Robust Probabilistic Positioning based on High-Level Sensor-Fusion and Map Knowledge

Technical Report No. 421, April 18, 2003

Jürgen Bohn and Harald Vogt

Institute for Pervasive Computing
Swiss Federal Institute of Technology
ETH Zurich, Switzerland
{bohn|vogt}@inf.ethz.ch

Abstract. Location information as a piece of context is considered to be of particular interest to ubiquitous computing applications. In this work, we present a probabilistic positioning service that employs an available ubiquitous computing infrastructure for the localization of mobile devices. The service draws on various heterogeneous sensors that serve as sources of location information. Data from these sources are transformed independently of each other into an abstract representation of location estimates. By means of a probabilistic fusion process, these estimates are then combined into a single position value. The quality of the result increases with the number of available sensors. Also, since the fusion procedure is not depending on the type and number of available sensors, failures of single sensors or sensor types can be tolerated.

1 Introduction

Thanks to advancements in the field of miniaturization of computer technology and mass production, many ubiquitous computing applications that once bore the stigma of utopia have become technically feasible today. In the meantime, ubiquitous computing technology has become increasingly wide-spread in public places and even in private homes. For instance, wireless communication infrastructures such as Wireless LAN (WLAN) or Bluetooth have become quite popular in airports, train stations, and public buildings, and not only as substitutes for expensive wired networking [10, 18]. Simultaneously, other typical pervasive computing technologies such as radio frequency identification (RFID) have become commonplace in industrial and commercial facilities, e.g. in manufacturing plants and department stores [22].

Considering emerging ubiquitous computing applications, context awareness has become a major field of research [5]. Here, the general idea is that a broad range of applications may benefit from the capability of responding to their current situation and to the prevailing circumstances [8, 19]. Location awareness is of particular interest, and location information has been identified to remain "the single most important piece of context used in ubicomp applications" [1]. As a consequence, numerous location aware systems [13] have been conceived, e.g. the Lancaster mobile tourist guide [6] or various indoor navigation systems [23, 25].

While outdoor positioning systems are mainly based on cellular networks and satellite-based technologies [16], the majority of indoor location systems usually relies on different technologies, usually of a single kind, e.g., on Wireless LAN signal strength measurements [3], video analysis [23], infrared beacons [25], or ultrasonic sound [28]. As typical outdoor positioning technologies such as GPS are not suitable for indoor usage, and vice versa, location and research in the field of indoor and outdoor positioning systems is generally conducted separately.

However, a positioning system that is solely based on one technology has various disadvantages. On the one hand, it is prone to service disruption and interferences, because the unavailability or failure of the single underlying technology leads to a complete failure of the service as a whole. Therefore, the dependence on dedicated hardware is a major drawback of existing positioning systems, especially if they are to be deployed in environments with stringent reliability and availability demands such as hospitals, for example.

On the other hand, current location systems do not fully exploit the sources of location information that are already available in existing ubiquitous computing environments. Thus they are not able to take advantage of various new sources of location information that are – as a side-effect – implicitly provided by the infrastructure. While technologies such as RFID, WLAN or Bluetooth may have been invented with a certain primary application in mind, e.g. to be used for object identification or wireless data transfer, these technologies often leak location information during operation. Furthermore, the majority of existing location systems usually require special dedicated hardware, e.g. grids of customized infrared or ultrasonic beacons, GPS receivers, etc., leading to additional costs.

In this work, we present a robust and scalable probabilistic positioning service which addresses the described shortcomings of common ubiquitous computing location systems. The service is based on high-level sensor fusion and makes use of symbolic information represented in maps. Its architecture is modularized and supports an arbitrary number of sensors to be integrated.

The remaining sections of this paper are organized as follows: In the next section, we explain how sensor fusion allows us to take advantage of various available sources of location information, thus contributing to increase the overall robustness and availability of the positioning service. We also discuss the benefits of using a grid based map model as part of the positioning procedure. In Section 3, we describe the overall architecture and the main components of a probabilistic positioning algorithm that uses map knowledge and high-level sensor fusion to increase the accuracy and plausibility of its positioning results. Then, in Section 4, we discuss how the positioning service addresses various dependability problems. Afterwards, we give an overview of related work in Section 5. Finally, in Section 6 we conclude with a short summary and give an outlook on future work.

2 Fundamentals

There are many systems available that use dedicated hardware infrastructures for localization. The most widely used system today is the Global Positioning System (GPS),

which is restricted to outdoor localization. Since the use of dedicated hardware is often very costly, both for the infrastructure and dedicated components in end-user devices, it is desirable to use features of already existing installations for localization. Communication technology such as wireless networks (WLAN, Bluetooth) or identification systems (RFID, barcodes) come in handy for that purpose. Our goal is the exploitation of this existing infrastructure for determining the location of objects. Features of wireless communication such as signal propagation time or signal strength are promising candidates. Identification systems where the location of one component is known (such as a stationary RFID reader) provide us with immediate position information.

2.1 High-Level Sensor-Fusion

Basically, there are three possible levels on which to perform sensor fusion [11]: on raw sensor data, on features extracted from raw data, and on the decision level.

