Platform for Hybrid Positioning based on a Sensor Description Language

Moritz Kessel and Sascha Schreier
Mobile and Distributed Systems Group
Ludwig-Maximilians-University Munich
Munich, Germany
moritz.kessel@ifi.lmu.de

***Abstract*—While in the last years many different methods for indoor positioning have been presented, no single system has emerged yet that satisfies the requirements of the many different application scenarios. We propose to solve this problem with a platform for hybrid positioning which chooses the most suitable from all available sensor information for position estimation. For that task, a sensor description language is defined which allows for the description of different sensors for indoor positioning, but also offers the possibility to describe various sensor fusion algorithms or even existing positioning systems. The contribution of our work is twofold: The platform for hybrid positioning and the sensor description language. The feasibility of our approach is demonstrated by a prototypical evaluation of the sensor combination mechanism and estimation component, showing that hybrid positioning with a sensor description language is possible and offers advantages over single existing or proprietary systems by being able to cope with dynamically changing environments and varying user preferences.**

***Keywords*-Adaptive Indoor Localization, Opportunistic Sensor Fusion, Sensor Description Language, Hybrid Positioning Platform.**

I. Introduction

Indoor positioning has been an active research area since more than 15 years. Nevertheless, it has gained an even higher attention since the mass market penetration of smartphones, cell phones with high computational power, internet connectivity, and a large number of sensors. Being familiar with outdoor positioning technologies such as GPS and map services such as OpenStreetMap or Google Maps, users would also benefit from the availability of position information in large and complex buildings.

Over the years, many indoor positioning systems have been developed using various technologies, positioning methods, or techniques for sensor fusion. A similar variety can be observed concerning the application scenarios and use cases of indoor positioning, ranging from pedestrian and robot navigation over interactive guides in museums to ad hoc positioning without infrastructure in search and rescue scenarios. Tracking of goods has other requirements than surveying elderly people in ambient assisted living. In the first case, multiple low-power sensors might be chosen for attachment to every item and then precisely tracked by a dedicated infrastructure. On the contrary an old person should better be equipped with a single sensor with no or little expenses for additional infrastructure. Unfortunately, there is still no ultimate positioning method available today, which satisfies the requirements of most possible application scenarios at low expenses.

The cost is one of the most limiting parameters of an indoor positioning system. While systems with high accuracy and precision often based on UWB [1] or lasers [2] (with the magnitude of centimeters to millimeters) are available, the cost of such systems limits the coverage to small areas, where the high quality compensates the expenses. Cheaper systems often based on inertial sensors [3], [4], WLAN [5], [6], [7], or cameras [8], [9] suffer from inaccuracies, high calibration efforts, or large latencies.

Another limiting factor is the provision of positioning systems to the user. Some systems require dedicated sensors or infrastructure [10], other systems work self-contained on mobile phones [3]. Nevertheless, most systems need to provide some kind of map data or reference system, a communication link, or an interface for location determination. Some systems are limited to regions owned by certain authorities, most communicate over proprietary protocols, and even if you have a positioning system running in your university it probably won't work elsewhere, even if the same sensors and infrastructure are available.

We propose to solve these problems with a platform for hybrid positioning which chooses the most suitable (or possibly a combination) from all available sensor information for position estimation. The platform also offers sensor discovery mechanisms and environmental information and serves as a single service point for positioning. For that task, a sensor description language is defined. The language allows for the description of different sensors for indoor positioning, but also offers the possibility to describe various sensor fusion algorithms or even existing positioning systems. We define a sensor to be an entity measuring some phenomenon, thus including also sensor fusion algorithms and positioning systems to the definition. Furthermore, the

978-1-4673-1954-6/12/$31.00 ©2012 IEEE

language is able to serve as communication means between several clients and the platform. Hence, the contribution of our work is twofold: we present a novel sensor description language and a platform for hybrid positioning based on that language.

The platform consists of a sensor management component, a sensor discovery mechanism, a component for position estimation, and an advanced sensor combination mechanism which dynamically chooses those sensors for fusion and position estimation that are best with respect to the user criteria. This is achieved by comparing the capabilities and requirements of all available sensors, which are described in the sensor description language, thus computing all possible combinations of sensors and the estimated properties of each combination. These are matched to the user criteria and the best matching combination is utilized for position estimation.

The rest of the paper is structured as follows: In the next section, related work is presented and some differences to our approach explained. Then follows the introduction of the sensor description language PositioningML in Section III. Section IV describes the platform and its components focusing on the reasoning and combination of several sensors, preceding an application scenario and empirical evaluation in Section V. The paper is finalized by a conclusion.

II. Related Work

In this section, an short overview on positioning systems and common sensor fusion mechanisms is given. The focus is on opportunistic sensors since dedicated systems often obtain sufficient accuracy without further fusion. In this paper, opportunistic sensors are understood as sensors which primary goal is not indoor positioning, but which can be used for position estimation. Examples are WLAN cards, low cost accelerometers and gyroscopes, cameras, and microphones. Then a quick introduction to sensor fusion is given, followed by a short presentation of existing sensor description languages.

