI values measured by the device using an Euclidean distance metric.

Before we can use anchor points for positioning, we first need to set up anchor points at known positions. Since we assume no prior knowledge about the building—in particular, no knowledge about anchor points—, anchor point setup has to be performed "on-the-fly" during the mapping process. For setting up an anchor point, a device needs to measure the RSSI value at a known location. Since devices know their location from INS while collecting traces in IM-areas, we set up anchor points during trace collection.

For setting up suitable anchor points we have to fulfill three requirements: First, since WiFi scans consume energy as shown in Sec. 4, we should only setup anchor points at few locations to minimize the energy overhead. Second, we need to select locations that make it easy for devices to find their coarse position from few WiFi scans to track the device position while moving in AM areas. Third, we need to select locations, that also allow for precise positioning to resume INS with an initial position.

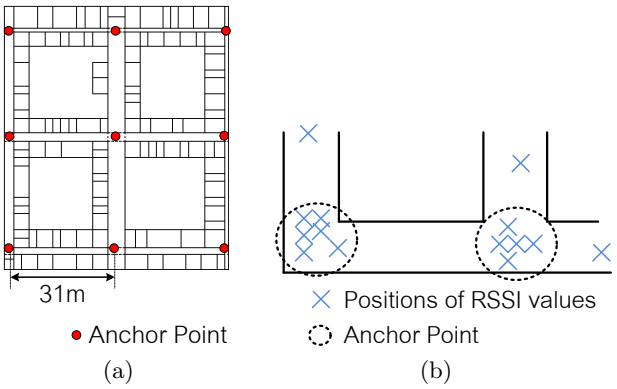

Figure 5: Building the WiFi anchor point database: The left figure shows the location of the WiFi anchor points. The right figure shows the points where the RSSI values where actually recorded.

To solve all three problems, we decided to place anchor points only at hallway intersections (see Fig. 5a). Since there are typically only few hallway intersections, the number of anchor points and necessary WiFi scans is limited. The position of intersections can be identified easily by looking for sharp turns of about 90 deg in the INS values during trace collection. To set up anchor points, devices perform WiFi scans whenever they make sharp turns and upload the measured RSSI value together with the current device position to the server. The server uses the uploaded RSSI values and associates them with an anchor point according to the INS position where the RSSI was recorded. RSSI values that were recorded aside from these anchor points are filtered out (see Fig. 5b). Note that the position of RSSI values can lie aside from intersections if the recorded INS trace is inaccurate or the user made a sharp turn to enter a room instead of turning at an intersection. The result is a list of anchor points, each one associated with a set of attached WiFi RSSI values and positions. This list is communicated back to the devices when the server sends the AM-areas to the devices.

These anchor points at intersections can be used for coarse positioning while devices move in AM-areas with INS turned off. For coarse WiFi positioning, a device performs a WiFi scan every 10 seconds. The device compares the obtained RSSI values with the anchor point list and uses the Euclidean distance between the RSSI values to identify the closest anchor point. In general, WiFi-based indoor positioning has an accuracy of about 4 m [9]. In general, hallway intersections in indoor environments are much further apart than 4 m, since a smaller distance between intersections would imply that there are rooms in between that are even smaller than this distance, which is not a realistic assumption. As a result, a device is able to identify the correct intersection with high accuracy (cf. Sec. 9.2). Using coarse WiFi positioning, a device is able to identify its closest intersection point and to track its current (topological) path through the building. This information serves as input for the scheduler presented in Sec. 8.

Finally, anchor points at intersections can also be used to obtain a precise position for resuming INS positioning by

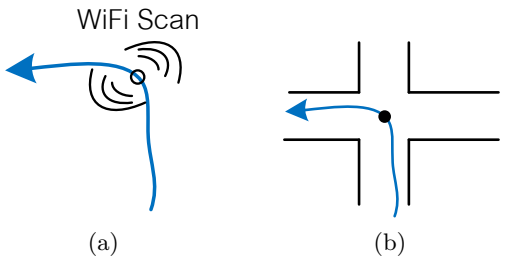

Figure 6: Reposition of a relative INS trace to the center of a intersection that was mapped by WiFi.

the INS Initialization component. A device that resumes INS positioning on request of the scheduler just has to turn on INS, wait for a 90 deg turn as indicated by the gyroscope (angular acceleration) values, and perform a WiFi scan at this point. By comparing these RSSI values with the fingerprints of anchor points and selecting the point with the best matching fingerprint, the device knows its precise location, namely, the location of the anchor point at the intersection (see Fig. 6a). Knowing this intersection together with the knowledge about the last hallway in which the device moved, the device can position the relative INS trace recorded so far and map it to the absolute center of the intersection (see Fig. 6b). At this point, INS initialization is finished and INS positioning continues.