Fusion on raw sensor data is only possible if the domain of all sensors is the same, i.e. they are of the same type and measure the same quantity. In our approach, this would be possible, but the fusion process would be located within a module representing a sensor and the outcome of that process would be regarded as the measurement of a single sensor. (As an example, consider multiple RFID tags located on the same object. Each tag, when recognized by a RFID reader, would provide a location of the object. But since this location is the same for all tags—the location of the RFID reader—the locations can simply be collapsed into one.)

Feature extraction is a technique that reduces the amount of data produced by a sensor and abstracts away all information that is irrelevant for the task at hand—in case of a positioning system, only information relevant to determining the current location is retained. Multiple sensors (working on different domains) can be combined after relevant features have been extracted from the raw data. On the one hand, feature extraction for RFID tags makes no sense, since RFID tags immediately yield an object's location. Therefore, data obtained from RFID tags and WLAN data cannot be fused on the feature level. On the other hand, for data obtained from Bluetooth and WLAN receivers, e.g. signal strength information, feature level fusion may be suitable. In its current state, our system does not support data fusion on the feature level directly, since no component for feature extraction and combination is incorporated. However, data fusion on the feature level can again be accomplished in the sensor modules themselves.

Currently, sensor fusion is performed mainly on the decision level, i.e. each sensor module provides the system with a set of possible values (object locations) represented as a probability distribution. These distributions are combined to compute a new probability distribution that represents the most likely location of the object.

This approach, sensor fusion at the decision level, facilitates a modular and extensible system architecture. The number and types of sensors are not limited. Processing the sensor data can be performed remotely (i.e., not on the object itself) and pushed to the object in the form of an internal location event. When single sensors fail (or are shut off due to power saving efforts), the quality of the localization is affected, but the system as a whole remains functional. If the quality of the positioning dropped to room-level accuracy, for instance, it would still serve applications for which it is sufficient to get

regular updates on the position whenever the user enters another room, being indifferent to all intermediary positions of the user.

2.2 Map Knowledge

The existence of a map model is an important aspect in our approach. The map model serves two main purposes. First, it provides a frame of reference within which all sensor data is interpreted. Second, it provides applications with symbolic and sub-symbolic location information, e.g. "near table" or "at coordinates (x, y) in room Z", respectively.

In our system, a map is internally represented as a 2-dimensional grid with a fixed cell size where the cell size can be chosen individually for each map. A cell is represented by a data structure, which contains the following information:

- a value for the probability that the tracked object is located within this cell;
- probability values for movement into the eight adjacent cells, and for staying in the cell.

There are also cells of a special type (inherited from the standard type) that mark the transition to another map; this allows switching to other maps when the object leaves the area of the current map.

The map model is used throughout the localization process. In a learning phase, symbolic locations are linked to map coordinates. During usage of the positioning, map knowledge is exploited in various ways, e.g. obstacles are considered when recomputing the position probabilities.

When no location update events arrive, the position gets increasingly blurred. This is reflected by the spreading of the probability distribution, i.e. the number of cells with a non-zero probability increases, but the probabilities for single cells are lowered. The spreading is guided by movement patterns (which can be acquired in a preprocessing stage) and map topology. For example, walls and static objects like furniture that are recorded in the map designate areas where the object simply cannot move, therefore the spreading stops at these points.

When a new position is to be determined, the outcome of the positioning process generally is a list of locations, ordered by probability. However, map knowledge can be exploited in order to eliminate certain points from that list, e.g. if a point is located outside of a building.

3 Probabilistic Positioning Service

The positioning system has been designed to combine the location information extracted from various sensors. It is capable of running self-sufficiently as a stand-alone service on the mobile device itself, only drawing upon the locally available sensors. However, the positioning service is also able to take advantage of sensory information provided by a surrounding infrastructure, if appropriate.

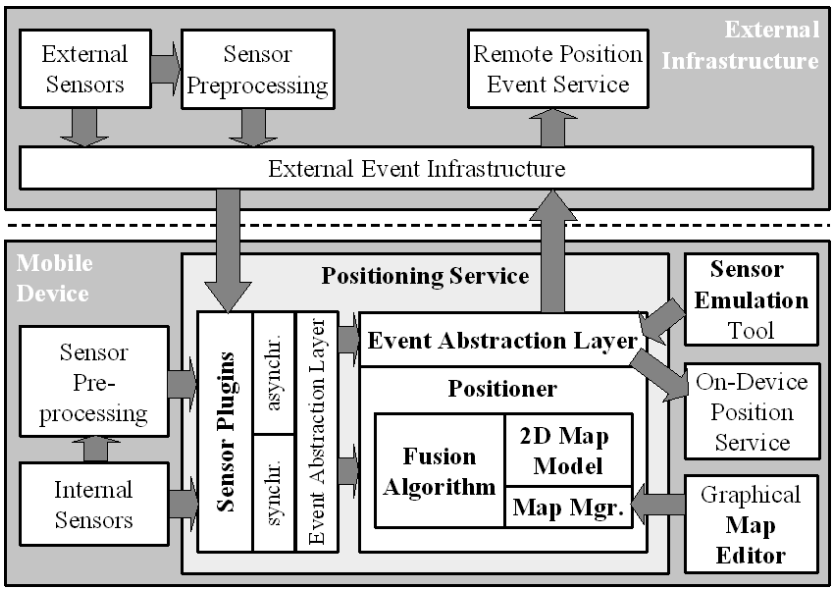

Fig. 1. Architecture of the probabilistic positioning service

3.1 Architecture

The implemented positioning system consists of several separate components as shown in Figure 1. The main components are:

- **Sensor Plugins**
  - Asynchronous plugins (push semantics)
  - Synchronous plugins (pull semantics)
- **Event Abstraction Layer**
  - Event buffer
  - Sensor events
  - Position events
- **Positioner**
  - Geographic map model (cell grid)
  - Map-based fusion algorithm
  - Map manager
- **Management Tools**
  - Graphical map builder
  - Graphical sensor emulation tool

In the following, we will refer to a *sensor* as an internal or external source of location information. In our context a sensor may be either a physical device or an application that in turn preprocesses the location information of other physical devices or applications. A *sensor plugin* is associated with one specific type of sensor. It preprocesses

and transforms sensory location information into an abstract representation of position estimates we call *sensor events*. We also differentiate between synchronous (active) and asynchronous (passive) sensor plugins. A *synchronous* sensor returns a position estimate whenever it is prompted to do so, whereas *asynchronous* sensors provide position estimates at irregular intervals only because their functioning is decoupled from the operation of the other components. For instance, Wireless LAN position estimates based on the received signal strength are actively triggered by a synchronous sensor plugin, while the scanning of a barcode or the detection of an RFID tag are passively monitored by an asynchronous sensor plugin. Since asynchronous sensor readings may occur at unforeseen times, these asynchronous sensor events are stored temporarily by means of an internal *event buffer*.

The central system component is the *positioner*, a separate module which performs the actual positioning computation. During each computational iteration step, the positioner calculates a single position by fusing the location information of the available sensor events. The fusion process is accomplished by means of a *probabilistic fusion algorithm* running on top of a two-dimensional map model. The fusion process will be explained in detail in Section 3.3. Map models represent geographic maps and are realized using an equidistant *cell grid*. The cell grid may contain obstacles (or walls) and arbitrary objects, e.g. markers for symbolic locations or sensors (see also Section 2.2). The cell grid data structure also offers a range of auxiliary methods (e.g., to access and modify cells or to resolve the position of objects) and advanced methods (e.g., to calculate the decay of occupancy probabilities of single cells or to compute the minimal distance between two cells with respect to blocking walls and obstacles). The *map manager* unit is responsible for the loading of maps and the updating of position probabilities in cell grid data structures.

Whenever a position computation has completed, the resulting position estimate is converted into a *position event* and fed into the external event infrastructure. Additionally, the position event is written into a local event buffer which is readable to applications and services running on the mobile device.

In order to create and edit maps, a graphical *map builder* has been implemented. The map builder provides the means to place and name objects, to draw walls and to insert transition cells to other maps, for instance. The editor has been enhanced by a simulation mode that allows the manual *emulation of sensor events* which proved suitable for preliminary testing of new heuristics and algorithms for positioning, as well as for the optimization of configurational parameters.

3.2 Sensor Events

All position information that is received from the sensor plugins is translated into an abstract internal format of absolute positions with respect to a given two-dimensional map model.

Depending on the sensor type, position estimates vary in terms of accuracy. In our system, we discern two categories of sensor events, exact and fuzzy sensor events. In the grid map model, an *exact point* sensor event (in short: exact sensor event) specifies a single cell which is considered to provide an exact match of the current position; i.e., the position probability of that cell yields one hundred percent. As a consequence,

exact sensor events can be exploited for immediate position updates. In contrast, a *fuzzy area* sensor event (in short: fuzzy sensor event) only specifies a certain region on the map, i.e. a collection of cells on the grid, with a certain characteristic distribution of position probabilities. Figuratively speaking, spacial sensor events define a fuzzy cloud of position probabilities, allowing the uncertainty of position information to be modeled with respect to the accuracy and precision of a single sensor.

The dispersion of the position probabilities for fuzzy events is performed by a parameterized *spreading function*, which is specific to each sensor class. Let a fuzzy sensor S_{f} return a symbolic location whose anchor point is located on the map at cell x_j. This cell receives the position probability value of $P(X = x_j | S_{\mathrm{f}} = s_k)$ with respect to the reading s_k of a single sensor. Now the spreading function for S_{f} also assigns position probability values $P(X = n_i | S_{\mathrm{f}} = s_k)$ to nearby cells n_i up to a specified maximum distance $d_{\max}$, so that $\mathrm{distance}(x_j, n_i) \leq d_{\max}$ holds. Depending on the characteristics of a sensor, the spreaded probability values for neighboring cells can be chosen according to a linear, exponential or any other (customized) mathematical function of distance d. At the moment, a linear and an exponential spreading function have been implemented.

3.3 Probabilistic Fusion Algorithm

The central system component of the positioning service, the *positioner*, runs in a separate thread, gathering sensor information and recalculating the position of the mobile device in a continuous loop. A single run consists of several computational iteration steps, during which the positioner calculates a single position. This is achieved by fusing the location information of the available sensor events into a single position estimate. For that, the positioner continuously collects available sensor location information. The actual fusion process is accomplished by means of a *probabilistic fusion algorithm* running on top of a two-dimensional map model. Figure 2 gives an overview of the different stages and procedures of the positioning algorithm that is described in this Section. In the following, for sake of clarity, we will use the term *current position* exclusively to refer to the last calculated position of the mobile device.