A. Single technology systems

WLAN positioning has been extensively researched in the past years. In laboratory environments, the median accuracy and precision of advanced systems can reach one meter [6]. Nevertheless, WLAN-based systems are error prone due to interferences and attenuation effects induced for example by humans in the environment [11]. While today's standard interfaces are limited in the use of time-based lateration techniques [12], signal strength based lateration, or fingerprinting enjoy great popularity. However, those systems are prone to jump between consecutive position estimations and thus lack capabilities in high quality tracking scenarios such as a continuous navigation aid.

Positioning systems based on inertial sensors, i.e., accelerometer, gyroscope, and digital compass, usually depend on step detection, step length estimation, and step heading detection [4], [3]. While those systems are especially suitable for continuous positioning since they offer relative positioning information, the overall error accumulates over time. When map information is available, it can be used to reduce the accumulation by applying map matching techniques. However, inaccurate sensors and building layout problems such as symmetries [13] still prevent inertial sensors from being a solution for reliable positioning over large distances.

Cameras offer multiple opportunities for position estimation. Either consecutive images are compared and the relative movement is deduced by the optical flow for dead reckoning [14] or image recognition is applied on calibrated databases [9]. However, camera-based systems can suffer from problems with occlusion, people moving through the field of view, high processing times, and high energy consumption.

B. Bayesian Filters for Sensor Fusion

Due to the insufficiency of single technology systems, most of today's indoor positioning systems rely on a multitude of sensors. Their measurements need to be combined to overcome the shortcomings of each single source of information. By using Bayesian filters, uncertainty of these multiple sensors can be expressed and processed. Therefore, most indoor positioning systems rely either on particle filters or on Kalman filters for sensor fusion of heterogeneous sources. Both present ways for recursively estimating the state of a dynamic system such as the position, orientation and velocity of a moving target. As usual with Bayesian filters, both work with probability distributions and two models. One, the measurement model, describes the impact on measurements on the probability distribution by constructing the posterior probability density function. The second, the system model, describes the evolution of the distribution with time. Read [15] for a detailed introduction to Bayesian tracking.

C. Platforms for Hybrid Positioning

In the past several approaches for positioning platforms have been presented. One of the first approaches for structuring location systems is the location stack by Hightower et al. [16]. They refer to standard ways to combine measurements from heterogeneous sensor sources, suggest a layered architecture, and postulate that uncertainty in measurements must be modeled and preserved to the application level. Their architecture starts with a sensor layer for hardware and software components. The measurement layer transforms the raw data to canonical measurement types (i.e., distance, angle, proximity, or asserted position). The next layer offers methods for transforming the measurements to time-stamped probabilistic state estimation. The rest of the layers concerns non-positioning relevant information and were not implemented in the universal location framework,

the implementation of the location stack [17]. Here, a particle filter is utilized to fuse position estimated from multiple sources with a motion model. While the location stack provides valuable information how localization should be included to applications, it is rather an abstraction than a working platform for localizing with heterogeneous sensors since no communication means or standardized interfaces, especially for uncertainty representation, are given.

MiddleWhere by Ranganathan et al. [18] offers a middleware for handling multiple location sensing technologies and a hybrid location model with symbolic and geometric representation of the physical world. The software uses adapters to map raw sensor data to a common representation. This is processed via a provider interface where a location service fuses multi-sensor information with a geometric approach. While the adapters provide a powerful mechanism for arbitrary sensors to be added to the system, the restriction to geometric approaches is a disadvantage.

MagicMap is a project aiming at a platform for cooperative positioning. Initially it was only designed for WLAN, but today a multitude of RF-based technologies, GPS, and inertial sensors are supported. Based on the received signal strength the distance of a sensor to the transmitter can be estimated. Those distance estimations are normalized to avoid hardware dependent errors and then used for position calculation by lateration [19]. For position calculation, the position estimation by lateration can be combined in a particle filter with a mobility model possibly supported by inertial data. However, the system is limited to certain data sources and concrete position estimation methods. A common interface for describing position estimation methods, their capabilities and their uncertainty models is not yet given.

In [20] Bohn and Vogt break with the classical sensor description since they also allow whole systems to be seen as sensors and thus allow sensors to require some kind of other measurement input. These measurements are processed by an event-based mechanism so that the platform can include previously unknown sensors. However, their sensor fusion only works with probabilistic position estimates on a high level. A grid-based approach is used to fuse position estimates by first accumulating the probability per sensor per cell and then normalizing each cell to obtain a probability density function on the full grid.

With MapUme [21], Najib et al. present a middleware for location aware applications. The middleware provides a platform for multi-sensor data fusion and a location service working on both symbolic and geometric coordinates. The layered architecture resembles the location stack and different tasks can be distributed among several machines. For adding new sensors or fusion mechanisms, interfaces need to be implemented in the measurement component or the fusion engine component respectively. While the approach is very promising, we want to focus on the dynamics of of available sensors, methods, and changes in user criteria for our platform, thus introducing a more flexible platform scheme.

D. Sensor Description Languages

Last but not least, a short overview on existing sensor description languages is given. One of the first initiatives was the Sensor Web Enablement (SWE) [22] supported by the Open Geospatial Consortium (OGC). The goal is the provision of arbitrary sensors over the internet for realizing web-based sensor networks. The initiative defined standards such as SensorML [23] and Observations & Measurements (OM) [24]. SensorML is responsible for the definition and description of sensor metadata, the measurement processes, and transformations of measured data. It already supports a rating of the sensor quality. OM defines a model for domain independent representation of spatial or temporal related measurements and is therefore especially suitable for exchanging measurement data.

