A Framework for Supporting the Configuration and Automatic Integration of Heterogeneous Location Sensors

Yoo Chul Chung, Yangwoo Ko, Youngrock Cha, Dongman Lee

Information and Communications University
{chungyc, newcat, u00cha, dlee}@icu.ac.kr

Abstract. We propose a framework that supports user-friendly configuration of a new location sensor system and its integration with a location manager. The proposed framework abstracts the diversity of heterogeneous sensor technologies using adapters that provide a common interface to the location manager. Configuration of a location sensor system requires information provided by the vendor of the location sensor system, so we propose a configuration protocol with which a newly deployed location sensor system can provide and obtain configuration options and parameters. An integration protocol is proposed as well so that a newly deployed sensing system can be integrated as part of an existing location manager. In order to verify the efficiency of the proposed framework, we measured configuration time with our framework and against manual configuration. Experimental results show that the proposed framework reduces configuration time significantly.

Keywords: location management, location service, location sensors, configuration, integration, coordinate mapping, transformation matrix, adapter

1 Introduction

In ubiquitous computing environments, smart objects interact with each other and users [7]. In these environments, the location of a user is an important piece of information about the user. Smart objects decide what services to provide to specific users based on this information.

As location sensors become cheaper, users may start deploying a variety of location sensor systems in order to obtain higher accuracy using sensor fusion and to provide diverse methods for obtaining locations. In order to deploy a new location sensor system, sensor-specific coordinates need to be transformed into reference coordinates used by the location manager. Manually configuring the sensor system to support this, as is done in MiddleWhere [3] and LORE [11], requires significant effort and skill. In addition, location sensor systems may be added, moved, or removed according to the needs of the users. Therefore, a ubiquitous computing environment must be able to simplify the work required for configuring and integrating diverse location sensor systems.

Many of the current location sensor systems [8, 9, 10] require careful configuration and calibration to work properly, so manual integration of heterogeneous location sensor systems may be comparatively insignificant additional work. However, we expect that location sensor systems will become increasingly self-configuring and self-calibrating in the near future [1, 2, 6], so the effort required for manual integration may become increasingly unacceptable.

Our research introduces a framework for supporting the configuration and integration of location sensor systems with a minimum of user interaction. Each location sensor system includes an adapter, which connects to the location manager using a standard interface that abstracts the underlying sensor technologies. It also transforms a local coordinate specific to a location sensor system to a reference coordinate usable by the location manager. As a result, the framework makes it easy for users to deploy a location sensor system. Experimental results from deploying Ubisense [10] on top of a WLAN-based location sensor system show that the proposed framework reduces configuration time significantly, although manual configuration still results in higher accuracy.

The rest of this paper is organized as follows. Section 2 examines related work. We then describe the design and implementation of our framework in Section 3. We describe experimental results with a sample application based on our implementation in Section 4, and conclude the paper in Section 5.

2 Related Work

[1] and [2] propose techniques for sensor localization in a large-scale sensor network which consists of a very large number of nodes. Each sensor node has the ability to estimate distance to nearby nodes. Based on this information, each sensor node knows where it is in a common frame of reference. This sensor localization is attained in real-time without the need for human intervention. However, their work assumes the use of only a single homogeneous location sensor system.

[6] also proposes sensor localization techniques for Cricket. Initially there is no coordinate system in each node. Sensor nodes gather to form a local cluster, where each cluster has its own coordinate system. Coordinate transforms can then be computed between overlapping clusters to stitch them into a global coordinate system. As in [1] and [2], this approach cannot be applied to heterogeneous location sensor systems because they assume all nodes are Cricket nodes. In contrast, our work supports the integration of heterogeneous location sensor systems which cannot directly interoperate with each other and where each system can have their own coordinate system.

