# The Smart-Condo™ Infrastructure and Experience

Iuliia Vlasenko, Meisam Vosoughpour Yazdchi, Veselin Ganev,
Ioanis Nikolaidis, and Eleni Stroulia

Department of Computing Science, 2-21 Athabasca Hall, University of Alberta,
Edmonton, Alberta, Canada, T6G 2E8
{vlasenko,vosoughp,vganev,nikolaidis,stroulia}@ualberta.ca

**Abstract.** The term "Smart Home" refers to a home equipped with sensors, which observe the environment and the actions of its occupants, and actuators, which automatically control the home ambience and devices. A Smart Home can provide a variety of services to its occupants, based on information gleaned from the data recorded by the sensors and using the automation afforded by the actuators. Our work in the Smart-Condo™ project has been motivated by healthcare concerns: we aim to support people with chronic conditions to live independently longer. To that end, our first objective has been to develop an accurate location- and activity-recognition method. In this paper, we describe the Smart-Condo™ middleware architecture, focusing on its occupant-localization feature. We report on simulation experiments with three sensor placements, one of which was deployed at a recent indoor localization competition. Finally, we draw a comparison between the simulated and real-world results, showing the potential practical significance of our methodology.

**Keywords:** Smart Home, Ambient Assisted Living, Wireless Sensor Networks, Indoor Localization, Virtual World, Simulation, Sensor Placement.

## 1 Introduction

The term "Smart Home" [5] refers to a home embedded with sensors, to observe the environment and its occupants' activities, and actuators, to automatically control the home ambience and devices in a way that improves the occupants' experience. Sensor-based systems are being studied as a means of non-intrusively monitoring a person's activity and providing this person, and his formal and informal caregivers, with information, useful for making decisions regarding his care [6][8][9][14]. In our work on the Smart-Condo™ project [3][4][15][16][17] we have been developing a comprehensive platform for addressing exactly this research problem.

The first version of the Smart-Condo™ platform was deployed in 2009 in an office reconceived (and sparsely refurnished) as the home of a three-person family. This deployment included only motion sensors, pressure sensors and switches, and served simply as a feasibility exercise.

In the summer of 2011, the platform was deployed in the Independent Living Suite (ILS) of the Glenrose Rehabilitation Hospital, in Edmonton, Alberta, Canada [1] [17].

S. Chessa and S. Knauth (Eds.): EvAAL 2012, CCIS 362, pp. 63–82, 2013.

The deployment was used to support discharge planning: patients, about to be discharged, stay in the ILS for several days, in order for the discharge team to determine whether they are ready to be discharged and live on their own, or whether they need to remain at the hospital longer. Typically, while patients stay in the ILS, nurses have to periodically check on them for the sake of their safety and wellbeing, which contradicts the purpose of the stay, namely for the patients to demonstrate their ability to live autonomously. The Glenrose clinicians have considered using video cameras to observe the patients' stay, however, some patients are unwilling to accept this technology out of privacy concerns. We deployed our platform and analyzed data recorded during the stay of two patients in the ILS. This deployment included a variety of sensors, through which we monitored movement, opening and closing of doors/drawers/kitchen cabinets, usage of electrical appliances, usage of furniture, bed occupancy, usage of the bathroom, and medication adherence. The collected data were used to generate reports and visualizations for the discharge team.

Between the two deployments, we have focused on two specific research problems, which have driven the evolution of our platform. First, we are concerned with extensibility of the platform, both in terms of the sensors that may be integrated in a home (including home-environment sensors, activity sensors and personal home-care devices), and in terms of the services that the platform may be required to support (including localization, activity recognition, alarm generation etc). In principle, standardization is required at (a) the sensor level, and at (b) the service-interface level. The former requirement is motivated by the need to seamlessly integrate new sensors, as they become available. When a new type of sensor reading and/or protocol is introduced to the system, the system must be able to use it and synthesize it with other pre-existing sensor data. The latter requirement captures the need for general-purpose platforms that can integrate a variety of services of differing computational complexity, as mentioned above. Second, we recognize that a major factor influencing the adoption of such technologies is the deployment and operational costs involved. To enable informed decision making on the part of potential adopters, we have developed a systematic process for simulating and evaluating the performance of the system under particular deployment conditions. Through this process, we can explore, in the pre-deployment phase, how we may reduce the number of sensors and yet be able to attain a certain performance level. To demonstrate the validity of our approach and compare the pre-deployment evaluation results with empirical results, we participated in the "Evaluating AAL Systems through Competitive Benchmarking" (EvAAL, http://evaal.aaloa.org/) competition in July 2012. The competition deployment became the third milestone for the Smart-Condo™ project; this paper highlights lessons learned while preparing for and during the competition.

In this paper, we discuss the high-level software architecture of the Smart-Condo™ platform (Section 2); we describe its support for simulation-based deployment configuration planning (Section 3); we explain its location-recognition algorithm (Section 4); we report on our experimental evaluation (Section 5); we review the most recent developments in the field of indoor localization (Section 6); and we conclude with a summary of our experience to date and some plans for future work (Section 7).

## 2 System Architecture

The high-level architecture of the Smart-Condo™ platform (shown in Figure 1) consists of three layers. The first layer corresponds to the sensor network, in the case of an actual deployment, or the sensor-network simulator, in the case of simulations for pre-deployment configuration planning. The sensor-network bridge component feeds the collected (or simulator-generated) data to the middle data-storage layer, using the MQTT[1] protocol. The top layer includes a variety of analyses and visualization tools for the purpose of extracting and communicating useful information to clinicians. These tools may rely on the archived data, accessed through a set of APIs supported by the data-storage layer, or on the run-time data accessed through a special-purpose client listening to the MQTT stream.

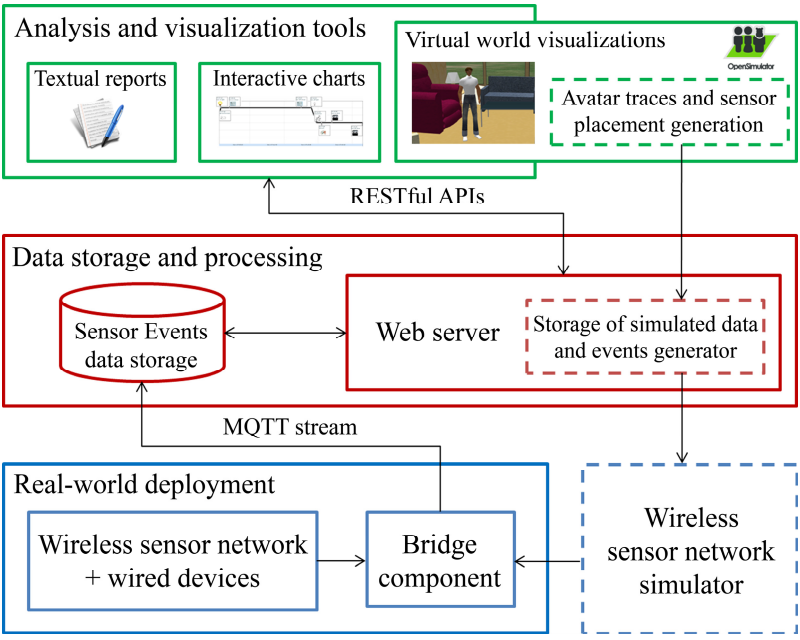


**Fig. 1.** The Smart-Condo™ Architecture

With the Smart-Condo™ architecture we provide two layers of abstraction to (i) flexibly integrate multiple (types of) sensors as necessary and (ii) to provide a range of services for health-care purposes. First, we have adopted a special-purpose bridge component, which already supports various adapters for collecting readings from different sensor protocols and produces as output an MQTT stream. Thus, the introduction of a new type of sensor involves the development of an intermediary software component that can read the sensor readings and communicate them in

