University of Alberta

Deployment Planning for Location Recognition in the Smart-Condo^{TM}: Simulation, Empirical Studies and Sensor Placement Optimization

by

Iuliia Vlasenko

A thesis submitted to the Faculty of Graduate Studies and Research
in partial fulfillment of the requirements for the degree of

Master of Science

Department of Computing Science

Fall 2013
Edmonton, Alberta

To my parents,
whose love and support never cease.

Abstract

The Smart-Condo^{TM} is a comprehensive platform that aims to provide a variety of services, based on information gleaned from sensors deployed in an apartment, that can potentially improve healthcare delivery. One of our main objectives has been to develop an accurate non-invasive occupant-localization method using passive infrared sensors. In this thesis, we present a simulation framework with which we investigate tradeoffs between the number of sensors and the localization accuracy of our platform. We compare the results of simulations and real-world trials and conclude that our simulation framework is a reliable estimator of the localization accuracy of a particular sensor configuration. We then propose a methodology for planning new deployments that takes into account geometric properties of the new space and the context of occupant's activities. More specifically, we describe a model with the potential to capture typical indoor mobility patterns and formulate a sensor placement optimization problem based on this model. We propose a placement algorithm with near-optimality guarantee. Through simulation-enabled evaluation, we demonstrate that this algorithm generates sensor configurations with localization accuracy superior to that achievable with the same number of sensors placed manually or randomly in the same environment.

Acknowledgements

I would like to thank my supervisors Eleni Stroulia and Ioanis Nikolaidis, whose unceasing energy and incredible adventurousness have been inspiring me to create, improvise, and to challenge myself. It has been a wonderful time to work with them and the broader Smart-Condo team on all the facets of the project: from building a dollhouse that demonstrates basic ideas of the smart-home technology with Robert Lederer and Ken Horne to going overseas and showcasing our system at the indoor localization competition with Ioanis. I would like to extend my thanks to Veselin Ganev who taught me how to use a soldering iron, assemble sensors, and many other things that I have never done before. I thank Meisam Vosoughpour who developed localization and simulation software used extensively in this thesis. I would like to thank the examination committee members Ken Wong, for his unmatched attention to detail, and Nilanjan Ray, for broad challenging questions. I extend my thanks to Chris Rayner, whose support, advice, and feedback have been crucial, and without whom I would have not enjoyed working on this thesis as much as I did.

Table of Contents

List of Tables

List of Figures

Chapter 1

Introduction

The term “Smart Home” refers to a home embedded with sensors, with which to observe the environment and its occupants’ activities, and actuators, with which to automatically control the home ambience and devices to improve the occupants’ experience [6]. Sensor-based systems are a common means of *non-intrusively* monitoring a person’s activity and providing this person, and his formal and informal caregivers, with useful information for making decisions regarding his care [7]. In our work on the Smart-Condo™ project [28] we have been developing a comprehensive platform for addressing this broad research problem.

The evolution of the Smart-Condo™ platform has been driven by two major requirements. Its architecture needs to be extensible in order to support (i) the flexible integration of a variety of sensing devices and (ii) the straightforward development of various analysis tools, as required by the specific clinical scenarios motivating the system deployment. With regard to the first requirement we have been able to employ (a) motion sensors, (b) reed switches, (c) pressure sensors, (d) electrical-current sensors, (e) light/temperature/humidity sensors, and are currently working on integrating (f) RFID readers. The deployed hardware coupled with the appropriate data analysis tools determine the core features of the system, such as location- and activity-recognition, alert generation, home automation, etc.

To date, the Smart-Condo™ platform has been deployed and evaluated in three different spaces. The first deployment took place in 2009 and served simply as a feasibility exercise. The second deployment was used to support discharge planning at the Glenrose Rehabilitation hospital (Edmonton, Alberta, Canada) in the summer of

2011 [34]. A patient, about to be discharged, stays in the Independent Living Suite (ILS) for several days in order for the discharge team to assess the patient's ability to live autonomously. Typically, while patients stay in the ILS, nurses have to periodically check on them for the sake of their safety and well-being, which contradicts the purpose of the stay. The Glenrose clinicians have considered using video surveillance, however, some patients are unwilling to accept this technology out of privacy concerns. We equipped the ILS with non-intrusive and privacy-respecting sensors and analyzed data recorded during the stay of two patients. At the end of each trial, we generated extensive reports and visualizations for the discharge team and thereby enabled the caretakers to make better informed decisions.