Another language, called the Sensor Fusion Modeling Language [25], defines a communication model for tracking systems. It allows the description of physical or functional properties of systems including type of sensor, position, and a description of the used sensor fusion mechanism. Furthermore, uncertainty can be modeled by giving an accuracy or confidence intervals for position estimates. The language is focused on the description of the output and not intended for the description of required input for tracking systems.

The Sensor Abstraction Layer [26] allows to hide heterogeneous sensor sources behind a common interface. Sensor descriptions are based on SensorML extended with service access points and commands a sensor can carry out. One of the great advantages is the automatic detection and configuration of new sensors.

III. PositioningML: A Sensor Description Language

For interoperability with a multitude of sensors, algorithms, positioning systems, environmental models, and users, a standardized language for sharing and processing information is defined. PositioningML is based on XML and extends the use of SWE [22] with languages such as SensorML [23] and OM [24] for the task of positioning. Further elements from MathML [27] or UncertML [28] add expressing power enabling the language to express even complicated algorithms or the uncertainty of positioning processes.

The description language is used for several tasks in the platform. One of these is the description of sensors. We define a sensor to be an entity measuring some phenomenon, thus including also positioning and sensor fusion algorithms, simulation models, and whole positioning systems to the

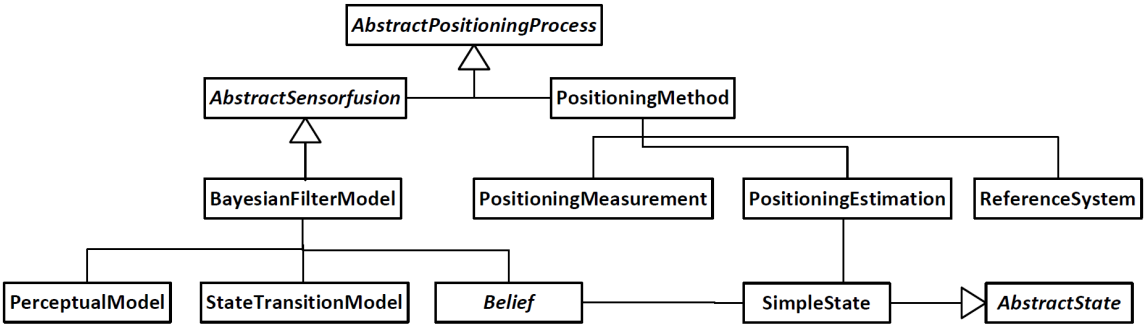

Figure 1: Associations between main classes in PositioningML

definition. PositioningML defines a type **AbstractPositioningProcess** as base for all positioning sensors, which is derived by the SensorML type AbstractProcess.

Sensors for positioning can come in two different fashions: Either the sensor represents a positioning method described by the type **PositioningMethod** or a sensor fusion mechanism described by the abstract type **AbstractSensorfusion**. At the moment, the only realization of the latter is the **BayesianFilterModel** including standard sensor fusion mechanisms such as Kalman and particle filters. Other methods could easily be added to the language by adapting the description of the Bayesian filters. Basic sensors, e.g., a compass, accelerometer, or camera, can be described by standard SensorML types, thus a variety of sensors can be included in the positioning process.

A. Bayesian Filter Model

Each Bayesian filter model can be described by a filter method of type **FilterMethod**, which can have a textual definition, a link to binary or source code in arbitrary programming languages, or a complete description of all the filter steps in MathML using the type **ProcessMethodExtended**. This enables the language to express algorithmic details, allowing the transfer of program logic from providers to the platform or even to client applications for terminal-based positioning.

Besides a description of the filter method, a **BayesianFilterModel** is characterized by one or more perceptual models described by the type **PerceptualModel**, a state transition model (**StateTransitionModel**), the current **Belief**, and properties of the fusion method, which are given in form of the type **FusionQualityCharacteristics**. Furthermore, input and output of each filter model are defined. The input consists of an observation vector and an optional control input vector modeling known input characteristics of the filter. The output is a state vector giving the state of a system after the input observation is processed. It combines the knowledge inferred by the state transition model and the perceptual models.

The **PerceptualModel** describes the deduction of a state from an observation or measurement. This model contains a list of sensor descriptions from sensor types which measurements can be processed by the model. Note that sensors in this case can also include positioning methods, systems, or sensor fusion mechanisms, as well as any basic sensors described in SensorML. If deviations or inaccuracies of a sensor are known, the information can be modeled as noise of that sensor affecting the estimation of a state based on a measurement. Noise can be described with UncertML for complex distributions or MathML for simple functions. The perceptual model can process an input vector of measurements given in form of an **AbstractState** which is either realized as **SimpleState** or **OMState**. A **SimpleState** consists of one or several measurements of sensors together with a timestamp and an optional belief representing the uncertainty of each measurement. As an alternative, an **OMState** in the standard OM syntax can be given for direct integration of existing standards. However, it is at the moment not possible to model uncertainty of measurements in OM syntax. A perceptual model can also be described in detail by a function, which is realized by a **ProcessMethodExtended** element exactly as a **FilterMethod**. Furthermore, map or environmental information such as walkable space can be included.