MiddleWhere [3] and LORE [11] are location managers which are able to integrate multiple location sensor systems. These systems acquire locations of objects from multiple location sensors and fuse the sensor data by using a variety of algorithms. Their focus is on producing more accurate locations and representing locations in various forms. Each location sensor system uses an adaptor to transparently integrate with the location manager. MiddleWhere and LORE require the manual configuration of each adapter, while our work overcomes this limitation by proposing a framework which eases configuration and automates the registration of heterogeneous location sensor systems with a location manager.

3 Approach

Our framework introduces several components required for configuring and integrating location sensor systems. We also define a protocol between the components that the vendors of location sensor systems can take advantage of.

3.1 Overview

When a location sensor system is deployed, the adapter for a location sensor broadcasts its existence, which is detected by the configurator. The configurator responds to the notification and replies with its own reference to the adapter. The adapter, the configurator, and the location manager then interact with each other to configure the location sensor system and integrate it with the location manager. We envision that the vendors of location sensor systems may implement an adapter for their sensor system in order to take advantage of our framework. The adapter should include sensor-specific information such as accuracy, precision, communication mode, type and reporting rate.

However, for a ranged location sensor system, which covers an area or a volume, the configuration of the adapter cannot be done completely automatically. A location sensor system will typically use a coordinate system that is independent of that used by the location manager, especially if the location sensor system is self-calibrated. This means that we must be able to transform a coordinate in the sensor-specific coordinate system to that in the reference coordinate system used by the location manager. Since such transformations are done on physical coordinates and separate location sensor systems cannot directly know how their coordinate systems are different from each other, the configuration required to support such transformations cannot be done purely in software.

The actual transformation of coordinates is done using a transformation matrix and a displacement vector as in Equation (1). The transformation matrix represents the scaling and rotation of the coordinate systems, while the displacement vector represents the displacement of the origins between the sensor-specific coordinate system and reference coordinate system. Also, *(x, y)* is a sensor-specific coordinate in the newly deployed coordinate system, while *(x’, y’)* is a reference coordinate in the location manager for the same physical location. In the rest of the paper, we will simply refer to both the transformation matrix and the displacement vector together as the “transformation matrix.”

$$\begin{pmatrix} a & b \\ c & d \end{pmatrix}\begin{pmatrix} x \\ y \end{pmatrix} + \begin{pmatrix} e \\ f \end{pmatrix} = \begin{pmatrix} x' \\ y' \end{pmatrix} \quad \textit{(a,b,c,d,e,f are constants)} \qquad \textbf{(1)}$$

In order to actually configure the transformation matrix, the framework collects coordinates from both the newly deployed location sensor system and the location manager for the same physical location. This is done by providing sensor tags to a user from the newly deployed location sensor system and the location sensor system already integrated with the location manager. The location sensor systems detect location based on these sensor tags. A user then moves as the framework directs him to while carrying both tags. As the user moves around, coordinates from both location sensor systems are collected. Since a user can be at only one place at a time, coordinates gathered at the same time must be from the same physical location.

The collected coordinates are specific cases of how *(x, y)* maps to *(x', y')* in Equation (1). With three such mappings, we can compute the transformation matrix, since the equation is actually two linear equations with three variables each. Using only three such mappings will probably result in large errors because of errors in the detected coordinates, so instead we use two-dimensional multiple linear regression with a large number of mappings. Our framework gathers mappings from coordinates in the newly deployed location sensor system to coordinates in the location manager until the rate of change in the computed transformation matrix falls below a specific threshold.

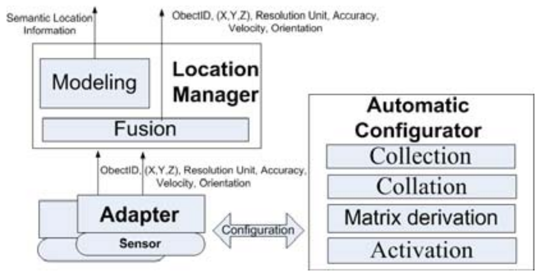

Semantic Location Information
ObectID, (X,Y,Z), Resolution Unit, Accuracy, Velocity, Orientation
Modeling
Location Manager
Fusion
Automatic Configurator
Collection
Collation
Matrix derivation
Activation
ObectID, (X,Y,Z), Resolution Unit, Accuracy, Velocity, Orientation
Adapter
Sensor
Configuration

Fig. 1. Overall architecture of the proposed framework

Once the transformation matrix is obtained, the framework sends the result to the adapter, which will use the transformation matrix to transform sensor-specific coordinates to reference coordinates when reporting locations to the location manager. Fig. 1 illustrates the architecture of our framework.