[1] A lightweight publish/subscribe protocol, `http://mqtt.org`

MQTT. Second, we have developed a layer of REST APIs[2] through which the sensor data is accessed by our analyses and visualization tools. In addition to a typical web-based 2D visualization component, our toolkit includes a 3D virtual world, OpenSim[3], where an avatar driven by the activity information inferred by the sensor data can "replay" the occupant's activities in the real world. In the following subsections we discuss the crucial elements of the architecture in more detail.

### 2.1 The Sensor Network

The Smart-Condo™ platform currently uses (a) passive infrared motion sensors, (b) magnetic reed switches, (c) electrical-current sensors, (d) light, temperature and humidity sensors, (e) bed/chair occupancy pressure sensors, (f) medication-dispensing devices (introduced for the purposes of clinical study at the Glenrose), and is being augmented with (g) RFID readers. Its architecture is not limited to these sensors, but we have been able to produce reasonable results with just the sensors listed here.

The existing variety of sensors can be classified according to the principles of data transmission and available power sources into three categories: (a) wireless nodes with autonomous power supply (*e.g.*, motion sensors); (b) devices wirelessly transmitting their data, but powered from the power line (*e.g.*, electrical-current sensors); and (c) devices that require wiring for both power supply and data delivery (*e.g.*, medication adherence device). If a quick, low-labor deployment is necessary (as stipulated by the EvAAL competition rules) the sensors most favored are primarily motion sensors due to their independence of the apartment infrastructure. Other sensor types (reed switches, pressure sensors, RFID readers, etc.), although they may not need much cabling, tend to be more labor intensive because they require careful placement and embedding in the available furniture. Having chosen the motion sensors as the bare hardware minimum for a new deployment, we confine the discussion to a sensor network of wireless nodes only.

Each wireless node consists of a low-end microcontroller, a low-power radio transceiver, a battery and one or more sensors. In case of long-term deployment, the requirement to minimize energy consumption of the wireless nodes becomes crucial. For this purpose, wireless nodes buffer their readings until either a certain amount of time has passed since the last transmission or until the buffer is full (whichever occurs first). Buffering, as opposed to immediately transmitting sensed data, reduces the node's energy consumption, because every data transmission incurs the costs of (a) waking up the radio and preparing it for transmission, and (b) transmitting not only the actual sensed data and its timestamp but also a set of necessary packet headers. However, buffering complicates the processing of the readings since observations that were made at the same time by different nodes may be transmitted at different times, resulting in observations arriving out of order.

---

[2] The REST architectural style for web-based applications is described in Roy Fielding's 2000 thesis, `http://www.ics.uci.edu/~fielding/pubs/dissertation/top.htm`

[3] `http://opensimulator.org`

### 2.2 Bridge Component and Sensor-Readings Processor

The bridge is a hardware/software component with a variety of adaptors, through which data from different types of sensors and protocols can be collected. It enables us to integrate standard sensing devices, *e.g.*, electrical-current ZigBee-compliant sensors, along with custom-made and specifically programmed nodes running PicOS [2], for which we have developed our own PicOS-to-bridge adapter. A diverse landscape of other standard lower-layer protocols, *e.g.*, Bluetooth, ANT+, Z-Wave, as well as various proprietary protocols, can be equally easily integrated. The bridge's additional functionality is to provide a layer of data-storage redundancy, *i.e.*, an extra lightweight database in which all the raw sensor readings are stored prior to filtering and processing. This data storage constitutes an intermediate data format which can be used for tracing and debugging errors that may occur at different levels of the system architecture during execution.

To address the out-of order observations introduced earlier, the bridge uses a sliding-window buffer to (re)order raw sensor readings by their reported timestamps. Every $t$ seconds ($t$ is configurable), all readings in the buffer with a timestamp up to $t$ seconds earlier than the current (wall clock) time are published to the MQTT broker.

The broker acts as a message queuing and filtering mechanism for clients that either (a) publish information updates under certain topics, or (b) subscribe to receive updates on topics of interest. Having decoupled producers and consumers of data, the MQTT-based middleware allows for greater modularity and heterogeneity to the extent that various devices with TCP/IP networking functionality[4] can be plugged into the system at the MQTT broker level, bypassing the bridge. In the current infrastructure, however, the bridge is an MQTT gateway for all the wireless devices. It publishes sensor readings under a pre-determined topic, and a special data-importing module, subscribed to that topic, gets notified and feeds the acquired data into a back-end implemented on top of WebSphere Sensor Events[5].

Readings imported into Sensor-Events are being processed by a set of database triggers and stored procedures, which are activated whenever a new entry is created in the raw sensor readings table. The triggers perform noise filtering and call stored procedures implementing activity-recognition logic. More specifically, in this paper we focus on "location and movement recognition", *i.e.*, determining the occupant's location, based on motion-sensor readings mainly. The Sensor-Events platform (with the underlying DB2 server) brings a benefit of implementing stored procedures in procedural SQL or Java, the latter being our language of choice for creating procedures of unconstrained flexibility and complexity.

The inferences generated as the application of location- and activity-recognition logic to the collected data through the triggers are stored in a separate database table, in a clear, client-independent format. These parsed readings are currently being used by a visualization client, implemented in the OpenSim virtual world, but they may be accessed by any type of a client via a call to an intermediary REST web service.

---

[4] Due to the fact that the MQTT is implemented on top of TCP/IP stack.

[5] WebSphere Sensor Events (`http://www.ibm.com/software/integration/sensor-events`) provides the core platform for developing, deploying, and managing end-to-end solutions that exploit the new real-world information available from networked sensors.

### 2.3 Virtual World Visualization

The virtual-world animation of the patient's activities has been developed as one of the visual-analysis tools of the Smart-Condo™ platform and addresses privacy issues associated with video surveillance. The generated animations provide sufficient level of detail comparable with video recording, yet have lower fidelity and are intrinsically non-personified. They are viewable both in real-time, *i.e.*, caregivers may monitor the avatar's actions and thus implicitly monitor the patient's actions as they occur, or off-line, *i.e.*, the caregivers can request a playback of a period in a patient's day based on the data stored in the Sensor-Events database. Figure 2 shows two alternative views of the 3D model of the apartment and the avatar (views are fully customizable).