The **StateTransitionModel** describes the transition of a state over time. It can also be described by function of type **ProcessMethodExtended** and is used to calculate the state vector continuously in between measurements processed by the perceptual model. Typically, state transition models refer to movement models or movement prediction models.

Belief expresses the degree of uncertainty in a state based on the inaccuracies in both the state transition and the perceptual model. Depending on the Bayesian filter model this can be in form of mean values and standard deviations for Gaussian distributed states, e.g., in the case of a Kalman filter, or a discretization of arbitrary distributions, e.g., the particles of a particle filter. The **Belief** is given in form of an UncertML element, offering several distributions and other means to express statistical information. These statistical representations enable comparison and processing between different filter models.

FusionQualityCharacteristics finally are essential for the mapping of sensor fusion algorithms to sensors. This includes the information for which types of data or systems the fusion is suitable. Kalman filters for example work best on linear systems with Gaussian distributed data. Hence, the characteristics offer valuable decision support for determining suitable input for the fusion. Furthermore, the filter can demand a certain number of dimensions of state or input vector, be limited to a certain fixed number of states, be able to express multi-modal distributions and so on. These attributes are also defined in the **FusionQualityCharacteristics**.

B. Positioning Methods

A **PositioningMethod** is modeled by a list of input sensors of the SensorML type AbstractProcess which are required for the described positioning method. These sensors can be other positioning methods, systems, sensor fusion mechanisms or any basic sensors, either described by PositioningML or SensorML. Instead of or in addition to sensors, a number of **PositioningMeasurements** can be given. The position estimation is described by a type **PositioningEstimation**. Both, **PositioningMeasurements** and **PositioningEstimations**, can be described in arbitrary detail from a textual description to the full functionality of calculation steps by a **ProcessMethodExtended** element. Each position corresponds to a certain **ReferenceSystem**. Furthermore, a positioning method can utilize an arbitrary number of the previously described sensor fusion mechanisms given by **AbstractSensorfusion** elements. Similar to fusion models, **PositioningQualityCharacteristics** are defined to express accuracy, precision, latency, and other properties of the positioning method. Thus, positioning methods for the diverse application scenarios can be distinguished and compared. Finally, a **PositioningMethod** has a defined input vector of measurements and a state vector as output. The types **PositioningMeasurement**, **PositioningEstimation**, and **ReferenceSystem** are described in more detail:

A **PositioningMeasurement** describes physical observations, requirements, transformations and qualities of measurements. The observations are characterized in the same way as an input vector in the case of the sensor fusion's perceptual model. In addition, a **PositioningMeasurement** is characterized by so called observables, which are properties derived from the measurements, e.g., a distance which could be derived by loss in signal strength or signal propagation time. For that purpose, reference points can be given. This might be required for training data in a fingerprinting system or to express the position of reference stations for lateration. Furthermore, a **PositioningMeasurement** can refer to special conditions, which are time-, value-, or token-dependent. The condition limits the validity of a measurement for the positioning method. Time conditions restrict the time and value conditions the value of a measurement. The latter can be utilized for example to limit the allowed measured signal strength for distance computation from signal strength in a path-loss model. Token conditions can be used to restrict textual value descriptions to some regular expression.

PositioningEstimation describes the deduction of a position from a **PositioningMeasurement**. Position estimation can include the description of algorithms, environmental maps and reference points. It can handle information about reference positions the same way as a **PositioningMeasurement**. A position estimation returns a state vector which contains the position information the positioning method can calculate. This is also the output state vector of the positioning method. **PositioningEstimation** is characterized by one or more of five basic techniques for positioning. These are angulation, lateration, fingerprinting, dead reckoning, and proximity detection which can be extended for special cases. Thus, positioning methods can be distinguished at this level, requiring different sensors.

A position is described concerning some **ReferenceSystem**. It can have an arbitrary format (symbolic, spatial, global, local, absolute, relative, ...) and serves as the context for interpreting a position. For enabling transformations between several coordinate systems on the platform, a transformation scheme should be given to a standard reference system such as WGS84.

IV. Platform for Hybrid Positioning

In this section, the platform is presented starting with requirements, then giving an overview, and finally describing the components with a focus on the reasoning for the combination of sensors.

A platform for hybrid positioning has to fulfill multiple requirements originating from various application scenarios and the inherent requirements from certain positioning methods. Other requirements come from the need for interoperability of the platform with existing systems. We identified the following requirements:

- A description of sensors is needed to include specific sensor characteristics, but also expected data input and output formats. Furthermore, such data formats as well as user preferences and communication channels need a standardized description. For this reason a description language named PositioningML has been introduced in Section III in detail.
- A matching between user preferences, sensors, and sensor requirements should be possible to offer positioning suitable for every situation.
- The platform should be able to cope with dynamically changing environments, e.g., seamless outdoor and indoor positioning, newly available or unavailable sensors and changing user criteria during the positioning process.
- It should be possible to dynamically combine multiple sensors and their measurements, utilize sensor fusion

mechanisms, and calibrate inaccurate sensors based on knowledge from more accurate sensors.