Note that the trace might not be perfectly accurate if the device has not turned exactly on the center of the intersection. We investigate this effect further in the evaluation in Sec. 9. Furthermore, it is possible that the INS indicates a 90 deg turn even though the device did not pass an intersection. This can be the case if the device moves into a room, for instance. Hence, to clearly identify that a device turns at an intersection, we also require that the trace before and after the 90 deg turn is straight for a distance that is longer than the depth of the biggest room. Our evaluations showed that this is sufficient to separate turns at intersections from other turns.

8. SCHEDULER

In this section, we introduce the scheduler algorithm, which decides when to turn INS on and off. As motivated in Sec. 5, INS has to be turned on ahead of time before entering the IM-area using a mobility model for predicting, whether the device will enter an IM-area in the near future. We start by introducing this mobility model indicating to which area a device is likely to move. Based on this model, we provide an algorithm that decides when the INS should be turned off or resumed.

8.1 Mobility Model

Given the current user trace as an input, the mobility model outputs the probabilities of possible continuations of the trace. From these probabilistic trajectories, we then determine the probability $P(a|t)$ that the device, having traveled along trace t, will visit a certain area a. Based on $P(a|t)$, the scheduler can then decide whether to stop or resume INS positioning.

Note that we make no further assumptions on the capabili-

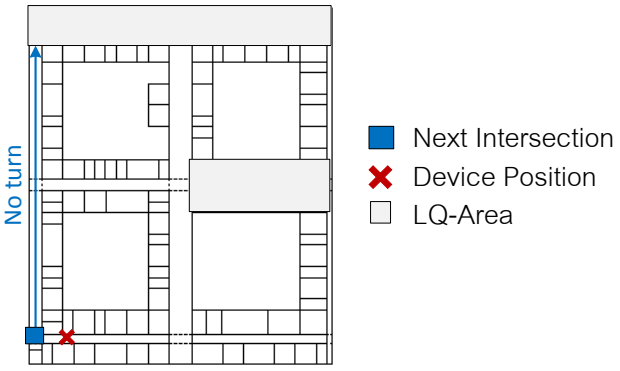

Figure 7: Only the upper IM-area can be reached without a turn from the next intersection that the device passes.

ties of the indoor mobility model. Designing a sophisticated indoor mobility model is an open research question beyond the scope of this paper. For the purpose of this work, we define a simple mobility model based on the (reasonable) assumption that users take the shortest path towards their destination. Obviously, more complex models taking, for instance, user habits into account, can lead to even more accurate predictions.

Given the path t that the user has previously traveled, we first compute the set A of reachable areas. An area a is reachable from trace t if t is a prefix of a shortest path from the origin of t to a (note that depending on the building layout, multiple shortest paths may exist). As in our simple model there is no information whether for any two destinations $a_i, a_j \in A$ one is more likely than the other.Hence, we assign a uniform visiting probability $P(a_i|t) = 1/|A|$ to each area a_i.

8.2 Turning off INS

To turn INS off, two conditions must be fulfilled: (1) The device is currently moving in an AM-area. (2) INS positioning can be resumed before re-entering an IM-area. If the device detects that both of these conditions are fulfilled, it turns off INS.

While the first condition can be easily be checked by comparing the current INS position of the device to the AM-areas, the second condition is harder to evaluate.

To resume INS positioning, a device must turn at an intersection by 90 deg before entering an IM-area. Otherwise the INS initialization cannot be completed because there is no anchor point to which the relative trace of the INS can be mapped to. This can be ensured as follows: When the INS is on, a device can determine towards which direction it is currently moving. We denote this intersection as I_n. In a next step, we define the set A_{I_n} to contains all IM-areas that are reachable from I_n without a turn (see Fig. 7). For any area $a \in A_{I_n}$, intersection I_n is the last turning point if the device walks directly towards a. Hence, the INS is not turned off if it is very likely that the device moves to one area $a \in A_{I_n}$.