Cost of deployment. After the first two deployments we realized that a major factor hindering the widespread adoption of the smart-home technology is the deployment and operational costs involved. There are at least three factors influencing the cost of a new deployment: (a) the cost of hiring an expert who can analyze the requirements and design a sensor placement that satisfies the geometric specifications of the new space and guarantees a desirable performance level, (b) the cost of equipment, and (c) the cost of manual labor needed for installing the sensors and maintaining the complete system (*e.g.,* replacing dead batteries in wireless nodes). Clearly, one way to reduce costs is to minimize the number of sensors necessary for generating data of acceptable quality. Our past experience suggests that the minimization of the number of sensors is challenging since it requires numerous trial runs on the fully deployed system until an acceptable sensor configuration is found. Organizing such trials is cumbersome as they require human participants and, possibly, relocation of *already deployed* sensors (if the initial configuration has not been successful). To alleviate this problem, a workflow for simulating and evaluating a sensor placement under particular deployment conditions in the *pre-deployment* phase has been proposed by Ganev *et al.* [13]. We continue using a slightly modified version of their approach in this thesis.

EvAAL competition. The validity of the aforementioned simulation-based approach was demonstrated through our participation in the indoor localization track of the "Evaluating AAL[1] Systems through Competitive Benchmarking" competition (EvAAL, http://evaal.aaloa.org/) in July 2012 [30]. This became the third milestone for the Smart-Condo™ project. Although our system has been conceived as a multipurpose platform, for the competition we primarily focused on developing an accurate location-recognition method.

The importance of the occupant's mobility data cannot be underestimated as, on a long-term basis, such data can be mined for patterns potentially useful for prevention and diagnosis of chronic conditions [26]. As a short-term benefit, a system with an accurate location-recognition method can substantially improve the occupant's living experience as it can provide home automation (via actuators) or guidance/alert services (*e.g.,* for people with vision impairment) based on the occupant's location or movement trajectory. Therefore, our competition deployment comprised of motion sensors only, more specifically, a wireless sensor network (WSN) of passive infrared (pyroelectric) sensors, and real-time localizing software. Note that the location- and activity-recognition results from the previous two deployments were qualitative in nature, partially because the experiment design did not allow knowledge of the ground truth of the occupants activities. Thus, the competition deployment was the first time when we were able to systematically evaluate the localization performance of our platform in real-world trials.

During preparation for the competition we investigated a tradeoff between the number of sensors (and an underlying sensor placement strategy) and localization accuracy, using extensive simulations. We manually designed three sensor configurations, one of which was eventually deployed at the competition. One of the contributions of this thesis will be to compare the results of the competition deployment to the corresponding simulation results and conclude that our simulation methodology has the potential to reliably estimate the localization accuracy achievable with a particular sensor placement.

[1] Ambient Assisted Living

Automated sensor placement. Despite multiple benefits, the simulation-based testing framework lacks a systematic approach for *designing* candidate sensor placements, which are assumed to be intuitive or suggested by experts. In this thesis we advance our pre-deployment testing methodology by introducing a semi-automated approach for generating sensor placements. Our approach is driven by the fact that "the relative sensor-target geometry can significantly affect the potential performance of any particular localization algorithm" [4]. Therefore, we hypothesize that the geometry of sensor placement can be optimized with respect to the measurements being taken, *i.e.,* localization of the occupant of the indoor environment. We show that it is essential for this approach to incorporate information about both geometric properties of the physical space and the context of anticipated activities performed by the occupant of the space.

Such an enhanced methodology may become the first step towards self-configuration of our system for deployments in new spaces. The crucial implications of this work are the following: (i) we eliminate the need in either expert knowledge or intuitive guesses required for manual sensor placement, and (ii) we may find (in a systematic fashion) a candidate sensor placement with a reduced number of sensors that is yet able to achieve a desired performance level. Therefore, our work makes a promising case for the reduction of the overall cost of a new deployment and, consequently, towards greater proliferation of the smart-home technology.