**Fig. 2.** Alternative views in the virtual world

The real-time virtual-world simulation of the patient's activities is generated as follows: when new sensor readings arrive to the Sensor-Events database, a special-purpose service is triggered to push specifically preformatted commands to the OpenSim server. The server parses the commands and updates the states and/or positions of the patient's avatar and the virtual objects corresponding to the furniture and appliances in the patient's real environment accordingly. The playback mode has to be initiated by the user who can select a desired time span and speed of replay; after the user request is issued, the rest of the procedure follows the same execution path as in the real-time mode.

## 3 The Smart-Condo™ Simulation Platform

Originally we conceived the virtual world as an environment for mirroring real-world activities. However, we have since expanded its functionality towards a simulation-based testing methodology, to support the planning of new potential deployments of the Smart-Condo™ platform. This became possible once we had developed virtual models of the real sensors. This functionality is especially important for accurate representation of motion sensors. The detection range of each motion sensor has a

complex volumetric shape; as such, estimating the detection capability of a sensor at a given position and orientation using 2D placement tools is a challenge.

Overall, our simulation methodology involves the following sequence of steps.

1. We build a model of the deployment space in the virtual world based on CAD drawings and any additional information about furnishings, appliances, etc.
2. Next, we place virtual sensors in this model, following the same principles and practices that one would adopt to place the real sensors in the real world.
3. At (simulation) run time, these virtual sensors are triggered by the avatar. This interaction is tracked by in-world tools and converted into artificial sensor events.

In this procedure, the virtual world is used to generate realistic action traces and corresponding sensor data. Through the virtual-world client, the avatar can be controlled by a user to perform a sequence of activities. The avatar is equipped with an "action-tracking" device, which records movement, sitting/standing posture, and interaction with other virtual objects (opening/closing doors, switching on/off light switches, etc.) as a sequence of <time, action, location> tuples. The generated action log is used as the ground truth trace at the evaluation stage. At the same time, the virtual sensors generate their own events as they are programmed to mimic the real sensors behavior. For example, the virtual motion sensors sense collision events when a moving object penetrates the corresponding 3D shapes; the virtual RFID reader is able to identify different avatars located within the reader's range as if they were wearing RFID tags. All the virtual sensor readings are collected in a separate log as <time, sensorID, sensorReading> tuples. These sensor readings are further propagated through the sensor-network simulator, which helps to more accurately model the operating environment.

### 3.1 Wireless Sensor Network Simulator

Using PicOS substantially simplifies the software development for the low-end hardware used in wireless sensor networks through the use of its source-level simulator, Virtual Underlay Emulation Engine ($VUE^2$) [4]. $VUE^2$ implements the PicOS API and simulates many of the hardware components. It takes into account location and movement of nodes (if applicable), simulates wireless propagation characteristics and noise, and thereby makes it possible to determine data loss and delays rates anticipated in the real environment. The development of the PicOS-to-bridge adapter enabled us to integrate $VUE^2$ with the Smart-Condo™ system and, therefore, to easily experiment with new deployments in simulation mode.

Whenever any of the simulated sensors are triggered, a command specifying that event is sent to the $VUE^2$ simulator, which, in turn, invokes the event handler in the PicOS application for the particular sensor. From that point the simulated node registers the observation and eventually sends it to the bridge. As long as the wireless nodes used are running PicOS, this model follows closely the operation of the real sensor network, since the same code runs in the $VUE^2$ simulator and on the actual nodes. After the sensor reading has reached the bridge component, it is processed exactly the same way real readings are processed, including being published to the MQTT broker and eventually reaching the Sensor-Events database.

### 3.2 Closing the Loop via the Virtual World

The location estimates generated by the localization component (and activity inferences produced by activity-recognition components) are stored in a separate database table. Through an API, the virtual-world client can access them and convert them into a corresponding set of movements and actions for the avatar. That is, the virtual-world controller extracts and processes the readings, and then sends commands to relevant in-world objects to represent each action. Effectively, this procedure constitutes the playback visualization mode (subsection 2.3). Through this simulation path, we can directly compare the original avatar trace from the ground truth log to the trace replayed by the virtual world. Based on their differences, we can assess the accuracy of our monitoring infrastructure.

Given the negligible cost of placing virtual sensors in the virtual world, a variety of alternative placements can be experimented with; by comparing their relative location-recognition accuracy, the configuration with the most accurate anticipated location recognition may be selected for deployment.

This *closed-loop* development and refinement process enables us to perform experiments that systematically evaluate the accuracy of the inhabitant's activity record captured by the architecture and the capabilities of the assumed sensors, *before the actual deployment*. Experiments that involve trial runs with participation of human subjects are cumbersome to organize and difficult to assess. The virtualized alternative allows for arbitrary experiments prior to deployment (to reach a desired level of precision) and allows insights into alternative deployment strategies or alternative sensor technologies that best capture the needs of the client.

## 4 Location Recognition

Once the system hardware is deployed, the location of the condo occupant is inferred from the readings of motion sensors, light switches, reed switches on the doors/cabinets, and pressure sensors in the bed or chairs if the occupant is interacting with the respective elements of the furniture during movement. Location recognition can be improved with RFID readers, if the occupant wears an RFID tag. If the installation time is limited and it is preferable that furniture is kept intact (as was the case with the EvAAL competition), motion sensors become the minimal set of devices necessary for localization.

Motion sensors incorporated in our platform are commercially available passive infrared sensors chosen for their miniature size, fairly wide area of detection and low energy consumption [13]. Being passive, they do not emit infrared light but rather collect incident infrared radiation from within the coverage area. Thus when a moving object with temperature higher than that of the background enters this area, the sensor will detect an increase in the amount of radiation. The output of these sensors is therefore mapped to binary: 0 for no motion, and 1 for motion detected anywhere within the detection area; as a result, given a single sensor, the position of the moving object cannot be discerned with any higher precision than the "radius" of the sensor footprint.

If the localization accuracy is of crucial importance for the deployment, we opt for the sensors with the smallest available detection area. Besides minimizing this parameter, another way to improve localization granularity is to have the sensor footprints overlap. In this case, the floor space is segmented in a number of polygons each one annotated by a bit vector, a 0/1 in the $n_{th}$ position of this vector signifies that the $n_{th}$ motion sensor covers/does not cover the polygon. Hence, the bit vector is a "signature" of the motion-sensor readings that are expected to occur if a person steps in the corresponding polygon. Accordingly, the sensor placement that yields no overlap becomes a particular case of this assignment since each sensor covers a single polygon and thus at any given time of trace execution a signature of readings can contain at most a single 1 and the rest 0's unless the sensors are malfunctioning.