To estimate the probability that the device moves to A_{I_n}, we use the mobility model. More precisely, for every $a \in A_{I_n}$, we get $P(a|t)$ from the mobility model and then calculate the probability that the device will move to at least one of these areas as:

$$p_{LQ} = 1 - \prod_{\forall a \in A_{I_n}} 1 - P(a|t) \quad (1)$$

To decide if the INS stays on or will be turned off, we draw a random sample s_r form the uniform distribution between $[0, 1]$ and set:

$$INS = \begin{cases} off, & \text{if } s_r > p_{LQ} \\ on & \text{else} \end{cases}$$

Once the INS was turned off, the following algorithm determines, when INS positioning will be resumed.

Note that this probabilistic decision also ensures that every now and then INS traces are recorded for areas that are already sufficiently mapped. This gives our system a chance to discover features that are observed by only very few traces, i.e., rooms that are rarely ever entered. Furthermore, it provides a way to notice changes in the floor plan, i.e., when an obstacle such as a desk has been moved.

8.3 Resuming INS Positioning

As defined in the last step, the INS can only be turned off when the device moves in an AM-area. Before the device leaves that area, INS positioning must be resumed.

Basically, this it the same requirement that is given in condition (2) in the last subsection. Hence, we apply the same algorithm to decide whether the INS initialization should be started. However, this algorithm takes the closest intersection I_n as input. Since the device switches to WiFi position recovery once the INS was turned off, the device can keep track of the hallway in which it is moving. Using the history of the last visited hallways, it can determine the intersection I_n towards which it is moving.

9. EVALUATION

The evaluation of our proposed system consists of three parts. In the first part, we analyze the accuracy of WiFi positioning and INS initialization. In the second part, we take a look at the energy efficiency of our system and compare it with existing solutions. Finally, we show how our system affects the quality of the generated floor plan. Before we describe each of these parts in detail, we introduce our experimental setup.

Note that we do not evaluate the performance of the underlying mapping algorithm. For an analysis of its properties and performance with respect to mapping quality, see the original publication [15]. However, we compare the performance of our optimized approach to that of the underlying mapping algorithm. Since the (pure) mapping algorithm performs indoor mapping by continuously running the INS, it will serve as reference approach for evaluating the quality of the energy-efficient mapping approach presented in this paper.

9.1 Experimental Setup

To evaluate our system, we used our MapGENIE dataset [15]. This publicly available data[1] includes 154 odometry traces, collected from four volunteers with an Android smartphone in combination with a foot-mounted inertial measurement unit. The total length of these traces is more than 22 km.

To test WiFi position recovery and INS initialization, we first collected WiFi RSSI data from intersection points in the same building in which the MapGENIE dataset was recorded. From this data, we set up the WiFi anchor point database. To evaluate our algorithms with the trace data from [15], we built a trace simulator that iteratively replays each trace and executes our energy-efficient trace generator. When a WiFi scan is performed by our algorithms, we assign the RSSI values that were collected at the respective intersection.

9.2 Positioning Accuracy

To test the WiFi position recovery algorithm, we performed WiFi scans at 30 arbitrary positions in the building with an Android phone and identified the closest anchor point from the database by comparing the RSSI values. We consider the assignment to be accurate if the closest intersection from the ground truth position is equal to the intersection associated with the identified anchor point. Furthermore, we are interested in how many WiFi scans per anchor point should be stored in the database in order to reach an accurate anchor point identification. Hence, we evaluated the accuracy of identifying the correct intersection when different number of scans per anchor point are stored in the database.

Figure 8a shows the success rate of this experiment. The y-axis shows the fraction of correctly identified anchor points while the x-axis shows the number of WiFi scans from the database used for comparison. We see that if at least three WiFi scans are available in the database at each anchor point, the closest anchor point can always be accurately identified. Hence, only a small set of reference data is needed to implement accurate WiFi position recovery. The energy overhead for building the anchor point database is therefore negligible.