3.2 System Components

As described in Section 3.1, the framework consists of three major components – location manager, configurator, and adapters.

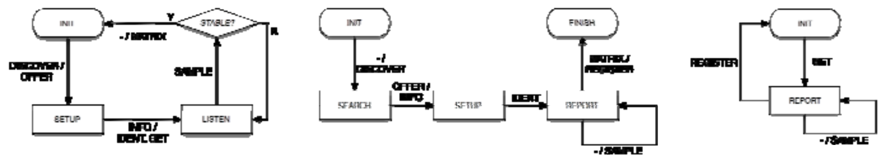
The location manager gathers data from various location sensor systems and tracks physical objects continuously. It performs sensor fusion in order to obtain more accurate locations. It also reports locations either symbolically or as physical coordinates according to application needs. The location manager is basically the central clearinghouse from which applications obtain locations of physical objects.

The configurator supports the automatic configuration of adapters so that they can integrate with the location manager. Some of the parameters that are configured are the transformation matrix and adapter identifier. In order to acquire the transformation matrix, the configurator has collection, collation, matrix derivation, and activation components. The collection component collects coordinates from the newly deployed location sensor system and the location manager. The collation component binds coordinates collected at the same time. The matrix derivation component computes the transformation matrix based on these coordinates. The activation component is responsible for configuring the transformation matrix in the adapter.

The adapter enables plug-and-play deployment of heterogeneous location sensors. Thanks to the adapter, the upper layers such as the location manager need not consider the diversity in underlying sensor technologies. It includes a transformation matrix in order to integrate with the location manager. The adapter includes error attributes which influence how the transformation matrix is obtained. An adapter also includes the estimated error when reporting locations.

3.3 Configuration and Integration Protocol

In this section we describe the protocol between the system components for configuring and integrating a new location sensor system.



(a) (b) (c)

Fig. 2 State-transition diagram for (a) configurator, (b) adapters, and (c) location manager

Figure 2(a) illustrates the state-transition diagram for the configurator. The configurator starts out in the INIT state, and waits for a DISCOVER message from an adapter. Once it receives a DISCOVER message, it replies with an OFFER message, which includes the network location of the configurator, and changes to the SETUP state. In the SETUP state, the configurator waits for an INFO message from the adapter, which includes information about the sensor system such as its type, reliability, precision and coverage. It then allocates an identifier for the adapter and sends it within an IDENT message to the adapter. Then IDENT message also includes the network location of the location manager. It also sends a GET message to the location manager. The configurator then changes to the LISTEN state, during which it continuously receives SAMPLE messages from both the adapter and the location manager. The SAMPLE messages include the coordinates of the designated sensor tags and are the adapter identifier they come from. These are gathered until enough data is collected so that the computed transformation matrix or location is stable for ranged sensors or point sensors, respectively. Once the configurator has determined that enough data has been collected, it sends a MATRIX message to the adapter, which includes the transformation matrix that should be used by the adapter, and returns to the INIT state where it waits for other new location sensor systems.

Figure 2(b) illustrates the state-transition diagram for adapters. Starting out in the INIT state, an adapter broadcasts a DISCOVER message over the network and changes to the SEARCH state. The configurator will respond with an OFFER message, which the adapter replies to with an INFO message and then changes to the SETUP state. The adapter waits for an IDENT message from the configurator in the SETUP state, from which it will learn its identifier and the network location of the location manager. It then switches to the REPORT state, where it continuously sends the coordinates of the designated sensor tag from the new location sensor system to the configurator in SAMPLE messages. It continues this until it receives a MATRIX message from the configurator. The adapter uses this message to configure its transformation matrix. It then registers its identifier, network location and information about the sensor system with the location manager using a REGISTER message, at which point configuration of the adapter is complete and it can start reporting locations to the location manager using the reference coordinate system.