1.1 Thesis Contributions

This thesis makes the broad contribution of a deployment planning methodology for indoor localization tasks in the Smart-Condo™ and other smart-home-like environments. The specific contributions are as follows.

- We conduct a thorough evaluation of the simulation-based testing framework, *i.e.,* we compare results achieved in simulations and real-world experiments. The analysis of these results grounds and gives insight into the development of a semi-automated sensor placement approach.
- We propose a sequence of primitive image transformations to be performed

on the floorplan of a new deployment space, and develop customizable parsing software that translates said floorplan into a set of versatile data structures describing the environment within a uniform grid.

- The resulting data structures address a number of issues related to the representation of various objects supported by the existing simulation software. Namely, the new grid representation of the space dispenses with the artificial notion of room boundaries (used to determine the obstruction effect of the walls on the coverage models of sensors) and instead allows for arbitrary walls and other obstacles (*e.g.,* columns). Significantly, one of the generated data structures represents movable furniture (*e.g.,* chairs).

- We propose a framework for modeling anticipated mobility patterns based on the grid representation of the space. This mobility modeling is used to generate a heatmap of the space, where the "heat" on a grid cell corresponds to visitation frequency. We further show how these frequencies translate into measures of *coverage utility*, which we use to represent how useful it is to monitor that location with a sensor.

- We define a sensing model of passive infrared sensors based on "quality of coverage", and incorporate properties of the sensor-obstacle geometry into this model (*e.g.,* to address sensors whose coverage extends through the doorways).

- Using the aforementioned coverage utility values and sensing model, we formulate a sensor placement optimization problem: maximizing coverage utility given a limited budget of sensing devices. We then propose an approximation algorithm that finds a solution to our problem within a reasonable time and with strong near-optimality guarantees.

- We finally conduct an extensive simulation-based evaluation of our approach by comparing the optimized placements against placements with the same number of sensors placed manually (assuming adequate expert knowledge) or randomly in the same space.

In summary, a major contribution of this thesis is a novel approach to sensor placement for indoor environments. To the best of our knowledge, the problem of sensor placement optimization, although well studied in a more general context of WSNs, has not yet been covered in the context of smart-home related research.

1.2 Thesis Organization

The thesis is organized in two major parts. In the first part we focus on the specifics of localization with passive infrared sensors and report on empirical studies performed in preparation for and during the indoor localization competition. This part starts with a review of relevant indoor localization sensor technologies in Chapter 2. We continue with a description of the Smart-Condo™ architecture and simulation framework in Chapter 3. Next, we present experimental results obtained in simulations and a competition deployment in Chapter 4.

The second part of the thesis is dedicated to sensor placement optimization. We review research efforts in the broader field of optimal sensor placement for WSNs, relate them to our problem formulation, and introduce background concepts and definitions in Chapter 5. We define an objective function and propose a greedy placement algorithm in Chapter 6. We report the results of the experimental evaluation of optimized placements in Chapter 7. We discuss limitations of our approach and avenues for future work in Chapter 8, and conclude with a summary of the results achieved in this thesis in Chapter 9.

Part I

Localization with Passive Infrared Sensors: Simulation and Practice

Chapter 2

Review of Indoor Localization Technologies

In this chapter we focus on approaches and sensor technologies for indoor localization in AAL environments. In particular, we present fundamentals of localization with passive infrared sensors and discuss advantages and limitations of alternative technologies.

2.1 Basics of Localization with Passive Infrared Sensors

The motion sensors used in our platform are commercially available passive infrared (pyroelectric) sensors chosen for their miniature size, reliable human presence detection and low energy consumption [24]. 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 sensors is therefore 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.

The pyroelectric variety is not susceptible to changes in the background temperature since only the transition in amount of radiation is detected. This specificity introduces a special case that should be considered in the positioning algorithm:

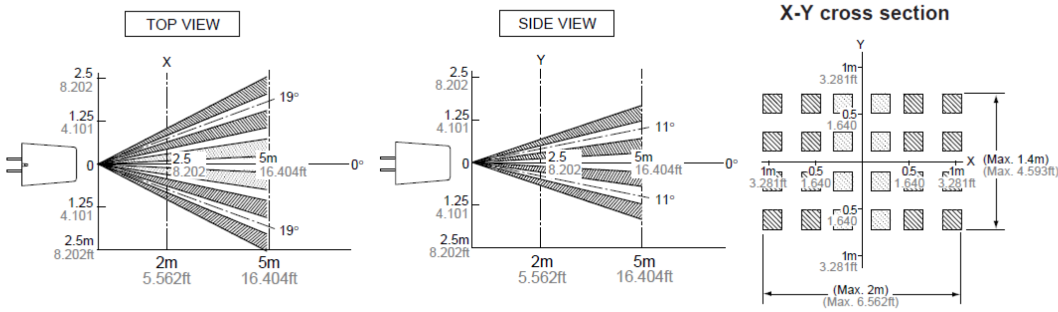


Figure 2.1: The volumetric coverage model of the "spot" sensor (from [24]).

the sensor will output 0 even if the person is within the coverage area but is completely still. One great advantage of these sensors over the majority of alternative technologies used in indoor localization is that the person being tracked does not need to carry additional devices and may remain unaware of the surrounding sensor infrastructure.

The selected sensors come in a variety of volumetric coverage shapes (a cone, a pyramid with a rectangular base), ranges (5-10m) and detection areas (ranging from 3 to 30m^2 in orthogonal cross-section at 2m-height). If the localization accuracy is of crucial importance for the deployment, we opt for the sensors with the smallest available detection area, *i.e.,* the "spot" type with the rectangular footprint of 2 m x 1.4 m in cross-section at 2m-height. Figure 2.1 reveals that the detection area of a sensor is formed by a grid of tiny detection zones and non-sensing strips. The spot type has the most regular and dense pattern of detection zones as well as the most precise boundaries of the sensing area of all the sensors considered for deployment. Therefore, its coverage is well approximated by geometric primitives used in our software. Further on we assume that the sensors are installed on the ceiling and their main optical axis is orthogonal to the floor plane, *i.e.,* the coverage pyramid projects into a rectangle.

Besides minimizing the detection area, another way to improve localization granularity is to have the sensor footprints overlap. In this case, the floor space is segmented into a number of polygons, each one annotated by a bit vector; a 1/0 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 read-

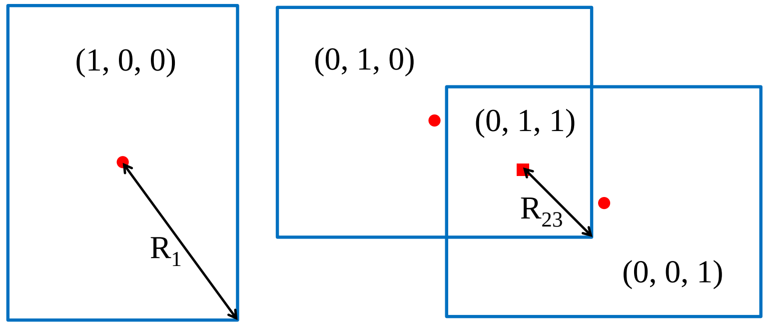


Figure 2.2: Overlapping sensors and corresponding bit-vectors.

ings that are expected to occur if a person steps in the corresponding polygon. Note Figure 2.2, in which the region with a bit-vector (0, 1, 1) has the maximum possible localization error $R_{23} < R_1$ (distance from the center of mass to the furthest corner of the region). A 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 system operation a signature of readings can contain at most a single 1 and the rest 0's unless the sensors are malfunctioning.

Although the strategy of overlapping sensors may significantly improve the localization accuracy, it also increases the cost and complexity of deployment. To better understand the implications of this tradeoff, we apply the simulation-based testing methodology to both overlapping and non-overlapping schemes of sensor placement in Section 4.1.

2.2 Alternative Indoor Localization Technologies

Two recent surveys cover the topics of indoor positioning techniques and a broader topic of localization within WSNs respectively [20, 32]. Both surveys assume that the object of localization is another mobile or stationary node in the network. In terms of AAL scenarios this means that a person to be tracked has to carry a piece of equipment compatible with the rest of the network.

This type of localization involves measuring various properties of signals exchanged between the wearable unit and the surrounding infrastructure. The signals used for measurements can be radio frequency (RF), acoustic, ultrasound, etc.