As mentioned before, the INS initialization cannot perfectly reestablish a position since it always maps the relative trace to the center of an intersection. If a user does not turn directly at the center of the intersection, the reestablished trace is shifted. Figure 8b shows a CDF of this shift error in meters. From this plot we see that in 50% of the cases this error was smaller than 3 meters and in 90% of all cases the error was smaller than 5 meters. Hence, in most of the cases INS positioning can be resumed without having a large positioning error. Moreover, this positioning errors do not accumulate over time, since at each intersection the position gets recalibrated. In Sec. 9.4, we will see that this small shift does not impact the quality of the generated floor plan.

9.3 Energy Efficiency

Figure 9 shows the total energy consumption of all devices for different number of traces that were replayed. To evaluate the energy consumption we used the energy measurements that we obtained from our study in Sec. 4. More precisely, we simulated the total energy consumption of a device by considering the energy for the different operations it performs, which we got from our measurements. The *basic* approach denotes the unoptimized execution of the mapping algorithm when all devices are running INS all the time. The *effMap* approach shows the energy for executing our approach as presented in this paper. The *savings* curve in the figure shows the relative energy consumption of our approach in comparison to the basic approach. This comparison considers the savings from the point in time when our approach starts to influence the mapping.

We see that up to 100 traces, both approaches consume the same amount of energy. The reason for this lies in the quality of the floor plan. When using less then 100 traces, no area can be marked as AM-area. Hence, there is no chance to turn off INS and to save energy. However, with more than 100 traces, the energy consumption diverges. At this point, our approach starts to influence the mapping by temporarily turning off INS. As a result, each additional trace saves energy in comparison to the basic approach. Considering the relative energy savings with respect to this point in time, our approach saves up to 25% of energy compared to the basic approach.

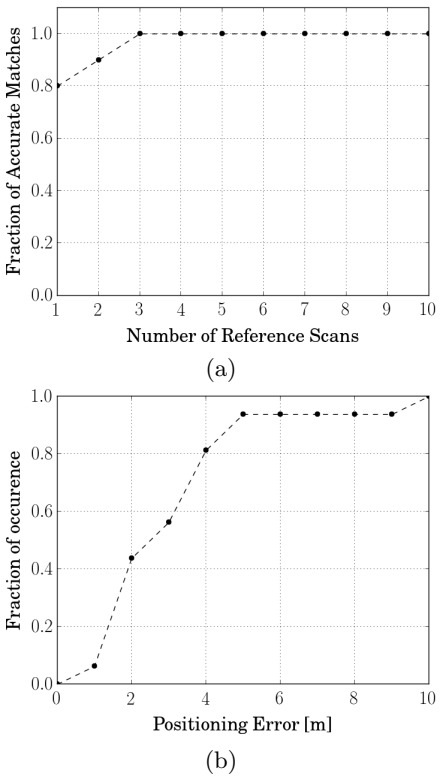

Figure 8: Positioning Accuracy: (a) Accuracy of anchor point identification. (b) CDF of shift errors when resuming INS.

9.4 Model Quality

Next, we have a look on how our approach impacts the quality of the generated floor plan. To evaluate the quality of

[1] see `http://www.comnsense.de/downloads`

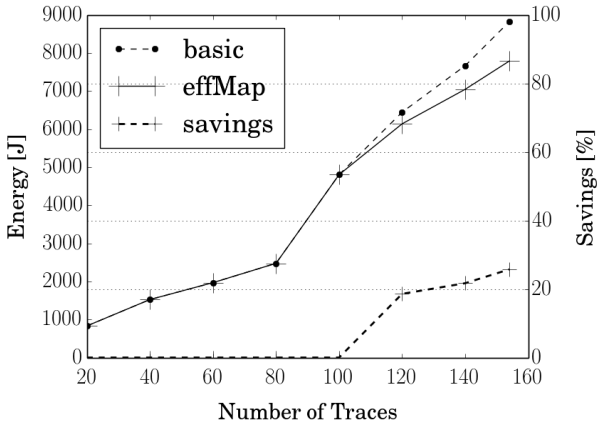

Figure 9: Total energy consumption of all devices.