Sensor Data Acquisition. In the initial phase of the positioning algorithm, position estimates are acquired from the available sensor plugins for a well-defined period of time. In doing so, synchronous sensor plugins are actively inquired at regular intervals (e.g., every second) and the obtained location information is processed (pull semantics). Asynchronous sensor input is preprocessed in a separate thread as soon as the data has been received (push semantics).

At the end of the data collection phase, the obtained sensor events are analyzed. Depending on the types of the received sensor events, there are two separate processing sequences, the immediate update and the fusion-based position derivation sequence.

a) Immediate Position Update Procedure. The *immediate update* procedure is chosen if at least one exact sensor event was received in the previous phase. First, the location of the latest exact sensor event is resolved on the map and instantly taken as the new

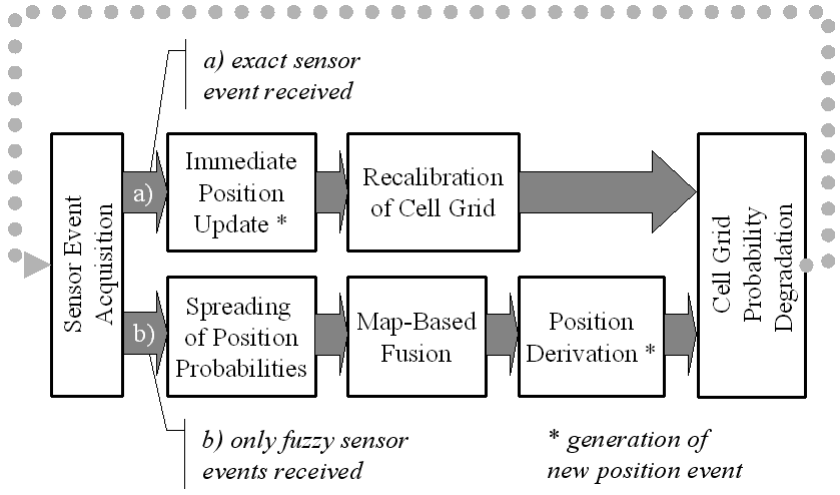

Fig. 2. Probabilistic sensor event fusion algorithm

current position of the mobile device. Thus the costly fusion-based position derivation procedure can be bypassed, resulting in an efficient (and in this case deterministic) position update. In a second step, the position probabilities in the grid map model are recalibrated: the position probability of the cell corresponding to the current location is marked with a one hundred percent probability, whereas the position probabilities of all other cells are reset to zero.

b) Fusion-Based Position Derivation Procedure. If merely fuzzy sensor events were received but no exact ones, the *fusion-based position derivation* procedure is executed: In the beginning, the position probability (cell occupancy probability) for each fuzzy sensor event is spread over a certain number of cells in the cell grid model, respecting obstacles and walls modeled as part of the grid.

While the spreading process of the individual sensor event probabilities is performed in compliance with the well-defined characteristics of the respective sensor type, the probability values that are distributed for each sensor event are accumulated and overlaid. The repeated execution of the spreading function on top of the same grid map data structure yields a *fusion of the individual sensor event probabilities*. The fused sensor event probabilities are then merged with the existing position probabilities stored in the cell grid, which is followed by a global probability normalization covering all active maps. The result is a common probability distribution for the current position described by probability value X and n sensors $S_1, \ldots, S_n$. (Please note that n is not an absolute number; instead, its value varies from execution to execution, depending on the number of available sensors that provide sensor events for the current processing cycle.) This

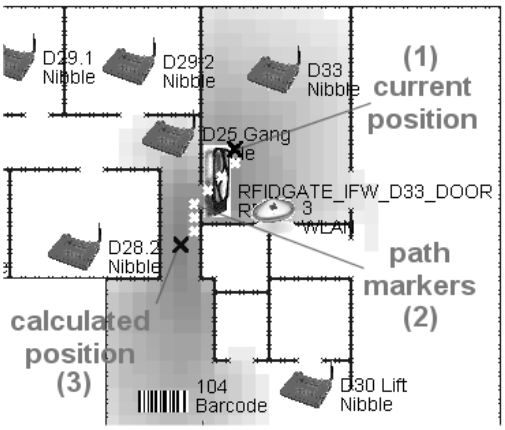

Fig. 3. Visualization of position probability distribution in cell grid during operation. The black x marks the last known position of the mobile device (1). Darker shaded areas on the map correspond to higher position probabilities. The sequence of white markers (2) defines the shortest path to the new position (3) as calculated by the positioning service

probability distribution is defined by Equation 1,

$$P(X|\bigwedge_{k=1,\ldots,n} S_k) := P(X = x|\bigwedge_{k=1,\ldots,n} S_k = s_k), \tag{1}$$

where $P(X|\bigwedge_{k=1,\ldots,n} S_k)$ equals the probability that cell x is the current position, given that each sensor S_k evaluates to the value s_k, with $k = 1, \ldots, n$. In the following, we will use a similar short notation of the kind depicted in Equation 2 whenever appropriate:

$$P(X|S) := P(X = x|S = s) \tag{2}$$

A metaphorical interpretation of the probability distribution defined in Equation 1 is a cloud of probability values covering the cells in the grid map: the denser the cloud at a certain cell is, the higher is the probability that the device is currently located in that spot (see example in Figure 3).