Figure 2(c) illustrates the state-transition diagram for the location manager. The location manager starts out in the INIT state, where it waits for a GET message from the configurator. Once this message is received, it starts reporting the coordinates of the designated sensor tag from an existing location sensor system using SAMPLE messages. It continues this until it receives a REGISTER message from the adapter. (SAMPLE messages sent from the location manager to the configurator after the configurator sends a MATRIX message but before the adapter sends a REGISTER message are discarded by the configurator.) Integration of the new location sensor system is complete at this point, and the location manager can start measuring locations and apply sensor fusion using the information in the REGISTER message.

3.4 User Actions

Our framework currently supports two methods for deploying a new location sensor system. In both methods, the user carries sensor tags designated by the configurator from both the location sensor system being deployed and a location sensor system already integrated with the location manager.

In the first method, which we will call the guideless configuration method, the user tells the adapter that configuration and integration should begin using a PDA. The user then randomly moves around the area covered by the location sensor system being deployed. Location coordinates are continuously and automatically gathered using the carried sensor tags, which the configurator uses to compute the transformation matrix. Once the configurator determines that it has gathered enough data, it continues with the configuration and integration procedure after notifying the user that he can stop moving via the PDA.

The second method, which we will call the guided configuration method, starts out similarly with the user starting the configuration with the PDA. However, an application on the PDA guides the actions of the user. The application uses a map viewer to suggest to the user a specific location to move to. The user then moves to the approximate location, waits several seconds, and via the PDA tells the adapters to report a fixed number of coordinates to the configurator. This repeats until the configurator determines that enough data has been gathered.

The first method is much more convenient for users in that no interaction with the configurator is required while coordinates are gathered. However, the second method can be more accurate because some sensor systems become more inaccurate when there is movement (e.g. the error in Ubisense is several times larger when walking compared to standing still). Using the map viewer to suggest locations to move to also helps ensure that coordinates are gathered from a widely dispersed range, which can help improve the accuracy of the derived transformation matrix, although this requires that a map of the area be available.

4 Evaluation

4.1 Test Environment

We implement our framework as part of the Active Surroundings environment, our ubiquitous computing middleware [5]. We test the proposed location framework in a testbed which is comprised of a 36m^2 room and includes a variety of sensors, hardware appliances, and software systems necessary for a ubiquitous computing environment.

Among the sensors included is the Ubisense location system [10]. Ubisense detects locations using fixed sensors installed inside the room. These sensors receive signals from a user-carried radio beacon and estimate the location of the beacon from the relative signal strengths. Our installation of Ubisense, which only covers the Active Surroundings environment, exhibits an error range of 25cm for non-moving beacons, although the error range grows several times larger while beacons are moving.

We also develop a WLAN-based location system for the Active Surroundings environment. It uses the signal strengths of multiple wireless access points to estimate location [8]. It covers the entire building but exhibits an error range of 3m.

We test a scenario where a Ubisense location sensor system is being installed into an environment with an already integrated WLAN-based location sensor system. We test manual configuration, guideless configuration and guideless configuration for deploying Ubisense. Each test is repeated five times.

The reference coordinate system is set to that used by the WLAN-based location sensor system. The origin of this coordinate system is set to one of the corners of the room, and the axes are oriented parallel to the walls of the room. The reference coordinate system is two-dimensional, so height is not represented.

4.2 Results and Analysis

We measure the amount of time to complete configuration for each test. We also measure how much error that is exhibited in each physical location when measuring locations with the Ubisense location sensor system after it is configured and integrated. Sensor fusion is not used when measuring the errors.