Within the RF group of signals we will name just a few: ultra-wide-band measurements (UWB), RF identification (RFID), Bluetooth, WLAN (IEEE 802.11), proprietary technologies using ultra-high frequencies, cellular-based for larger buildings, etc. Some of the properties measured – which we will discuss in some detail – include time of arrival, also called time of flight (TOF), time difference of arrival, received signal strength (RSS), angle of arrival, phase difference, etc. Depending on the network topology and measuring principles, various positioning techniques can be applied, *e.g.,* triangulation, multilateration, RF-scene fingerprinting.

All the aforementioned technologies differ in cost, applicability to AAL environments, localization accuracy and by many other parameters. Therefore, we only review parallel developments in the EvAAL community as we believe they adequately represent the variety of localization techniques and are the most relevant to our system. Notably, 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). In the case of the EvAAL competition, most of the competing groups belonged to the first group.

RF fingerprinting. 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 popular approach in such systems is RF fingerprinting, which was used by three competitors. Grupo TAIS from the University of Seville, Spain, develops a fingerprint-based system comprised of ZigBee devices [22]. The major downside of their system is that it reliably localizes only at the room identification level. Similarly, the LOCOSmotion project from the University of Duisburg-Essen, Germany [11], 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.

To overcome this limitation, the OwlPS system [8] 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 that appeared in the EvAAL competition, comprising of only four Wi-Fi access points. However, with respect to localization quality, all three fingerprint-based systems presented in the competition achieved the lowest accuracy scores among all teams.

TOF measurements. The iLocPlus system [17] is an ultrasonic time-of-flight measurement system that comprises of reference nodes and an electronic badge-transmitter worn by the person being tracked. 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 that of our system, however, installation is more time-consuming due to the large number of reference nodes required to overcome (i) the obstruction effects of the body and (ii) ultrasound interference caused by background noise.

Dead-reckoning. The localization system developed by the Centre for Automation and Robotics (CAR), Spain [15], combines dead-reckoning with absolute-position estimation obtained from ambient infrastructure. The wearable unit collects inertial data that are translated into position estimates, characterized by a distinctively smooth trajectory on one hand, and an accumulated drift on the other. To minimize drift effects, the system is enhanced with RFID infrastructure that provides absolute position references. 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 wearable systems, it is arguable whether such a solution will become acceptable for everyday use in a typical AAL environment.

Particular to the Smart-Condo^{TM} project is our motivation to keep the system minimally invasive, and therefore to 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 his or 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 visitors (such as the nurse) while the patient remains unaware of the surrounding RFID infrastructure.

Device-free RSS-based measurements. In the EvAAL competition, one competitor was driven by a motivation similar to ours. The RSS-based device-free localization system from the CPS Group of the University of Utah [16] 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 due 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 large 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

Table 2.1: Comparison of indoor localization technologies by evaluation criteria used in the EvAAL competition.

Participant	Accuracy	Availability	Installation Complexity	User Acceptance	Integrability in AAL	Overall Score
CAR [15]	7.5663	8.2083	10.0000	6.5625	6.8095	7.6953
CPS Group [16]	6.9800	10.0000	3.4367	8.1009	7.7827	7.4531
OwlPS [8]	0.7845	10.0000	9.7067	6.3942	6.9167	6.2882
Smart-Condo™	2.8079	9.0617	1.0000	6.8510	6.9048	5.4128
LOCOSmotion [11]	0.6416	9.9593	1.7667	7.2276	6.7351	5.2344
iLocPlus [17]	3.6363	9.8377	0.0000	4.9038	5.0417	4.8588
Grupo TAIS [22]	0.6718	10.0000	0.0000	5.1122	5.1548	4.2192

even underneath a table/desk if we want to detect that specific location. They are battery powered and communicate wirelessly, thus, require no cabling. If a sensor needs to be relocated, 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.

All of the aforementioned systems were evaluated by a number of criteria on a scale from 0 to 10 for each individual component (for more detail please refer to the online documentation of the competition [2]). The final score was obtained as a weighted sum of individual scores. Table 2.1 presents the results of the competition. Although our system did not prove to have the best localization accuracy in the competition (ranking 4th out of 8[1]), 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 space provided for the competition (integration was optional for all participants). 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. As we have mentioned, our work aims to develop a flexible architecture for supporting ambient-assisted living, on one hand, and experimentation with sensor-network deployment, on the other.

[1]One team is not presented here as they did not wish to disclose their results.