Apart from better localization granularity, the strategy of overlapping sensor footprints introduces synchronization issues. Ideally, to properly fuse the data from two sensors triggered by the same motion event, they must (a) arrive at the bridge at the same time and (b) be associated with identical timestamps. As discussed in section 2.1, buffering on the nodes causes the readings to arrive out of time order. This problem is alleviated by the sliding time window during which the readings are properly reordered and the expired readings are discarded. However, using buffering and a sliding window is acceptable only if the location estimates are not required immediately. When real-time performance of the system is important and energy conservation can be sacrificed (both are the case for the EvAAL competition which lasts only 3 hours for each competitor), the platform can be configured to deliver the sensor readings "instantaneously", *i.e.*, there is no buffering on the nodes and the only added time is for wireless transmission of the packets to the sink.

Having excluded buffering, we still have to work around the second requirement since there is no global synchronization across all nodes. Each node has its own internal clock, and the global timestamps at the bridge are inferred from the time of packet arrival and two timestamps reported: local time on the node when the packet was built and local time when the event was registered. If the first-time delivery of the packet was unsuccessful, the node will retransmit the same packet with an updated timestamp of its generation, accounting for the round-trip retransmission delays. However, there is no account for the usually unpredictable (due to medium access protocol behavior) one-way transmission time and extra processing times at the bridge: if two simultaneously issued packets travel in a crowded wireless environment or arrive at the bridge when it is under high load, they may end up with quite different global timestamps.

To better understand the trade-off of improved localization granularity *vs.* complications in data fusion and find the best suited for real-time deployment, we apply the closed-loop testing methodology to both overlapping and non-overlapping schemes of sensor placement.

## 5 Experiments

In preparation for the EVAAL competition, we systematically evaluated the localization accuracy of the Smart-Condo™ system through simulation. For each <time, action, location> tuple, as logged by the action-tracking device collecting the avatar traces, the error is defined as the distance, in meters, between the avatar's

position, and the position of the corresponding action inferred by the localization component for the given timestamp (interpolated, if timestamps do not match). For the experiment as a whole, we consider the average error, as well as other descriptive statistics, such as standard deviation and error distribution. Assuming the sensors are all functioning properly, these metrics can give us an idea of the accuracy of the processed avatar locations, with respect to the original sensor readings. If a sensor is misbehaving, on the other hand, these readings can help us identify the malfunctioning sensor by returning a larger-than-expected error value for that sensor.

### 5.1 Simulation Results

During setup, the Smart-Condo™ localization component relies on knowledge of (a) the architectural diagram of the deployment space, (b) volumetric coverage models of the motion sensors, and (c) the coordinates and mounting angle of where the motion sensors have been placed to construct a special-purpose map of the space.

In preparation for the competition, we used the architectural diagram of the Living Lab[6] located in Madrid, Spain. From the four available types of passive infrared sensors varying in coverage area, we chose the one with the smallest detection range, that is, the spot type with footprint of 2x1.4m in cross-section at 2m away from the sensor. The more detailed inspection of the sensors datasheet revealed that the detection area of a sensor represents a grid of tiny detection zones and non-sensing strips. The spot type has the most regular and dense pattern of detection zones of all the sensors considered for deployment. Its coverage is well approximated by a rectangular-based pyramid; this pyramid is the geometric representation used by both the simulations in the virtual world and our localization component.

Due to installation time constraints in the competition, we narrowed down the variety of mounting techniques and opted for the mounting on the ceiling with sensors facing the floor in a way that the pyramid base is parallel to the floor plane. Therefore, the 2D sensor coverage map of the space becomes a grid of rectangles that may vary in orientation. Note that this type of mounting is not always possible or preferable. *E.g.*, in previous deployments we have attached the sensors to the walls, hence, the projections of the pyramids on the floor are complex polygons (trapezoids in most generic cases) and the sensor map as a whole is highly irregular. This is when the virtual-world-based planning of deployment becomes an invaluable tool in terms of ease of operation, expressiveness and automation of otherwise manual tasks.

The next stage of simulation is the generation of alternative deployments. As we mentioned before, we focus on the two most important scenarios: overlapping and non-overlapping placements. In both cases, the objective is to minimize the number of sensors while still fully covering the space. Another alternative placement we would like to consider arises as we relax the latter requirement and cap the number of sensors at the fairly small but still reasonable number that covers at least 50% of the space. By this additional placement we would like to test (i) whether there exists a monotonic relation between the localization accuracy and the density of the deployed

[6] The floor-plan of the Living Lab
`http://evaal.aaloa.org/images/LL-coordinates.jpg`

sensors, and (ii) whether the less crowded wireless environment will have noticeable effect on the accuracy. This third alternative deployment is also important if we consider the case when some number of the sensors prepared for the competition cannot properly function and there are no extra devices for a backup.

Therefore, we consider three placements (a) overlapping with 30 sensors, (b) dense non-overlapping with 22 sensors, and (c) non-overlapping with significant gaps in coverage with 13 sensors, shown in Figure 3.