In order to solve Equation 1, we first apply the rule of conditional probabilities

$$P(A|B) = \frac{P(A \wedge B)}{P(B)}, \tag{3}$$

and we get

$$P(X|S_1 \wedge S_2 \wedge \cdots \wedge S_n) = \frac{P(X \wedge S_1 \wedge S_2 \wedge \cdots \wedge S_n)}{P(S_1 \wedge S_2 \wedge \cdots \wedge S_n)}. \tag{4}$$

.

By applying Equation 3 once more to the numerator of Equation 4, we obtain

$$P(X|S_1 \wedge S_2 \wedge \cdots \wedge S_n) = \frac{P(S_1 \wedge S_2 \wedge \cdots \wedge S_n|X) \cdot P(X)}{P(S_1 \wedge S_2 \wedge \cdots \wedge S_n)}. \tag{5}$$

Let $S_1, \ldots, S_n$ be n statistically independent sensors,

$$\forall i \, \forall j \; P(S_i \wedge S_j|X) = P(S_i|X) \cdot P(S_j|X), \tag{6}$$

then Equation 5 can be transformed into

$$P(X|S_1 \wedge S_2 \wedge \cdots \wedge S_n) = \frac{P(S_1|X) \cdot P(S_2|X) \cdot \ldots \cdot P(S_n|X) \cdot P(X)}{P(S_1 \wedge S_2 \wedge \cdots \wedge S_n)}. \tag{7}$$

Let us now define a, s, p as short notations for the following probabilities:

$$\begin{aligned} a(s_1, s_2, \ldots s_n) &:= P(S_1 = s_1 \wedge S_2 = s_2 \wedge \cdots \wedge S_n = s_n) \\ p(x) &:= P(X = x) \\ s(s, x) &:= P(S = s|X = x) \end{aligned} \tag{8}$$

Thus we can rewrite Equation 7 as

$$P(X|S_1 \wedge S_2 \wedge \cdots \wedge S_n) = \frac{s(s_1, x) \cdot s(s_2, x) \cdot \ldots \cdot s(s_n, x) \cdot p(x)}{a(s_1, s_2, \ldots s_n)}. \tag{9}$$

This last expression can be resolved, because we can determine the value of all terms on the right-hand side of Equation 9: First, $s(s, x)$ is the probability that sensor S returns value s at the known position x; these values may be influenced by external conditions such as time, network load, etc., and need to be determined a-priori for each sensor, e.g. by means of calibration, initial learning, or mathematical model. [1] Second, $p(x)$ is the probability that cell x is our current position in the map model, based on the position probabilities calculated during earlier grid-based fusion processes. Third, the term a can be determined indirectly, using the theorem of total probabilities: Let $A_i, \ldots, A_n$ be n mutually exclusive events that form a partition of the event space, $A_i \cap A_j = \emptyset$ for $i \neq j$ and $\bigcup_i A_i = \Omega$, then their probabilities sum to unity:

$$\sum_{i=1,\ldots,n} P(A_i) = 1 \tag{10}$$

This allows us to determine the missing term $a(s_1, s_2, \ldots s_n)$ by solving the following equation:

$$\begin{aligned} \sum_i \frac{s(s_1, x_i) \cdot s(s_2, x_i) \cdot \ldots \cdot s(s_n, x_i) \cdot p(x_i)}{a(s_1, s_2, \ldots s_n)} &= 1 \\ \frac{1}{a(s_1, s_2, \ldots s_n)} \sum_i s(s_1, x_i) \cdot s(s_2, x_i) \cdot \ldots \cdot s(s_n, x_i) \cdot p(x_i) &= 1 \end{aligned} \tag{11}$$

[1] A varying number of people in a building, for instance, is known to cause considerable variations of radio frequency signal propagation, thus interfering with signal strength based positioning systems in a Wireless LAN environment [3].

By transformation of Equation 11, the term $a(s_1, s_2, \ldots s_n)$ can be isolated:

$$a(s_1, s_2, \ldots s_n) = \sum_i s(s_1, x_i) \cdot s(s_2, x_i) \cdot \ldots \cdot s(s_n, x_i) \cdot p(x_i) \qquad (12)$$

As a result, all right-hand side terms of Equation 7 are known, and Equation 1 is solved.

Now the final step of the fusion-based position derivation procedure, after the fusion procedure has terminated and the probability distribution on the cell grid has been updated, is the determination of the cell which is the best fitting choice for the new current position of the mobile device. This *position derivation* is performed by means of a heuristic which reduces the cell search space to one single cell. The heuristic searches for the cell with the highest position probability on the currently active grid map(s) within a maximum distance of the cell representing the last known device position. The heuristic includes plausibility checks that reduce the search space to positions that can actually be reached within the 2-dimensional model (i.e., by considering walls and a maximum movement rate), starting from the current position.