Chapter 3

The Smart-Condo^{TM} Infrastructure

In this chapter we briefly review the crucial elements of the Smart-Condo^{TM} architecture and its support for simulation-based testing.

3.1 Architecture

The high-level architecture of the Smart-Condo^{TM} platform (Figure 3.1) consists of three layers. The first layer corresponds to the sensor network (in the case of an actual deployment) or the sensor-events generator (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 Message Queue Telemetry Transport (MQTT) protocol [1]. The top layer includes a variety of analysis 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 REST APIs [12] supported by the data-storage layer, or on the run-time data accessed through a special-purpose client listening to the MQTT stream.

Sensor Network. Since this thesis has a specific focus on location recognition, we confine the description of the sensor network to those devices we use for motion detection, *i.e.,* passive infrared (pyroelectric) sensors [24]. Each motion sensor is attached to a wireless node. In general, a single node can be equipped with multiple sensors measuring different phenomena. For motion sensing it is reasonable to spatially distribute the sensors with minimal visible cabling, thus, each sensor-node

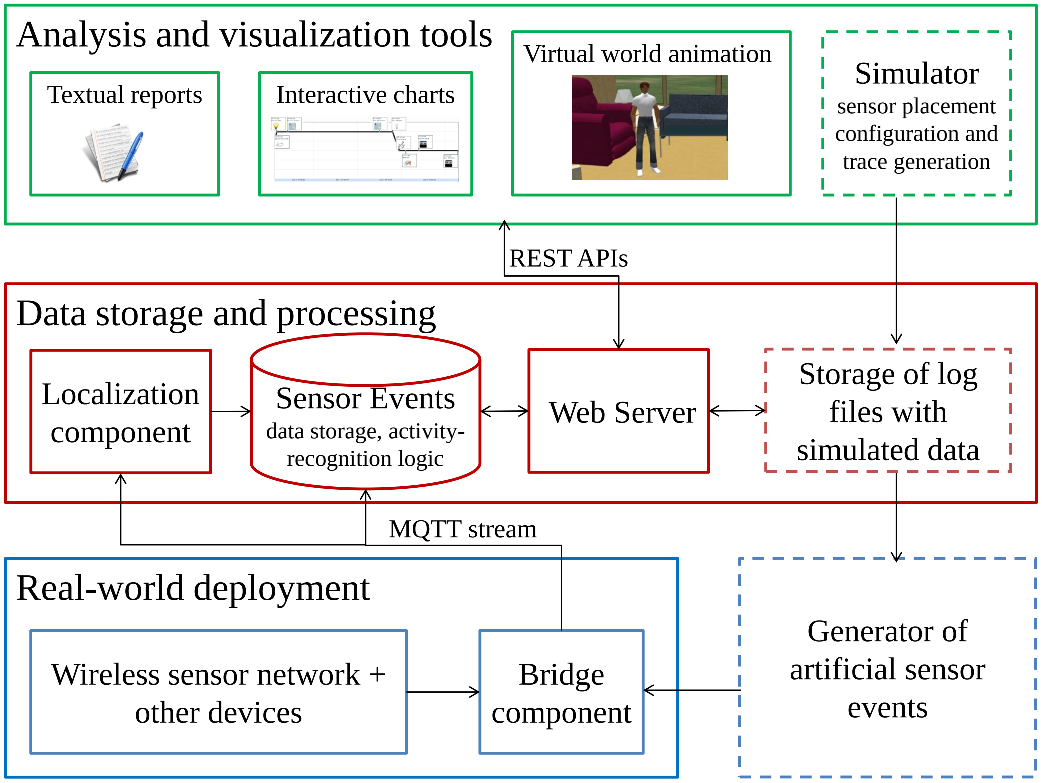


Figure 3.1: The Smart-Condo™ architecture.

pair is atomic and unique in our deployment. The following discussion will use the terms "sensor" and "node" interchangeably. The WSN of motion sensors has a star topology (single-hop), hence, all the nodes transmit the sensed readings directly to the sink node, connected to the bridge component.