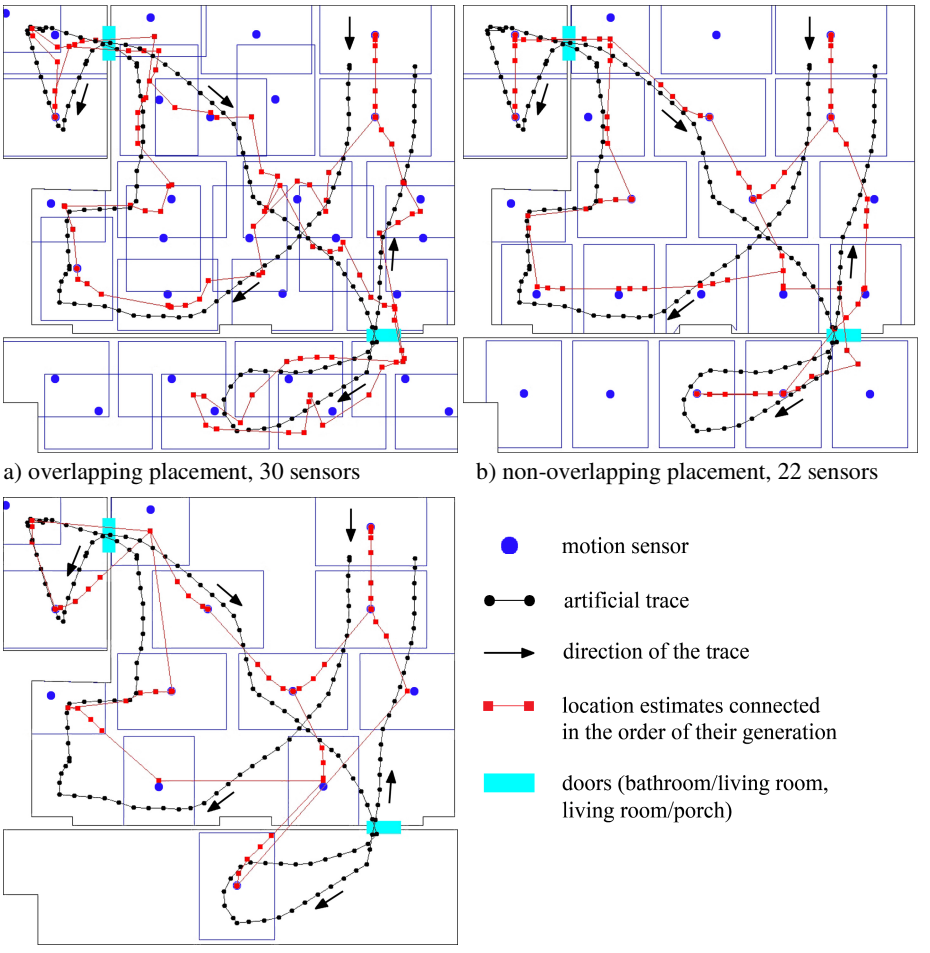


**Fig. 3.** Three alternative placements; an example of artificially generated trace execution

As our methodology suggests, we generated a number of artificial traces; each of them has been tested against all three placements under identical conditions; the results of our simulations are presented in Table 1. Figure 4 displays the error distribution for each of the tested placements.

**Table 1.** Descriptive statistics for three types of sensor placement

| Description | # of sensors | Average error, m | Standard deviation, m |
|---|---|---|---|
| Overlapping, full coverage | 30 | 0.5286 | 0.3155 |
| Non-overlapping, full coverage | 22 | 0.6127 | 0.3269 |
| Non-overlapping, partial coverage | 13 | 1.1619 | 0.9232 |

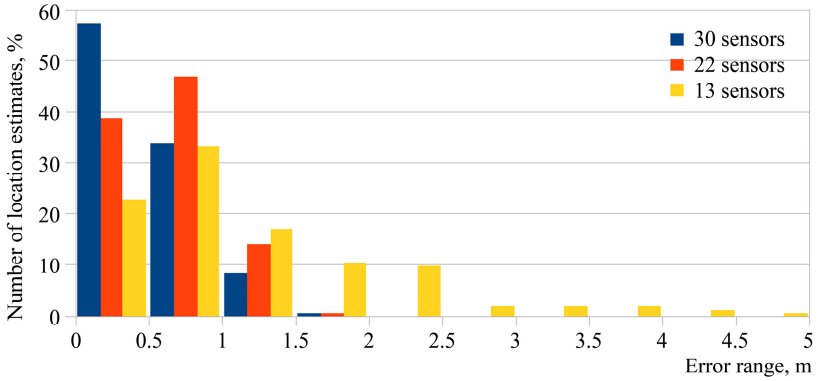


**Fig. 4.** Error distribution for three types of placements used in simulations

Overall we observe that the overlapping placement shows better results, as expected. However, it is worth noting that despite of our best effort to realistically model all non-software aspects of the system operation (*e.g.*, signal propagation in a wireless environment), certain issues intrinsic to overlapping sensor placement have not been properly addressed and only became evident during real-world tests.

More specifically, we noted two problems in the way that sensors get triggered: (i) the sensor will output 0 even if the person is within the coverage area but is completely still; (ii) the signal is rapidly oscillating between 1 and 0 (as fast as 10Hz) during continuous motion within a single sensor coverage area. These problems were not addressed in the simulations due to the basic assumption that as long as the avatar's location lies within the covered area, the sensor outputs 1 regardless whether the avatar is moving or not. The first type of idiosyncrasy is easily addressed by checking whether the avatar's location is not changing (the person is still), and the appropriate model of sensor behavior has been promptly applied. Unfortunately, signal oscillations have proved more intricate to model. Moreover, this behavior significantly undermined our data fusion mechanism, eventually forcing us to abandon the overlapping strategy of deployment.

Another important factor that influenced our sensor-placement decision was the number of sensors that needed to be transported to the remote location, reinforced by the limited time for their installation. In addition, the competition benchmarking tests included trials with two people when only one had to be localized. Considering that our system does not stipulate any wearable equipment (RFID readers have been left out during the initial phase of competition planning), non-overlapping placement

becomes the most appealing strategy due to its ability to distinguish between adjacent sensor footprints and to fairly easily detect anomalies in sensor readings signatures.

On the software side, these simulations proved essential for debugging our localization algorithm. The algorithm's initial coarse estimate is the center of mass of the polygon corresponding to the overlap of the most recently triggered sensors. Subsequent estimates are generated along a physically plausible trajectory until reaching the center of mass of the next adjacent "triggered" area. Figure 3 depicts both the original trace and the results of our localization component calculations.

Closer inspection of Figure 3 suggests possible improvement both in terms of sensor placement and tweaking the algorithm. Note Figure 3a: besides rectangles of various sizes it has a number of polygons with slim protruding parts (as some at the very bottom of the map). The center of mass of such an oddly shaped polygon usually lies in its bigger section causing erroneous estimates when the sensor is triggered from the polygon's "slim" part. On the contrary, Figure 3b shows a generally smoother calculated trajectory (although with a bigger average error) which is, perhaps, due to regularity of the sensor coverage grid. This type of analysis prompts us to continue experimenting with both strategies of sensor placement.

As anticipated, the non-overlapping placement with 13 sensors is roughly twice as bad as the placement with 22 sensors. However, it can also be seen that the coverage grid is not regular and can be much improved even with the given number of sensors. One glaring issue in Figure 3c is that the localization algorithm is not taking into consideration the walls and doors (based on trace transitions from the living room to the porch or the bathroom). We are currently working to address this issue.

From the simulations, we learned that the most reliable placement should be the second type with 22 sensors and assured ourselves that even if a fairly big number of sensors are not working, our system will still be able to generate good results.

### 5.2 Experiment Results

The actual competition deployment was identical to the third placement considered in the previous section. According to the competition protocol, there were 8 tests overall, 4 with one person, 4 with two persons, and 2 more with one person which, however, assessed the ability of our system to detect the presence of the person in a number of predefined *areas of interest* (AoI). In this paper, we discuss the results of the tests with one person only since the other type of tests (with two people) was our first attempt to perform this sort of task and was neither thoroughly tested in the simulations nor was it our priority in this competition. Therefore, we considered 6 trials ("one person" and "AoI detection") for which we are reporting the average error and standard deviation in Table 2.

Note one clear outlier in this table, which is deemed to occur during a period of our system hardware malfunction. That is, trial "path2-1" has an average error of 3.4m whereas all other errors lie between 1.59 and 2.12m. This fact prompted us to exclude this trial from further consideration for reporting our results in this paper. For a total of 1321 location estimates from 5 valid traces, the overall average error is 1.9257m and standard deviation is 1.2423m. Figure 5 depicts the error distribution; 85% of all location estimates generated by our system lie within 3m-error range.