The average amount of time taken to complete manual configuration is about 20 minutes. The amount of skill required for manual configuration suggests that it would be difficult and more time consuming for casual users to manually configure and integrate a new location sensor system.

The average amount of time taken to complete guideless configuration is about 5 minutes. The configurator collects 375 mappings of sensor-specific coordinates to reference coordinates during this time. The average amount of time taken to complete guided configuration is about 10 minutes. The configurator collected 15 samples each in 25 locations during this time. The resulting error distributions of the configured Ubisense system are shown in Figures 3, 4, and 5 for manual, guideless, and guided configuration.

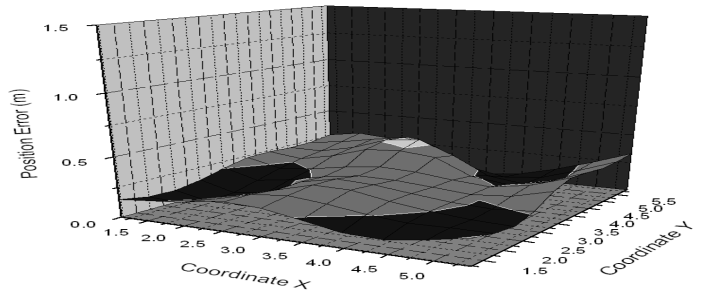


Fig. 3. Error distribution when manually configured.

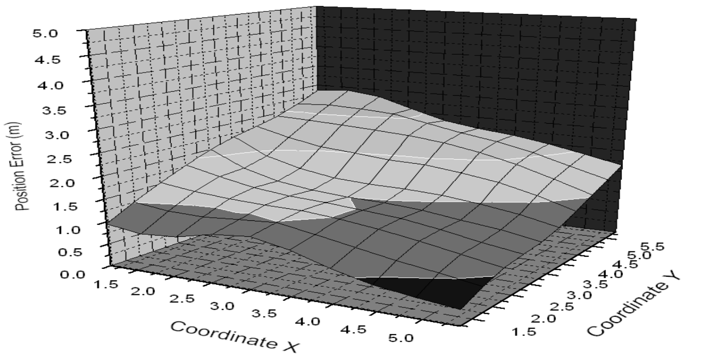


Fig. 4. Error distribution with guideless configuration.

When comparing the error distributions between the different configuration methods, we see that manual configuration is the most accurate with a maximum error of 25cm, guided configuration is next with a maximum error of 2.5m, and guideless configuration is the least accurate with a maximum error of 3m. Manual configuration is much more accurate because the most of the error comes from only Ubisense, which has an error of 25cm. Guideless and guided configuration are not only affected by errors in Ubisense, but also by errors in the WLAN-based system, which has an error of 3m. The errors resulting from the use of our framework is within the limits of the error of the WLAN-based system.

We also measure how the error changes in guided configuration when varying the number of locations from which coordinates are gathered and the number of samples gathered at each location. At least eight mappings are required to halve the measured errors. We also found that it is much more important to gather coordinates from more distinct locations than gathering more samples from each location. This suggests that the user should move in a large region covered by both location sensor systems instead of limiting movement to a small region, since moving around in a large region would result in more locations from which coordinates can be gathered.

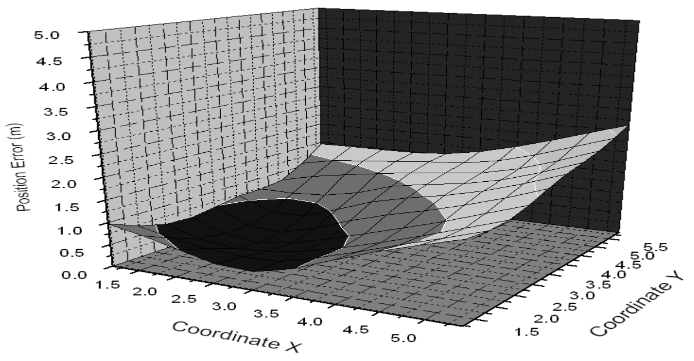


Fig. 5. Error distribution with guided configuration.