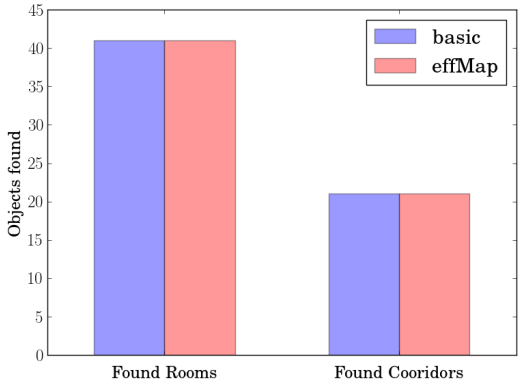

Figure 10: Rooms and corridors that were found.

the floor plan, we compared the plan generated by the basic approach and the plan generated by our approach against the ground truth floor plan. Figure 10 shows how many of the rooms and corridors from the ground truth floor plan were found in the respective approach. More precisely, we compare the position of the rooms and corridors in the generated plan and the ground truth plan and check how many of them are at the same position. We see that our approach finds the same number of rooms and corridors as the basic approach.

Furthermore, we investigated how big the area error for the derived rooms is, i.e., how different are the derived room sizes from the ones in the ground truth. The area error for our approach, averaged over all rooms, is 10.9 m. In comparison, the error for the basic approach is 10.5 m. If we have a look on how much the door position of rooms is shifted compared to the ground truth model, we get 3.5 m for our approach and 2.5 m for the basic approach. Hence, in our approach the room area error is on average only insignificantly bigger and the entrance of the rooms are shifted on average by 1 m. This can be explained by the shifted position obtained from the INS initialization algorithm (see Fig. 8b), since a shifted trace also results in a shifted room detection. However, further improvements of the INS initialization algorithm could help to reduce this room shifting error.

10. SUMMARY AND FUTURE WORK

In this paper, we presented an approach to reduce the energy consumption of automatic indoor mapping algorithms by collecting mobility traces only in areas where it is necessary to improve map quality. To this end, we proposed a quality metric to distinguish already accurately mapped areas requiring no further traces and so far inaccurately mapped areas where traces should be collected. This information is used by a scheduler to adaptively activate and deactivate trace collection by mobile devices. Finally, we showed how precise but energy-intensive INS can be combined with coarse but energy-efficient WiFi position recovery, to get a combined system where INS can be turned on on demand. Our evaluation showed that using our approach, we can save up to 25% energy without compromising on the quality of the generated floor plan.

In future work, we will extend the scope of our work to 3D indoor mapping. To this end, we need to decide which areas should be captured by camera images and, again, when positioning can be turned off. However, this requires new concepts also taking into consideration the energy-intensive processing of images and 3D data. While the quality model has to be extended to handle also 3D models, we also plan to consider the distribution of 3D processing between devices and/or servers.

Acknowledgements

This work is supported by the German Research Foundation (Deutsche Forschungsgemeinschaft, DFG), grants FR 823/25-1 and RO 1086/17-1, and the Ministry Of Science, Research and the Arts Baden-Württemberg.

11. REFERENCES

[1] M. Angermann and P. Robertson. Footslam: Pedestrian simultaneous localization and mapping without exteroceptive sensors – hitchhiking on human perception and cognition. *Proceedings of the IEEE*, 100:1840–1848, 2012.

[2] P. Baier, H. Weinschrott, F. Dürr, and K. Rothermel. MapCorrect: Automatic Correction and Validation of Road Maps Using Public Sensing. In *Proceedings of the Conference on Local Computer Networks (LCN)*, 2011.

[3] J. Eberle, Z. Yan, and K. Aberer. Energy-efficient opportunistic collaborative sensing. In *Proceedings of the Conference on Mobile Ad-Hoc and Sensor Systems (MASS)*, 2013.

[4] J. Hallberg, M. Nilsson, and K. Synnes. Positioning with bluetooth. In *Proceedings of the Conference on Telecommunications*, 2003.

[5] S. Haner and A. Heyden. Optimal view path planning for visual slam. In *Lecture Notes in Computer Science*, 2011.

[6] Y. Jiang, Y. Xiang, X. Pan, K. Li, Q. Lv, R. P. Dick, L. Shang, and M. Hannigan. Hallway based automatic indoor floorplan construction using room fingerprints. In *Proceedings of the Joint Conference on Pervasive and Ubiquitous Computing (UbiComp)*, 2013.

[7] A. Kerin. Behind the sixth sense of smartphones: the snapdragon processor sensor engine, Apr. 2014.