**Table 2.** Descriptive statistics for individual trials, one-person localization

| Name of a trial | Average error, m | Standard deviation, m |
|---|---|---|
| path1-2 | 1.8470 | 1.1372 |
| path1-3 | 1.5906 | 1.0486 |
| path2-1 | 3.3999 | 1.6688 |
| path2-3 | 2.1267 | 1.3809 |
| pathRs-1 | 1.9004 | 1.1210 |
| pathRs-2 | 1.9663 | 1.3406 |
| **Overall** | **1.9257** | **1.2423** |

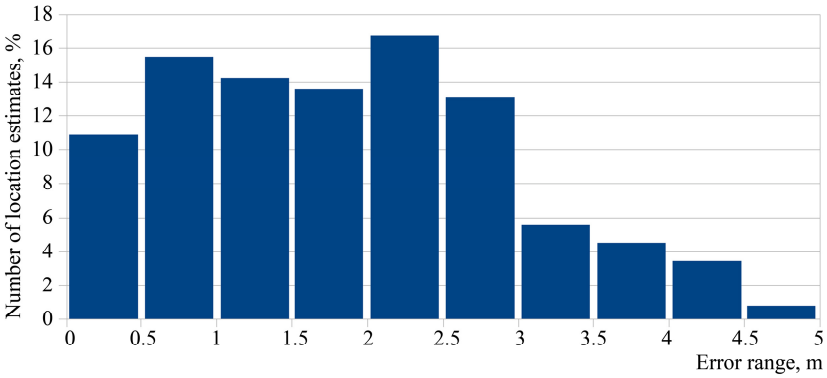


**Fig. 5.** Number of location estimates (%) *vs.* error range for the competition results

It is worth noting that the average error of experimental results exceeds the average error of simulations with identical placement by 66%. There are at least three factors that could have influenced the experimental results that we are aware of, and two of them had significant deteriorating effects: (i) imperfections in sensor installation and (ii) unanticipated delays in generation of location estimates by our system.

With regard to the sensor installation there was a problem with adhesive materials that we used for attaching sensors to the boxes with wireless nodes. These devices are assembled independently so that the sensors or the nodes are easily replaceable. The final custom device consists of a plastic box enclosing a node, and a sensor sitting outside of the box, attached to the node with a wire. When the deployment configuration is known, the sensor has to be firmly attached to the box with adhesive materials. During the competition, one sensor unglued from the box and freely hung on the ceiling causing a lot of misfiring. Figure 6 illustrates how in one of the trials this problem caused confusion of our localization component specifically at this sensor's location. This image was generated during trial "path2-3", and in comparison with images from other trials it clearly shows the effect of this mechanical failure on the operation of the localization component. To support this claim, trial "path2-3" has the largest average error and standard deviation of the five trials (without one outlier) considered for this paper.

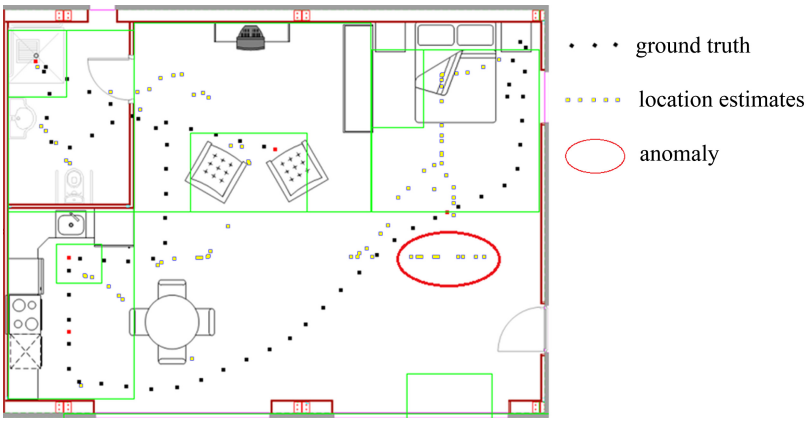


**Fig. 6.** Anomaly in predicting location estimates due to the detached sensor (image generated and provided by the competition organizers)

Another problem that negatively impacted our results is both hardware-related and a matter of the error measure used. Due to the competition requirement for the competing systems to provide real-time updates of data, we chose not to use buffering on the nodes as explained in section 4. Therefore, this functionality was disabled during the simulations and real-world trials that we ran in preparation for the competition. The delays between the moment when a particular reading is registered on the node and when the location estimate is generated consists of the time needed for the packet delivery to the sink, and times for parsing the packet and calculating a new estimate, which all together sum up to 2s in the worst case (based on our real-world experiments). During the competition, however, we encountered certain hardware difficulties, which urged us to use the buffering period of 2s in order to reduce the number of packets being simultaneously transmitted. That is, the discrepancy between the timestamps of corresponding location estimates reported by our system and the timestamps when they were registered by the EvAAL server was about 4s. Unfortunately, the error computation stipulated by the competition implied that the reported location estimates should be associated with the EvAAL-registered timestamps, which also were used as the reference for comparing to the ground truth. The same error computation is used for the results shown above. In order to better understand our results, we also implemented an alternative error computation in which every timestamp reported by our system is matched against the ground truth data, if the exactly matching timestamp is not found, the estimate of the ground truth position is calculated as interpolation of two data points with the closest timestamps. Results of this alternative error computation are presented in Table 3.

Figure 7 compares the error distributions of the simulation experiment and the competition tests with identical sensor placements after applying the alternative error calculation scheme. These results bear more resemblance between each other, therefore, to certain extent proving the value of our methodology.

**Table 3.** Descriptive statistics based on matching reported timestamps

| Name of a trial | Average error, m | Standard deviation, m |
|---|---|---|
| path1-2 | 1.0701 | 0.8726 |
| path1-3 | 0.9147 | 0.5932 |
| path2-3 | 1.2576 | 1.0323 |
| pathRs-1 | 1.2079 | 0.9127 |
| pathRs-2 | 1.4650 | 1.1727 |
| **overall** | **1.2704** | **1.0193** |