To prevent great leaps between subsequently calculated positions, the system allows to set an average motion speed which is then used to compute intermediate positions that gradually approach the latest position derived by the fusion algorithm. For instance, the black marker in Figure 3 does not jump to the newly calculated position. Instead, it gradually approaches the new position at the specified movement rate, e.g. matching the walking speed of the person carrying the device, thus smoothing the transition between calculated positions.

$n_{2,j}$	$n_{3,j}$	$n_{4,j}$
$n_{1,j}$	x_j $= n_{0,j}$	$n_{5,j}$
$n_{8,j}$	$n_{7,j}$	$n_{6,j}$

Fig. 4. Cell x_j with neighboring cells $n_{i,j}$ $(0 \leq i \leq 8)$

Cell Grid Probability Degradation. At the end of either event processing procedure, for all cells in the cell grid a defined *decay function* is calculated to account for the gradual increase of position uncertainty over time (if no new probabilities are added). Let X_t be the position at time t and $n_{i,j}$ $(0 \leq i \leq 8)$ the neighboring cells of cell x_j (see Figure 4). Then the decay function is applied to the position probability of each

cell x_j as described in Equation 13. $P(X_t = x_j | X_{t-1} = n_{i,j})$ is the probability of a transition from the neighboring cell $n_{i,j}$ at time $t-1$ to cell x_j at time t.[2]

$$P(X_t = x_j) := \sum_i (P(X_{t-1} = n_{i,j}) \cdot P(X_t = x_j | X_{t-1} = n_{i,j})) \qquad (13)$$

This *probability degradation* ensures that position probabilities of earlier probability fusion calculations wear off over time: all cells gradually spread their position probabilities over adjacent cells if they are not allotted additional probability values during subsequent processing loops.

Initially, the transition probabilities for each cell are defined as follows: let w be the number of neighboring cells (including the cell itself) that can be reached, then the transition probability for these transitions is equally set to $\frac{1}{w}$, favoring no particular direction. For all remaining directions, e.g. in case the neighboring cell is blocked by a wall or does not exist, the transition probability is set to zero. Later, these cell transition probabilities can be adjusted according to characteristic personal movement patterns acquired during a learning phase, for example.

4 Discussion

We have presented a positioning system for ubiquitous computing that emphasizes the robustness of the core functionality, namely self-localization of a "smart object" by means of sensory input. The system architecture allows for simple integration of multiple sensors due to the loose coupling between the acquisition of sensory data and the positioning algorithm by means of an event-based communication mechanism. Previously unknown (or returning) modules for sensors can simply fire events to trigger an update of the positioning.

The positioning service is capable of operating in a decentralized and self-sufficient manner. Since it is designed to incorporate a wide variety of sensors, it can tolerate the failure of large subsets of them. The functioning of the service is not interrupted in the presence of frequent sensor interruptions. This feature can even be exploited to increase battery lifetime of mobile devices by switching off certain sensors if they don't contribute to the required sensor quality.

Due to the probabilistic nature of the position algorithm and the high-level sensor fusion, sensor failure leads to graceful degradation of the quality of the position information. The position fusion algorithm is robust in terms of tolerating the failure of single sensors. The absence of sensors only affects the quality of the positioning, but as long as there is some sensory input available, the positioning algorithm stays "alive" and continues to generate position estimates, maximizing the *availability* of the service: If a highly accurate sensor failed and the overall accuracy of position information dropped from cell level to room level, applications that are satisfied with rough position estimates would still be fully functional. Additionally, even in case all sensory

[2] The use of discrete time values in the equations has been motivated by the fact that position events are generated at particular, discrete moments. Alternatively, the notion "time t" could also be replaced with "position event t".

input temporarily fails to appear, movement heuristics and dead reckoning techniques can be applied to get a rough picture of the location of the object, e.g. by counting the number of footsteps of a walking person [26]. Alternatively, the momentary absence of all sensory input can be compensated by deploying heuristics for a short-time position prediction.

The system is also resilient to sensors providing inaccurate data: Location information gained from such sensors is mapped to fuzzy sensor events that, by defining a region rather than an exact position, reflect the corresponding uncertainty of position information. Even false sensor data or outliers that occur sporadically, e.g. due to transient interference, are tolerated by considering the reachability of positions and a maximum speed of movements while calculating a position from the current probability distribution.

By introducing more sensor modules, the quality of positioning results can be improved, at the cost that a potentially higher number of sensor data has to be processed. For synchronous sensors, the increase of sensor data depends on the frequency by which the system polls them for updates. Overall, the complexity of the position estimation process is *linear* in the number of sensor modules and sensor readings. Thus, *scalability* with regard to the number of sensors is maintained, and the system is open to quality improvement.

In many contexts, location information is considered as sensitive information [12]. Concerns about the *privacy* of such information are addressed by exclusively using local sensor information, that is such information which is passively available and requires no external requests. Thus, by solely relying on local sensors, the system can be configured not to introduce new channels by which location information about the device itself is revealed. However, in some cases such a policy is difficult to enforce due to technical restrictions: For instance, a node communicating over WLAN reveals its location (to a certain degree) to the base station, which cannot be prevented by our system.

The use of an existing communication and identification infrastructure instead of dedicated hardware for localization has the advantage of avoiding additional *cost* for installation and maintenance. However, our system can easily integrate a dedicated positioning infrastructure. This makes it suitable for environments where certain areas require high location resolution, accuracy and precision. By using a unified format for sensor events, our system supports a seamless transition between areas where different positioning technologies are available, including the transition between indoor and outdoor areas.