[8] F. Li, C. Zhao, G. Ding, J. Gong, C. Liu, and F. Zhao. A reliable and accurate indoor localization method

using phone inertial sensors. In *Proceedings of the Conference on Ubiquitous Computing (UbiComp)*, 2012.

[9] H. Liu, Y. Gan, J. Yang, S. Sidhom, Y. Wang, Y. Chen, and F. Ye. Push the limit of wifi based localization for smartphones. In *Proceedings of the Conference on Mobile Computing and Networking (MobiCom)*, 2012.

[10] D. Mendez, A. Perez, M. Labrador, and J. Marron. P-sense: A participatory sensing system for air pollution monitoring and control. In *Proceedings of the Workshops of Pervasive Computing and Communications (PERCOM Workshops)*, 2011.

[11] A. Moustafa and Y. Moustafa. Crowdinside: Automatic construction of indoor floorplans. In *Proceedings of the Conference on Advances in Geographic Information Systems (GIS)*, 2012.

[12] L. Ni, Y. Liu, Y. C. Lau, and A. Patil. LANDMARC: Indoor location sensing using active RFID. In *Proceedings of the Conference on Pervasive Computing and Communications (PerCom)*, 2003.

[13] Y. Noh, H. Yamaguchi, U. Lee, P. Vij, J. Joy, and M. Gerla. Clips: Infrastructure-free collaborative indoor positioning scheme for time-critical team operations. In *Proceedings of the Conference on Pervasive Computing and Communications (PerCom)*, 2013.

[14] I. Oksar. A bluetooth signal strength based indoor localization method. In *Proceedings of the Conference Conference on Systems, Signals and Image Processing (IWSSIP)*, 2014.

[15] D. Philipp, P. Baier, C. Dibak, F. Dürr, K. Rothermel, S. Becker, M. Peter, and D. Fritsch. MapGENIE: Grammar-enhanced Indoor Map Construction from Crowd-sourced Data. In *Proceedings of the Conference on Pervasive Computing and Communications (PerCom)*, 2014.

[16] D. Philipp, J. Stachowiak, P. Alt, F. Dürr, and K. Rothermel. DrOPS: Model-Driven Optimization for Public Sensing Systems. In *Proceedings of the Conference on Pervasive Computing and Communications (PerCom)*, 2013.

[17] D. Philipp, J. Stachowiak, F. Dürr, and K. Rothermel. Model-Driven Public Sensing in Sparse Networks. In *Proceedings of the Conference on Mobile and Ubiquitous Systems: Computing, Networking and Services (MobiQuitous)*, 2013.

[18] B. Priyantha, D. Lymberopoulos, and J. Liu. Eers: Energy efficient responsive sleeping on mobile phones. In *Intl. Workshop on Sensing for App Phones (PhoneSense)*, 2010.

[19] V. Radu, L. Kriara, and M. Marina. Pazl: A mobile crowdsensing based indoor wifi monitoring system. In *Proceedings of the Conference on Network and Service Management (CNSM)*, 2013.

[20] R. Ritchie. Apple m7, Apr. 2014.

[21] G. Shen, Z. Chen, P. Zhang, T. Moscibroda, and Y. Zhang. Walkie-markie: Indoor pathway mapping made easy. In *Proceedings of the USENIX Conference on Networked Systems Design and Implementation (NSDI)*, 2013.

[22] H. Shin, Y. Chon, and H. Cha. Unsupervised construction of an indoor floor plan using a smartphone. *IEEE Trans. Syst., Man, and Cybernetics, C: Applicat. and Reviews*, 42:889–898, 2012.

[23] H. Wang, S. Sen, A. Elgohary, M. Farid, M. Youssef, and R. R. Choudhury. No need to war-drive: Unsupervised indoor localization. In *Proceedings of the Conference on Mobile Systems, Applications, and Services (MobiSys)*, 2012.

[24] H. Weinschrott, F. Dürr, and K. Rothermel. Efficient Capturing of Environmental Data with Mobile RFID Readers. In *Proceedings of the Conference on Mobile Data Management (MDM)*, 2009.

[25] H. Weinschrott, J. Weisser, F. Dürr, and K. Rothermel. Participatory Sensing Algorithms for Mobile Object Discovery in Urban Areas. In *Proceedings of the Conference on Pervasive Computing and Communications (PerCom)*, 2011.