From the experimental results in this section, we can see that our framework makes the deployment and integration of new location sensor systems with an existing location manager much easier.

5 Conclusion

The goal of our research is to build a location manager that eases the configuration and automates integration of heterogeneous location sensor systems. Our proposed framework uses adapters to abstract the heterogeneity of various location sensor systems. It also defines configuration and integration protocols between location system components, which vendors of location sensors can take advantage of for easing the deployment of their sensors.

The proposed framework significantly reduces the amount of time and effort required to deploy a new location sensor system. One of the methods supported simply requires two button presses on a PDA and random movement of the user in between the button presses, while another provides a user interface with a map viewer to guide the movement of the user.

We have implemented the proposed framework in the Active Surroundings environment and successfully tested the deployment of a Ubisense location sensor system on an existing WLAN-based location sensor system. Our results show that the proposed framework is much easier to use compared to manual configuration, with accuracy being constrained only by the underlying sensor technologies.

Further issues that need to be studied are techniques for reducing errors, especially when accurate sensors are being deployed in an environment with existing inaccurate sensors, and a mathematical analysis of how errors in the underlying sensors affect the final results. We also plan to use the Active Surroundings location manager developed in this work to investigate how multiple location managers can cooperate with each other with the goal of providing more extensive location services.

Acknowledgments. This research is supported by the Ubiquitous Computing and Network Project, the Ministry of Information and Communication 21st Century Frontier R&D Program in Korea.

References

1. A. Brooks, S. Williams. and A. Makarenko, “Automatic online localization of nodes in an active sensor network”, In IEEE 2003 International Conference on Robotics and Automation, vol. 5, pp. 4821—4826.
2. A. Efrat, D. Forrester, A. Iyer, and S. G. Kobourov, C. Erten, “Force-Directed Approaches to Sensor Localization”, 8th Workshop on Algorithm Engineering and Experiments (ALENEX), 2006, p. 108-118.
3. A. Ranganathan, J. Al-Muhtadi, S. Chetan, R. Campbell, M. Dennis Mickunas, “MiddleWhere: A Middleware for Location Awareness in Ubiquitous Computing Applications”, ACM/IFIP/USENIX 5'th International Middleware Conference , Toronto, Ontario, Canada, October 18th - 22nd, 2004.
4. C. Jiang and P. Steenkiste, "A hybrid location model with a computable location identifier for ubiquitous computing", Lecture Notes in Computer Science, 2498, UbiComp, 2002
5. D. Lee, S. Han, Insuk Park, S.K., K. Lee, S.J. Hyun, Y.H. Lee, and G.H. Lee, "A Group-Aware Middleware for Ubiquitous Computing Environments", ICAT 2004, November 2004.
6. D. Moore, J. Leonard, D. Rus, and S. Teller, "Robust distributed network localization with noisy range measurements," in Proceedings of the Second ACM Conference on Embedded Networked Sensor Systems (SenSys '04), Baltimore, MD, November 2004.
7. D. Saha, A. Mukherjee, "A Paradigm for the 21st Century", IEEE Computer, IEEE Computer Society Press, pp. 25-31, March 2003.
8. M. A. Yousief, “HORUS: A WLAN-based indoor location determination system”, Ph.D. dissertation, University of Maryland, College Park, MD, 2004.
9. N. B. Priyantha, A. Chakraborty, and H. Balakrishnan. “The cricket location-support system”, In Proceddings of MOBICOM 2000, pages 32-43, Boston, MA, August 2000. ACM, ACM Press
10. UbiSense, http://www.ubisense.net/.
11. Y. Chen, X.Y. Chen, F.Y. Rao, X.L. Yu, Y. Li, and D. Liu, "LORE: An infrastructure to support location -aware services", VOL. 48, NO. 5/6, SEPTEMBER/NOVEMBER 2004