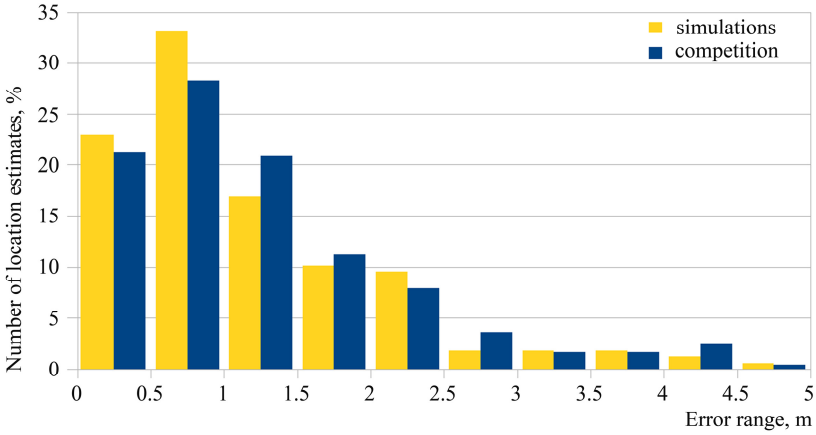


**Fig. 7.** Error distribution for identical placements in simulation and competition

There is one more factor that has affected these results but in a favorable way. One of the competition requirements was for each competitor to interface their system with the benchmarking system. This included optional integration with the contextual events created by sensors already installed in the Living Lab. The flexibility of our system architecture made this part fairly straightforward. We described the provided devices (light switches and a bicycle) in terms of our platform's sensor-specification language. Since other similar devices have already been successfully incorporated in our system (various switches, pressure sensors, etc.), there was no need to modify the localization component in order to use the information gleaned from the Living Lab devices. The integration went smoothly, was completely transparent to our system, and eventually paid off by improving our localization results. We cannot quantitatively estimate the effect of integration since, to our knowledge, every benchmark test contained a number of contextual events. Also, in our simulations we did not include devices unambiguously specifying action location (switches, etc.), since we focused primarily on the use of motion sensors. Therefore, it is not possible yet to draw any conclusive comparisons between simulations and experiments, but we tend to believe that our testing methodology, enhanced with additional models of hardware behavior and ability to detect various anomalies (even mechanical ones), will eventually eliminate costly real-world trials prior to final deployment.

## 6 Related Work

Having reviewed the crucial elements of the Smart-Condo™ platform and its location recognition feature, let us review parallel developments in the EvAAL community dedicated specifically to localization techniques.

The tendency to avoid optical tracking techniques is evident throughout recent work on localization, perhaps, due to similar privacy concerns that motivated our own system development. Thus, two major groups of techniques are (i) localization with wearable equipment, and (ii) ambient localization (*i.e.*, similar to our system). The majority of competitors, however, belong to the first group, and only one other competitor presented a device-free localization system [11].

Localization techniques that rely on wearable equipment most often consist of a network of transceivers (short-range radio signals, ultrasound, etc.) and a device installed on a moving target. A rather popular approach in such systems is received signal strength (RSS) fingerprinting, which was used by three competitors. Grupo TAIS from the University of Seville, Spain, develops a fingerprint-based system comprised of ZigBee devices. Similarly, the LOCOSmotion project from the University of Duisburg-Essen, Germany, relies on fingerprinting collected from Wi-Fi access points and uses a smartphone as a wearable device. Additional information is obtained from an accelerometer embedded in the smartphone. Although such systems are usually easy to deploy or can even exploit the existing infrastructure (most indoor environments already have multiple Wi-Fi access points), the fingerprinting phase can be rather tedious since a database of signal fingerprints for all possible mobile device locations has to be collected prior to localization tests. In addition, this approach is sensitive to any changes in the environment, thus, the created fingerprint database requires continuous maintenance. In order to overcome this limitation, the OwlPS system [7], has an auto-calibration mechanism, which eliminates manual fingerprint collection phase and continuously updates its fingerprint database during execution. The OwlPS deployment is one of the quickest in the competition, comprising of four Wi-Fi access points installed in the corners of the Living Lab. However, with respect to localization quality, the three fingerprint-based systems presented in the competition achieved the lowest accuracy scores among all teams.

The iLocPlus system [12] is an ultrasonic time-of-flight measurement system that comprises of reference nodes and an electronic badge/transmitter worn by the tracked person. Successful localization relies on the line-of-sight between the receiver nodes and the transmitter, therefore, the body of the badge wearer may cause deterioration of localization quality in certain positions. Overall, the accuracy score is better than of our system, however, installation is more time-consuming due to a large number of reference nodes required to overcome (i) the obstruction effects of the body and (ii) ultrasound interference caused by background noise.

The localization system developed by the Centre for Automation and Robotics (CAR), Spain [10], combines dead-reckoning with absolute-position estimation obtained from ambient infrastructure. That is, the wearable unit generates inertial data that translates into position estimates characterized by a distinctively smooth trajectory on one hand, and accumulated drift on the other. To minimize drift effects, the system is enhanced with RFID infrastructure that provides absolute position

references. More specifically, a portable RFID reader is installed on the tracked person and active RFID tags are deployed in the space. This system requires minimum installation effort and has one of the best accuracy scores. However, as in the case with all the systems using wearable devices, it is arguable whether such a solution will become acceptable for everyday use in a typical AAL environment.

Particular to the Smart-Condo™ project is our motivation to keep the system minimally invasive, and therefore we avoid technologies that involve bulky wearable devices. For example, we are currently augmenting our system with the RFID technology; in our setup the readers are embedded in the ambient infrastructure, and the RFID tags are attached to the moving objects (as opposed to the CAR deployment with a portable reader). The tags are lightweight and cheap and can be easily incorporated into a variety of objects, *e.g.*, clothes, a wheelchair, a walker. The main purpose of integrating the RFID technology is to distinguish between the patient and all other people located in the condo. In a clinically-motivated scenario, the patient staying in the condo is visited by a nurse who has her own RFID tag (perhaps, sewn into the uniform). Therefore, we are able to distinguish between the object of localization interest (the patient) and the "disturber" (the nurse) while the patient remains unaware of the surrounding RFID infrastructure.

In this competition, only one other competitor was driven by a similar motivation. The RSS-based device-free localization system from the CPS Group of the University of Utah [11] consists of static nodes deployed along the inside perimeter of an apartment generating an interconnected graph of wireless links. When the person crosses the line-of-sight between any of the links, their baseline RSS values start fluctuating thus indicating a particular location upon fusion of the data from all the links. This system overcomes the typical for RSS-based approaches issues with dynamically changing environments thanks to continuous online self-recalibration. The localization accuracy of this system is among the best in the competition. There are a few shortcomings: a fairly big number of nodes required for high accuracy results (*e.g.*, 33 nodes in $58m^2$), they are powered from the wall outlets which involves extra cabling, and they have to be installed along the walls on a fixed height not exceeding the tracked person height. The latter can be impossible due to existing furniture. On the contrary, our motion sensors can be installed on the ceiling, anywhere on the walls or even underneath a table/desk if we want to detect that specific location. They require no cabling. If a sensor needs to be moved, we only need to change the configuration file since our localization component by default takes into account every possible location and orientation of a sensor in 3D space.