5 Related Work

An overview of (indoor) positioning systems for ubiquitous computing environments and their relation to robot positioning is given in [13]. There are systems based on dedicated hardware, such as cameras [7, 17, 23], infrared beacons [27], ultrasonic emitters [21, 28], pressure sensors [2, 20], and special hardware for dead reckoning [26]. Some effort has also been invested in systems that require no additional hardware to locate objects. The RADAR project [3] draws location information from the signal

strength of WLAN installations. In our system, we used basically the same approach. [1] uses biometric sensors and RFID for immediate location updates.

Our system provides applications which are executed on a "smart object" with location information about that same object. In contrast to this decentralized approach, the *Location Service* [1] is a centralized service providing access to location information about arbitrary entities. It supports a variety of sensors, but doesn't emphasize fusion of data acquired from these sensors. Rather, it seems to consider all sensor input as accurate, though possibly not being complete (e.g. lacking orientation information). By merging different sensor data, it generates complete location updates for tracked objects. The most significant difference to our work is the Location Service's global availability, i.e. it is built to supply all applications with location information, not only the tracked object itself. From a reliability point of view, such a centralized service constitutes a single point of failure, which is not acceptable for critical applications.

The *Location Stack* model proposed by Hightower et al. in [14] provides a multi-layered design abstraction for location-aware ubiquitous computing systems. The fusion of sensor information is described in general terms and attributed to one of the layers in the stack. Two exemplary systems are discussed where a customized data fusion process is used to merge the location information available from a specific set of sensors. In contrast, our work concentrates on the fusion aspect, combining an arbitrary number of sensors and using a scalable and flexible fusion algorithm that is independent of the number of sensors available at a certain point in time. In this context, however, integrating our proposed fusion algorithm with the *Location Stack* model could be an interesting option.

Data fusion for positioning in wireless networks is discussed in [16] with an emphasis on lower levels. Since that work is based on quantities like time of arrival and time difference of arrival, it is not directly applicable to indoor positioning. However, such techniques can be incorporated in our system in order to increase the accuracy of position information drawn from outdoor sensors.

Grid based positioning is a standard technique for robot positioning [4], but more advanced techniques have been developed more recently [9], requiring less computational power. While the basic mathematical techniques are applicable also in positioning for ubiquitous computing, the goal is quite different. In ubiquitous computing, positioning of people is prevalent, thus different sensors are used, and many applications don't require a high degree of precision (a precision of approx. one meter is often sufficient, or only symbolic location information is necessary).

6 Conclusion and Future Work

In this paper, we have presented a robust probabilistic positioning system that incorporates high-level sensor fusion and map knowledge to provide applications with accurate position information. We have discussed the importance of dependable positioning services for ubiquitous computing. As applications increasingly rely on the availability of accurate location information, it is important that positioning services stay functional in the presence of (partial) failures and degrade gracefully. We achieve this by the combi-

nation of heterogeneous sensing technology, based on an existing ubiquitous computing infrastructure, and by means of a decoupled system architecture.

The positioning service is using a grid model, which allows a straightforward implementation of the fusion process and provides a frame of reference for absolute positions. While the implemented system has been optimized to support devices the size of a PDA, the comparably high computational and memory complexity of our approach still constitutes a drawback for the use with even smaller devices, e.g. Smart-Its [15], that distinctly have stronger resource limitations. A further, implementation-specific limitation of our system is the use of a fixed cell size for a single map. We are currently investigating the use of variable cell sizes that would allow a more fine-grained resolution for specific areas of a map if needed, while avoiding too many cells in areas where applications are satisfied with less precise position information. By using variable cell sizes, we could also reduce memory requirements. Last but not least, we are in the process of conducting a number of experimental evaluations.

Future work on our system includes the integration of a greater variety of sensors and dedicated localization infrastructures, the latter allowing us to increase accuracy and precision of location information for specific areas. Another goal is the incorporation of heuristics for movement prediction. This would allow for further improvements on the quality of position estimates in the absence of sensor data.

7 Acknowledgments

This paper is based on Christian Schär's master's thesis "Heuristiken zur Positionsbestimmung in Gebäuden mittels Sensor-Fusion und 2D-Kartenmodell" [24].

References

1. Gregory D. Abowd, Agathe Battestini, and Thomas O'Connell. The Location Service: A Framework for Handling Multiple Location Sensing Technologies. www.cc.gatech.edu/fce/ahri/publications/location_service.pdf, 2002.
2. Michael D. Addlesee, Alan Jones, Finnbar Livesey, and Ferdinando Smaria. The ORL Active Floor. *IEEE Personal Communications*, 4(5):35–41, October 1997.
3. Paramvir Bahl and Venkata N. Padmanabhan. RADAR: An In-Building RF-Based User Location and Tracking System. In *INFOCOM 2000*, pages 775–784, 2000.
4. Wolfram Burgard, Dieter Fox, and Daniel Hennig. Fast Grid-based Position Tracking for Mobile Robots. In *Proc. of the 21th German Conference on Artificial Intelligence*, volume 1303 of *LNCS*. Springer-Verlag, 1997.
5. Guanling Chen and David Kotz. A survey of context-aware mobile computing research. Technical Report TR2000-381, Dept. of Computer Science, Dartmouth College, November 2000.
6. Nigel Davies, Keith Cheverst, Keith Mitchell, and Alon Efrat. Using and determining location in a context-sensitive tour guide. *IEEE Computer*, 34(8):35–41, August 2001.
7. Diego López de Ipiña. Vdeo-Based Sensing for Wide Deployment of Sentient Spaces. In *Proc. of 2nd PACT 2001 Workshop on Ubiquitous Computing and Communications*, September 2001.