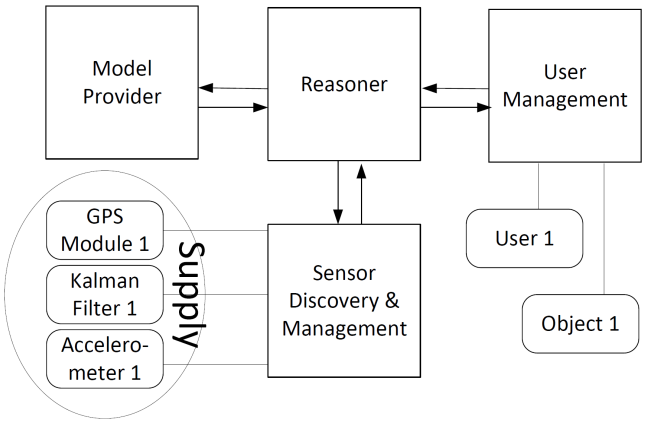

Figure 2: An overview of the components of the platform with example sensors and users

To fulfill requirements, the platform consists of four components (compare Figure 2). First a short overview on the platform is given, before the components are described in detail. One component consists of all the sensors currently available to the platform, which is called the *supply*. Availability means a sensor is temporarily available (e.g., a positioning system of a supermarket is enabled or is accessible at a time) and spatially available (e.g., a mobile enters the area covered by an indoor positioning system, sensors are plugged in, or server are connected). A sensor discovery mechanism constantly supervises the sensors and detects the unavailability of existing sensors as well as the appearance of new sensors. Discovered sensors provide their description in PositioningML, which is parsed and the sensor then added to the *supply*. The most essential component is the reasoner. It continuously accesses measurements issued by a sensor from the *supply*, matches the measurement with other sensors from the *supply* and generates a position estimate suitable for the current user criteria. The reasoner has access to environmental information which is provided by the model provider and can also access former measurements of registered sensors (e.g., last known GPS position as initial position for dead reckoning method) as well as all sensors from the supply. Another component manages users with their criteria and devices.

A. Sensor Management and Discovery

The platform provides an interface for external sensors to register and offer their observations to the platform. The registration of new sensors and the availability check of existing sensors are done in the component for sensor discovery. Sensors are described by corresponding XML-types of PositioningML defining the capabilities, the possibly required input vector of data, and the observations as output vectors. New measurements and new sensors generate events which are handed to the reasoner component for further processing. The sensor management component also serves as a transparency layer for sensor abstraction and handles the communication between platform and sensors via the sensor interface. The syntax of messages is given by PositiningML. So a provider of an external sensor can easily integrate it to the platform by adding a communication stub for interpreting PositioningML messages.

The sensor discovery component is responsible for finding available sensors and adding them to the platform. For this task, common service discovery mechanisms such as lookup-server or agent-based systems can be used. This component also checks the availability of existing sensors and removes sensors which are no longer available. All currently available sensors are managed in a database suitable for dynamic environments. This is needed as the number of sensors in the supply changes just as their properties such as workload and power consumption. Their interfaces are made available for the reasoner component. If a new sensor becomes available, its PositioningML description is stored in the database. The various SensorModels representing physical or logical sensors, as well as sensor fusion algorithms and positioning methods are processed by the reasoner.

B. User Management

This component manages the platform's users and their criteria. A user can have certain criteria such as energy consumption, positioning accuracy, or needs a positioning method without former training sequences, restricting the use of certain methods or sensors. The criteria are utilized by the reasoner component, meaning that those sensors are combined for positioning which output and processing characteristics such as processing time are most suitable to the criteria. If a user has all the sensors of a modern smartphone available and his criteria are low energy consumption and street level accuracy, then GPS should be replaced by WLAN positioning, dead reckoning and map matching. If criteria are conflicting, the user can either give priorities or the platform tries to fulfill them as best as possible. This means for example choosing the method with the lowest energy consumption from all methods fitting a certain accuracy requirement. Note that our definition of a user also includes non-human targets such as objects and that it is also possible to obtain the position of another user. This should surely be restricted out of privacy concerns, but the rights management of the platform is outside the scope of this paper.

C. Model Provider

The model provider stores and manages environmental information such as floorplans, maps, and pattern matching databases, e.g., for fingerprinting algorithms. It offers coordinate transformation services between models and a global

positioning system such as WGS84. For this task, several parameters for each map such as the shift, rotation, and scale need to be defined and provided together with each model. Models are assigned a unique identifier and an absolute positioning system should always refer to this id when returning a position. Environmental information is exchanged in a standardized format. While fingerprint databases can be described in PositioningML, building models can either be exchanged as bitmaps conform to a certain color scheme (i.e., black walls and white walkable space) or in a building description language such as BIGML [29].

D. Reasoning and Combination

The main component of the platform is the reasoner. It has the task of processing new measurements, compare them with user criteria, compare SensorModels of the supply to each other, and decide how to further process measurements according to their SensorModels. If the measurement already fits to user criteria, e.g., if it corresponds to a position estimation, it can be stored in a database and/or provided directly to the user. If the measurement needs to be further processed, e.g., in the case of raw data, the reasoner checks whether it has the means to do the processing itself or assign the task to another sensor. When a measurement is sent to another sensor, which description declares its ability to handle the measurement, the sensor's answer is again given in form of a new measurement. The checking procedure is explained in more detail below. Communication with external sensors is done via PositioningML. Whenever the reasoner or a sensor requires environmental information such as a map, topological data, or a fingerprint database, it can be provided by the model provider to the reasoner and then further distributed to the sensors.