We would like to note that even though our system did not prove to have the best localization accuracy, we managed to showcase the flexibility and interoperability of our architecture. Our system was the only deployment that was successfully integrated with the sensors pre-installed in the Living Lab. It is also worth noting that our system was conceived differently from the competing systems, in that localization is not its sole purpose but merely one of the features supported by the platform. Our work aims to develop a flexible architecture for supporting ambient-assisted living, on one hand, and experimentation with sensor development and sensor-network deployment, on the other. In this architecture, if we replace our existing motion sensor technology, the change will be transparent to the rest of the system.

## 7 Conclusions

The Smart-Condo™ project aims to support people with chronic conditions to live independently longer by developing a platform for unobtrusively observing the activities of a home's occupant (with a variety of sensors) and automatically controlling the home ambience and devices to improve the occupant's experience.

The Smart-Condo™ platform is architected in three layers. The sensor layer is responsible for collecting sensor readings, whether from a sensor-network actually deployed in a home or from a simulator. The middle layer analyzes the collected sensor data to infer the occupant's location and activities. The interactive-visualization layer communicates these inferences in a graphical manner and as avatar-based animations in a virtual-world model of the occupant's home. An important feature of the Smart-Condo™ platform is its support for simulating and evaluating a particular sensor deployment with respect to its inference accuracy. Through this feature, we are able to explore alternative deployments and their relative deployment-cost *vs.* inference accuracy trade-offs. To this end, the avenues for future work include (i) incorporating improved models for simulating the physical sensor behavior, (ii) overcoming the global synchronization issues for overlapping sensor footprints, and (iii) implementing detection of anomalies in sensor readings of various nature.

To date, the Smart-Condo™ platform has been deployed and evaluated three times in three different spaces. The results from the first two deployments were qualitative, because the experiment design did not allow knowledge of the ground truth of the occupant's activities. In the context of the EvAAL competition, we were able to precisely evaluate the utility and effectiveness of the simulation-based deployment-planning feature of our platform. Furthermore, the experiments demonstrated (a) the ease-of-deployment of our platform, (b) its flexibility and extensibility, as it was the single competitor able to integrate pre-existing sensors in the space, and (c) its high-quality (although not optimal) location-recognition accuracy.

**Acknowledgments.** We thank N. Boers, J. Huang, and P. Gburzynski for their work in developing the initial version of the Smart-Condo™ system. We thank L. Liu, S. King, and R. Lederer for their role in the broader Smart-Condo™ project. This research was supported by IBM, NSERC, iCORE and OlsoNet.

## References

1. Abbey, B., Alipour, A., Gilmour, L., Camp, C., Hofer, C., Lederer, R., Rasmussen, G., Liu, L., Nikolaidis, I., Stroulia, E., Sadowski, C.: A remotely programmable smart pillbox for enhancing medication adherence. In: 2012 25th International Symposium on Computer-Based Medical Systems (CBMS), pp. 1–4. IEEE (2012)
2. Akhmetshina, E., Gburzynski, P., Vizeacoumar, F.: PicOS: A tiny operating system for extremely small embedded platforms. In: Proc. of ESA 2003, pp. 116–122 (2003)
3. Boers, N., Chodos, D., Huang, J., Gburzynski, P., Nikolaidis, I., Stroulia, E.: The Smart Condo: Visualizing Independent Living Environments in a Virtual World. In: Pervasive Computing Technologies for Healthcare, PervasiveHealth 2009, pp. 1–8. IEEE (2009)

4. Boers, N.M., Gburzynski, P., Nikolaidis, I., Olesinski, W.: Supporting Wireless Application Development via Virtual Execution. In: International Multiconference on Computer Science and Information Technology, IMCSIT 2008, pp. 853–860. IEEE (2008)
5. Chan, M., Campo, E., Estève, D., Fourniols, J.: Smart Homes — Current Features and Future Perspectives. Maturitas 64(2), 90–97 (2009)
6. Cook, D.J.: Health Monitoring and Assistance to Support Aging in Place. J. Univers. Comput. Sci. 12(1), 15–29 (2006)
7. Cypriani, M., Canalda, P., Spies, F.: OwLPS: A Self-calibrated Fingerprint-Based Wi-Fi Positioning System. In: Chessa, S., Knauth, S. (eds.) EvAAL 2011. CCIS, vol. 309, pp. 36–51. Springer, Heidelberg (2012)
8. Helal, S., Mann, W.C., El-Zabadani, H., King, J., Kaddoura, Y., Jansen, E.: The Gator Tech Smart House: A Programmable Pervasive Space. IEEE Computer 38(3), 50–60 (2005)
9. Hori, T., Nishida, Y., Murakami, S.: Pervasive Sensor System for Evidence-Based Nursing Care Ssupport. In: Proc. of the IEEE Int. Conf. on Robotics and Automation, ICRA 2006, pp. 1680–1685 (2006)
10. Jimenez Ruiz, A.R., Seco Granja, F., Prieto Honorato, J.C., Guevara Rosas, J.I.: Accurate Pedestrian Indoor Navigation by Tightly Coupling Foot-Mounted IMU and RFID Measurements. IEEE Transactions on Instrumentation and Measurement 61(1), 178–189 (2012)
11. Kaltiokallio, O., Bocca, M., Patwari, N.: Follow @grandma: Long-Term Device-Free Localization for Residential Monitoring. In: The 7th IEEE International Workshop on Practical Issues in Building Sensor Network Applications (2012)
12. Knauth, S., Kaufmann, L., Jost, C., Kistler, R., Klapproth, A.: The iLoc Ultrasound Indoor Localization System at the EvAAL 2011 Competition. In: Chessa, S., Knauth, S. (eds.) EvAAL 2011. CCIS, vol. 309, pp. 52–64. Springer, Heidelberg (2012)
13. Panasonic Electric Works, MP motion sensor "NaPIOn" (passive infrared), datasheet, `http://pewa.panasonic.com/assets/pcsd/catalog/napion-catalog.pdf`
14. Skubic, M., Alexander, G., Popescu, M., Rantz, M., Keller, J.: A Smart Home Application to Eldercare: Current Status and Lessons Learned. Technology and Health Care 17(3), 183–201 (2009)
15. Stroulia, E., Chodos, D., Boers, N., Huang, J., Gburzynski, P., Nikolaidis, I.: Software Engineering for Health Education and Care Delivery Systems: The Smart Condo Project. In: ICSE Workshop on Software Engineering in Healthcare, SEHC 2009, pp. 20–28. IEEE (2009)
16. Stroulia, E.: Smart Services Across the Real and Virtual Worlds. In: Chignell, M., Cordy, J., Ng, J., Yesha, Y. (eds.) The Smart Internet. LNCS, vol. 6400, pp. 178–196. Springer, Heidelberg (2010)
17. Woo, K., Ganev, V., Stroulia, E., Nikolaidis, I., Liu, L., Lederer, R.: Sensors as an Evaluative Tool for Independent Living. In: Duffy, V.G. (ed.) Advances in Human Aspects of Healthcare, pp. 612–621. CRC Press (2012)