8. Anind Dey, Gregory Abowd, and Daniel Salber. A conceptual framework and a toolkit for supporting the rapid prototyping of context-aware applications. *Human-computer interaction: a journal of theoretical, empirical, and methodological issues of user science and of system design*, 16(2–4):97–166, 2001.
9. D. Fox, W. Burgard, F. Dellaert, and S. Thrun. Monte Carlo Localization: Efficient Position Estimation for Mobile Robots. In *Proc. of the National Conferenceon Artificial Intelligence*. AAAI Press / The MIT Press, 1999.
10. Jack Glas, Mihai Banu, Vladimir Prodanov, and Peter Kiss. Wireless LANs. In *Proceedings of the Workshop on Advances in Analog Circuit Design*, March 2002.
11. David L. Hall and James Llinas, editors. *Handbook of Multisensor Data Fusion*. CRC Press, 2001.
12. Urs Hengartner and Peter Steenkiste. Protecting People Location Information. UBICOMP Workshop on Security in Ubiquitous Computing, 2002.
13. Jeffrey Hightower and Gaetano Borriello. Location Systems for Ubiquitous Computing. *IEEE Computer*, 34(8):57–66, August 2001.
14. Jeffrey Hightower, Barry Brumitt, and Gaetano Borriello. The location stack: A layered model for location in ubiquitous computing. In *Proceedings of the 4th IEEE Workshop on Mobile Computing Systems & Applications (WMCSA 2002)*, pages 22–28, June 2002.
15. Oliver Kasten and Marc Langheinrich. First experiences with bluetooth in the smart-its distributed sensor network. *Proc. of the Parallel Architecture and Compilation Techniques 2001*, Oct. 2001.
16. T. Kleine-Ostmann and A. E. Bell. A data fusion architecture for enhanced position estimation in wireless networks. *IEEE Communications Letters*, 5(8):343–345, 2001.
17. J. Krumm, S. Harris, B. Meyers, B. Brumitt, M. Hale, and S. Shafer. Multi-Camera Multi-Person Tracking for EasyLiving. In *Proc. of Third IEEE Workshop on Visual Surveillance*, 2000.
18. Stephen Lawson. Bluetooth may bring flood of in-building services. www.idg.net/idgns/2000/12/15/BluetoothMayBringFloodOfInBuilding.shtml, December 2000.
19. Giles John Nelson. *Context-Aware and Location Systems*. PhD thesis, Clare College, University of Cambridge, UK, January 1998.
20. Robert J. Orr and Gregory D. Abowd. The Smart Floor: A Mechanism for Natural User Identification and Tracking. Technical Report GIT-GVU-00-02, GVU, January 2000.
21. Nissanka B. Priyantha, Anit Chakraborty, and Hari Balakrishnan. The Cricket Location-Support System. In *Proc. of the Sixth Annual ACM International Conference on Mobile Computing and Networking (MOBICOM)*, August 2000.
22. RFID Journal. www.rfidjournal.com.
23. Wasinee Rungsarityotin and Thad Starner. Finding location using omnidirectional video on a wearable computing platform. In *Proceedings of IEEE International Symposium on Wearable Computing(ISWC 2000)*, pages 61–68, 2000.
24. Christian Schär. Heuristiken zur Positionsbestimmung in Gebäuden mittels Sensor-Fusion und 2D-Kartenmodell[3] (in German). Master's thesis, Institute for Pervasive Computing, Distributed Systems Group, Federal Institute of Technology Zurich (ETH), Switzerland, August 2002.
25. Yehuda Sonnenblick. An indoor navigation system for blind individuals. In CSUN Center On Disabilities, editor, *CSUN 1998 Conference, California State University Northridge*, Los Angeles, March 1998.

[3] Heuristics for positioning in buildings using sensor-fusion and a 2-dimensional map model. (translated by authors)

26. Elena Vildjounaite, Esko-Juhani Malm, Jouni Kaartinen, and Petteri Alahuhta. Location Estimation Indoors by Means of Small Computing Power Devices, Accelerometers, Magnetic Sensors and Map Knowledge. In *Pervasive 2002*, volume LNCS 2414 of *Lecture Notes in Computer Science*, pages 211–224, Atlanta, Georgia, USA, 2002. Springer-Verlag.
27. Roy Want, Andy Hopper, Veronica Falcao, and Jonathon Gibbons. The Active Badge Location System. *ACM Transactions on Information Systems*, 10(1), January 1992.
28. Andy Ward, Alan Jones, and Andy Hopper. A new location technique for the active office. *IEEE Personal Communications*, 4(5):42–47, October 1997.