Since the event of a new measurement usually occurs more often than changes in the available sensors, the workflow of processing measurements, i.e. the reasoning, can be optimized for a single user. For efficiently calculating a position from a fixed set of heterogeneous sensors, the process of reasoning should be carried out only once by the following steps:

1) For each sensor model representing a positioning method the required input sensor models are found. This is done by matching the result vectors of the sensor candidate with the required input vector. If both fit, a link is established. If the input only fits partially, the search is continued for the missing parameters. At that point the in Section III-B mentioned observables and basic techniques for positioning are matched. A sensor delivering distances can be used in circular lateration methods. Here, the method can also be supplied with environmental information.
2) For each sensor model representing a positioning method available sensor fusion mechanisms specified for a positioning method are found and linked to the method. If several fusion mechanisms need to be processed in a certain order, they are linked in that order. The particularity of the reasoner is, however, that other sensor fusion mechanism of the supply not specified for that positioning method can be linked. This is done in the next step.
3) For each sensor model representing a sensor fusion algorithm, all positioning methods or systems are found and linked the sensor fusion algorithm can process. For this task, the state transition model and the control input vector are analyzed and compared to the input and output of the candidate sensors. Input data is compared in order to fuse signal data on a lower level, output data in order to fuse positioning results on a higher level. The decision is also based on the quality criteria of the fusion algorithm. It is checked for example if the filter demands a certain number of dimension for data to be fused. For all matching sensors, links are established.
4) For each sensor model representing a sensor fusion algorithm all further required sensors are found that do not necessarily correspond to already linked sensors. This is achieved by matching the sensor fusion's input vector to the candidate sensor's output similar to step 1. In this step, environmental models are also distributed to the sensor fusion mechanism.
5) Finally, each of the linked process chains is evaluated with respect to the user criteria resulting in optimal positioning for the user from the current supply. If enough energy or processing capabilities are available, multiple process chains can be executed and the best result returned to the user. Besides returning a positioning result to the user, it is also stored at the reasoner. Thus, it can be processed by other positioning or sensor fusion methods, e.g., as reference points for dead reckoning methods, fusion with other positioning results, or as input for a perceptual model.

This process should be carried out in the case a new sensor becomes available, has disappeared, or the user criteria have changed. An example is given in the prototypical evaluation. After finishing these steps, the processing of each new measurement can directly follow the established links without the need of further reasoning.

E. Theoretical Evaluation

The problem of the dynamic combination of different sensors, positioning systems, and algorithms for position determination or sensor fusion is hereby solved by the fact that all components are described as necessary so that dependencies and combination possibilities can be determined. Thus at any instant the best combination of sensors of the supply can be calculated which is eventually used for position estimation.

The platform has some advantages over similar approaches in literature. The main strength is its flexibility, allowing to include various sensors, use them for positioning, and to operate on portable devices such as smartphones or in a server infrastructure. It can be administered by building operators for local use or as a large scale solution for global operation.

The platform enables the provision of models, positioning methods, algorithms over the boundaries of the platform itself. This means that clients can be provided with so detailed descriptions of algorithms that automated code generation mechanisms can be utilized to even port the position calculation to clients of the platform, allowing for self-contained or privacy preserving positioning on mobile devices or the computation of complex algorithms on a server for energy preservation on a mobile device.

V. Application Scenario and Evaluation

In this section, a prototype of the platform at our university is evaluated. The evaluation concerns the ability to handle heterogeneous sensor data as well as changing sensor availability and information fusion.

A. Scenario

To demonstrate our approach, we applied the developed platform to a case study. We assume that a student is using the positioning platform and his mobile device for navigation to a lecture room in the university building. Therefore, the student has specified certain criteria for the current positioning process: The accuracy of the positioning process should be suitable for navigation and the calculation of the position may need a maximum of three seconds.

The platform recognizes these user criteria and the mobile device with his GPS sensor, WLAN sensor, accelerometer and compass. These sensors are described in the PositioningML language. Afterwards, the platform waits for an absolute position estimate, which is in this case returned from the GPS position sensor since no matching WLAN positioning method is found for the active WLAN scan results. The user criteria are met and the positioning is started. Since the platform incorporates a step detection algorithm, it can also calculate the student's mean step length from his GPS positions and save it for later processing.

Subsequently, we assume the student enters a building, where the platform detects an existing WLAN infrastructure and WLAN fingerprint database described in PositioningML. Furthermore, a particle filter based on accelerometer and compass as well as a simple kNN-algorithm located on a dedicated server are detected and added to the available sensors. Since no GPS position is returned for three seconds, the platform checks registered algorithms and positioning methods available for the user. Its intent is to find an algorithm, which can offer an indoor position fix according to the user criteria. Since the particle filter is not yet initialized, the simple kNN-algorithm is selected and the positioning is seamlessly continued.