Bridge Component. The bridge is a hardware/software component (a plug-computer with Internet connectivity) with a variety of adapters, through which data from different types of sensors and protocols can be collected. It enables us to integrate a diverse landscape of standard lower-layer protocols, *e.g.,* ZigBee, Bluetooth, ANT+, Z-Wave, as well as various proprietary protocols. The currently implemented WSN for motion sensing is comprised of custom-made and specifically programmed nodes running PicOS [3], for which we have developed our own PicOS-to-bridge adapter. This adapter parses raw sensor readings, eliminates duplicates and expired readings, and finally publishes them to the MQTT broker.

Messaging Middleware. According to the publish/subscribe paradigm implemented in the MQTT protocol, 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. The bridge component is an MQTT gateway for all the wireless devices. It publishes sensor readings under a pre-determined topic, and a number of special data-importing modules, subscribed to that topic, get notified of the corresponding events and feed the acquired data into the interested clients, those being a localization component and a Sensor-Events back-end [14].

Localization Component. The localization component receives the raw sensor readings pertaining to location recognition (*e.g.,* from motion sensors, reed switches, pressure sensors). If the reading is generated by a pressure sensor/switch, then the sensor's location unambiguously indicates the position of the actual motion event. Processing of motion sensor readings is not trivial, hence, the following localization algorithm has been implemented within the localization component. The algorithm's initial coarse estimate is the center of mass of the polygon corresponding to the overlap of the most recently triggered motion sensors. The mechanism of coarse estimates is activated every time when the short history of previous moves is unknown or considered unreliable. Once a limited-size buffer of previous locations has been collected, the algorithm starts generating refined estimates along a physically plausible trajectory until reaching the center of mass of the next adjacent "triggered" area. For a more detailed description of the localization software used in our platform please refer to the thesis by Vosoughpour [31]. Figure 3.4c from Section 3.2 illustrates the input and the output of the localization algorithm.

Sensor-Readings Processor and Storage. The Sensor-Events back-end receives raw readings from the whole array of sensors (*i.e.,* not only those used for location recognition as in the case with the localization component). 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-

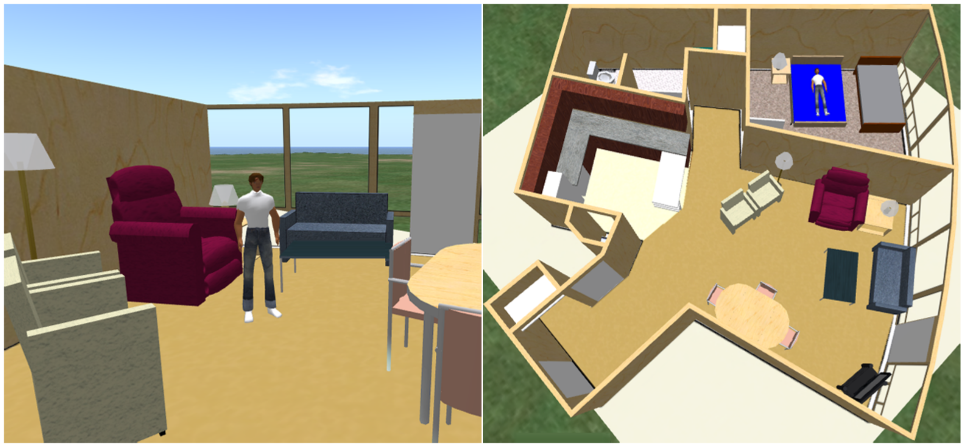


Figure 3.2: Alternative views in the virtual world (the Glenrose setup).

readings table. These triggers perform two types of operations. The first is data cleanup, including noise filtering and general sanity checking. This functionality results in creating a smaller, more coherent set of readings without losing accuracy with respect to the real-world actions that were sensed. The second functionality of triggers is the application of activity-recognition logic to the collected data (implemented as stored procedures). Finally, the generated inferences are combined with the output of the localization component and 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 client via a call to an intermediary web service.

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$^{\text{TM}}$ 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 patients 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 3.2 shows two alternative views of the 3D model of the apartment and the avatar (views are fully customizable).

The animation is implemented as follows. Through an API, the virtual-world client accesses the parsed sensor readings stored in Sensor-Events, converts them into specifically formatted commands, and sends them to the corresponding in-world objects (including the avatar). In a playback mode, the virtual-world controller extracts the readings that fall within a specified time window.