During his movement inside the building, the number of visible WLAN access points strongly fluctuates, as well as their signal strengths. To improve the stability of the positioning process and to retain accuracy of positioning, calculation speed and energy consumption, the platform checks sensors and algorithms for use in the positioning process. The particle filter is used after an initial position is given by the kNN-algorithm. Alternatively a GPS-fix near the building could be used for initialization. Accelerometer and compass data from the mobile device are sent to the reasoner and handed on to the interface of the server hosting the particle filter. There, the data is processed and combined. Additionally, the platform is able to provide the filter with a personalized mean step length of the user calculated from positions estimated by previously executed methods. The positioning results of the particle filter are returned to the platform and, when matching the user criteria, handed on to the student on his mobile phone. While the filter's accuracy is higher than that of the kNN-algorithm, the particle filter is utilized for positioning. If the WLAN position is more accurate it is returned to the user and used for reinitialization of the particle filter.

B. Platform Capabilities

The scenario stresses some of the platform's capabilities. It is not only able to handle measurements from heterogeneous sensor sources, but also to provide positioning methods and sensor fusion algorithms with the required input data. Position data from several sensors can be evaluated with respect to the belief and the platform can decide how to proceed.

In the scenario, the estimated accuracy of position information is used to decide which position should be sent to the user. As long as GPS is available, it is used for positioning. At the same time, step detection is carried out to calibrate the step length of the user with the measured GPS distance. When GPS is not or no longer available (after a certain timeout, in this case the three seconds update time from user criteria), an alternative in form of WLAN positioning is found and also used for initialization of the particle filter. The WLAN positioning method is provided with a fingerprint database by the platform and the particle filter with the user's mean step length, a map for map matching, and an initial position. Subsequently, the kNN algorithm and particle filter steps are executed in parallel. The reasoner compares the accuracies of positions (given in form of the **Belief**) estimated by both methods and returns only the more accurate position to the user. In case of additional energy constraints, the method with a better overall accuracy could have been utilized or the more energy efficient method when its accuracy is sufficient.

Figure 3: Two tracks corresponding the scenario, estimated track in red (left) and blue (right), real tracks in black.

What is more, the platform can directly provide environmental and fingerprint information and utilize the position information to support the search for the right building model. At first (if available and timely) the last known world coordinate can be transformed to a local coordinate and the distance to the available models calculated. When a model is near and at least three access points in the database correspond to access points in the current WLAN measurement, it will most probably be the environment the user is situated in. Multiple candidate models can be further distinguished by including all candidates in position estimation and returning the model with better fitting fingerprint data.

Last but not least, the platform can provide position information relative to all available models, transform the coordinates and thus allows providing the position information in the form needed by the user. Existing outdoor LBS can so also use indoor position information transformed to world coordinates offered by the platform.

C. Prototypical Evaluation

The platform capabilities in the scenario were evaluated concerning two tracks at our site. Both start some distance away from the building and end inside it. The switching of positioning technology works reliable. Note that the GPS signal often failed a short distance before entering the building. Especially while recording the blue track on the right side in Figure 3, the GPS position was quite inaccurate and GPS support failed already 15 meter before entering. The largest distances between real and estimated tracks at that point have come from the timeout of three seconds as well as the fact that the fingerprint database only had reference point inside the building. Nevertheless, the indoor positioning was successfully applied and very accurate position information returned.

VI. Conclusion

In this paper, we introduced PositioningML, a sensor and position description language based on SensorML. The main elements of the language where explained and the special use for a platform for hybrid positioning stressed. Furthermore, the platform and their components for sensor management, user management, model provider, and reasoning and combination were described in detail. The platform offers seamless positioning based on heterogeneous sensors in dynamically changing environments while fulfilling certain user criteria such as positioning accuracy, position update time, or energy constraints. A prototypical evaluation underlines the platforms capabilities and serves as a proof of concept in a seamless outdoor-indoor positioning scenario.

References

[1] P. Steggles and S. Gschwind, "The Ubisense Smart Space Platform," in *Proceedings of the Third International Conference on Pervasive Computing (PERVASIVE'05)*, 2005, pp. 73–76.

[2] S. Tilch and R. Mautz, "Developement of a new laser-based optical indoor positioning system," *International Archives of Photogrammetry, Remote Sensing and Spatial Information Sciences*, vol. 38, no. 5, pp. 575–580, 2010.

[3] D. Gusenbauer, C. Isert, and J. Krösche, "Self-Contained Indoor Positioning on Off-The-Shelf Mobile Devices," in *Proceedings of the 2010 International Conference on Indoor Positioning and Indoor Navigation (IPIN'10)*, 2010.

[4] J. A. B. Link, P. Smith, N. Viol, and K. Wehrle, "FootPath: Accurate Map-based Indoor Navigation Using Smartphones," in *Proceedings of the 2011 International Conference on Indoor Positioning and Indoor Navigation (IPIN'11)*, 2011.

[5] P. Bahl and V. N. Padmanabhan, "RADAR: An In-Building RF-based User Location and Tracking System," in *Proceedings of the Annual Joint Conference of the IEEE Computer and Communications Societies (INFOCOM'00)*, vol. 2, 2000, pp. 775–784.

[6] M. Youssef and A. Agrawala, "The Horus WLAN Location Determination System," in *Proceedings of the International Conference on Mobile Systems, Applications, and Services (MobiSys'05)*, 2005, pp. 205–218.

[7] M. Kessel and M. Werner, "SMARTPOS: Accurate and Precise Indoor Positioning on Mobile Phones," in *Proceedings of the International Conference on Mobile Services, Resources, and Users (MOBILITY'11)*, 2011, pp. 158–163.

[8] H. Hile and G. Borriello, "Information Overlay for Camera Phones in Indoor Environments," in *Location-and Context-Awareness: Third International Symposium (LoCA'07)*, 2007, pp. 68–84.

[9] M. Werner, M. Kessel, and C. Marouane, "Indoor Positioning Using Smartphone Camera," in *Proceedings of the 2011 International Conference on Indoor Positioning and Indoor Navigation (IPIN'11)*, 2011.

[10] A. Smith, H. Balakrishnan, M. Goraczko, and N. Priyantha, "Tracking moving devices with the cricket location system," in *Proceedings of the 2nd international conference on Mobile systems, applications, and services (MobiSys'04)*, 2004, pp. 190–202.

[11] P. Mirowski, H. Steck, P. Whiting, R. Palaniappan, M. MacDonald, and T. K. Ho, "KL-Divergence Kernel Regression for Non-Gaussian Fingerprint Based Localization," in *Proceedings of the 2011 International Conference on Indoor Positioning and Indoor Navigation (IPIN'11)*, 2011.

[12] P. Uthansakul and M. Uthansakul, "WLAN Positioning Based on Joint TOA and RSS Characteristics," *World Academy of Science, Engineering and Technology*, vol. 52, pp. 1261–1268, 2009.

[13] O. Woodman and R. Harle, "Pedestrian Localisation for Indoor Environments," in *Proceedings of the International Conference on Ubiquitous Computing (UbiComp'08)*, 2008, pp. 114–123.

[14] M. Muffert, J. Siegemund, and W. Frstner, "The Estimation of Spatial Positions by Using an Omnidirectional Camera System," in *Proceedings of the 2nd International Conference on Machine Control and Guidance*, 2010, pp. 95–104.

[15] M. S. Arulampalam, S. Maskell, N. Gordon, and T. Clapp, "A Tutorial on Particle Filters for Online Nonlinear/Non-Gaussian Bayesian Tracking," *IEEE Transactions on Signal Processing*, vol. 50, no. 2, pp. 174–188, 2002.

[16] J. Hightower, B. Brumitt, and G. Borriello, "The Location Stack: a Layered Model for Location in Ubiquitous Computing," in *Proceedings of the Fourth IEEE Workshop on Mobile Computing Systems and Applications (WMCSA'02)*, 2002, pp. 21–28.

[17] D. Graumann, J. Hightower, W. Lara, and G. Borriello, "Real-world Implementation of the Location Stack: The Universal Location Framework," in *Proceedings of the Fifth IEEE Workshop on Mobile Computing Systems and Applications (WMCSA'03)*, 2003, pp. 122–128.

[18] A. Ranganathan, J. Al-Muhtadi, S. Chetan, R. Campbell, and M. D. Mickunas, "Middlewhere: a middleware for location awareness in ubiquitous computing applications," in *Proceedings of the 5th International Conference on Middleware (Middleware'04)*, 2004, pp. 397–416.

[19] S. Brüning, J. Zapotoczky, P. Ibach, and V. Stantchev, "Co-operative Positioning with MagicMap," in *Proceedings of Workshop on Positioning, Navigation and Communication (WPNC'07)*, 2007.

[20] J. Bohn and H. Vogt, "Robust Probabilistic Positioning based on High-Level Sensor-Fusion and Map Knowledge," Swiss Federal Institute of Technology, ETH Zurich, Switzerland, Tech. Rep. 421, April 2003.

[21] W. Najib, M. Klepal, and S. B. Wibowo, "Mapume: Scalable middleware for location aware computing applications," in *Proceedings of the 2011 International Conference on Indoor Positioning and Indoor Navigation (IPIN'11)*, 2011.

[22] *OGC SWE Common Data Model Encoding Standard*, Open Geospatial Consortium Inc. Std. OGC 10-025r1, 2011.

[23] *OpenGIS Sensor Model Language (SensorML) Implementation Specification*, Open Geospatial Consortium Inc. Std. OGC 07-000, 2007.

[24] *Observations and Measurements-XML Implementation*, Open Geospatial Consortium Inc. Std. OGC 10-025r1, 2011.

[25] T. Schardt and C. Yuan, "A dynamic communication model for loosely coupled hybrid tracking systems," in *Proceedings of the 5th International Conference on Information Fusion (ISIF'02)*, 2002, pp. 1236–1242.

[26] G. Gigan and I. Atkinson, "Sensor abstraction layer: a unique software interface to effectively manage sensor networks," in *Proceedings of the 3rd International Conference on Intelligent Sensors, Sensor Networks and Information (ISSNIP'07)*, 2007, pp. 479–484.

[27] *Mathematical Markup Language (MathML) Version 3.0*, W3C Std., 2010. [Online]. Available: http://www.w3.org/TR/MathML3/

[28] "UncertML: Describing and Exchanging Uncertainty," online, Juli 2012, http://www.uncertml.org/.

[29] M. Kessel, P. Ruppel, and F. Gschwandtner, "Bigml: A location model with individual waypoint graphs for indoor location-based services," *Praxis der Informationsverarbeitung und Kommunikation*, vol. 33, no. 4, pp. 261–267, 2010.