3.2 Simulation Framework

The simulation-enabled debugging process lets us perform experiments that systematically evaluate the accuracy of our localizing software given a specific sensor placement, *before the actual deployment*. Experiments that involve trial runs with participation of human subjects are cumbersome to organize and difficult to assess. The simulation-based alternative allows for arbitrary experiments prior to deployment (to reach a desired level of accuracy) and allows insights into alternative deployment strategies. Our simulation methodology involves the following workflow.

1. We build a 2D model of the deployment space based on floorplan drawings (*e.g.,* Figure 3.3) and input it into the simulator.

2. We generate a sensor placement and integrate the corresponding 2D sensor-coverage map into the model of the space.

3. At simulation run time, the avatar is manually-controlled/scripted to walk through the space. The avatar trace is recorded by the simulator as a sequence of $<$*timestamp, location*$>$ tuples.

4. The sequence of tuples is then evaluated against the sensor-coverage map thus resulting into a stream of artificial sensor events supplied to the bridge component in a format identical to that of the real sensor readings.

The result of the first two steps of the workflow is depicted in Figure 3.4a. That is, every room is represented as a separate polygon, where the impassable obstacles, adjacent to the boundary of the room, are excluded. The room polygons are used to determine the obstruction effect of walls on the sensor coverage footprints. A

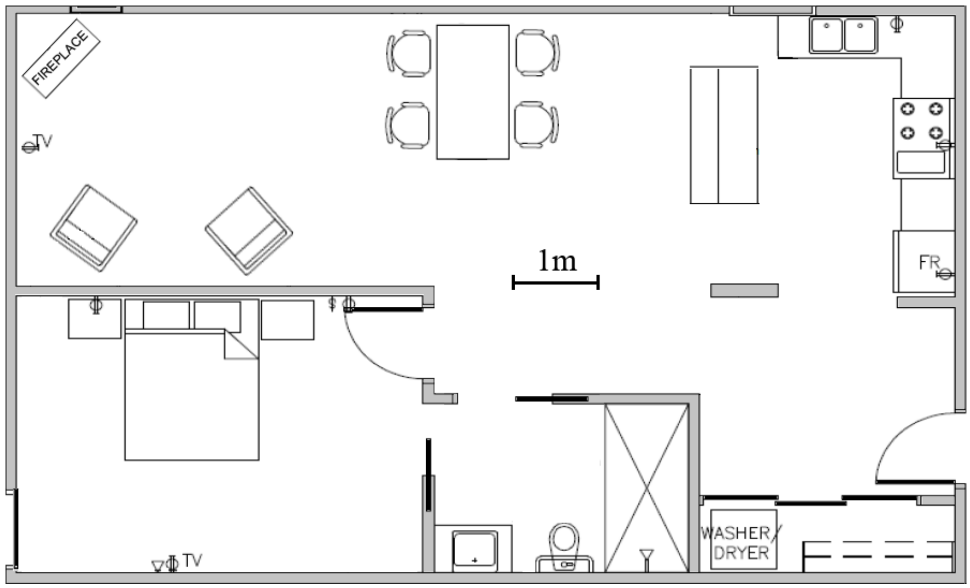
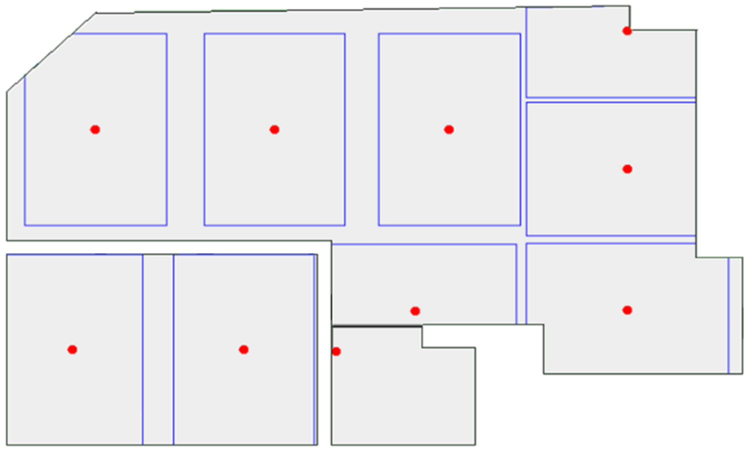


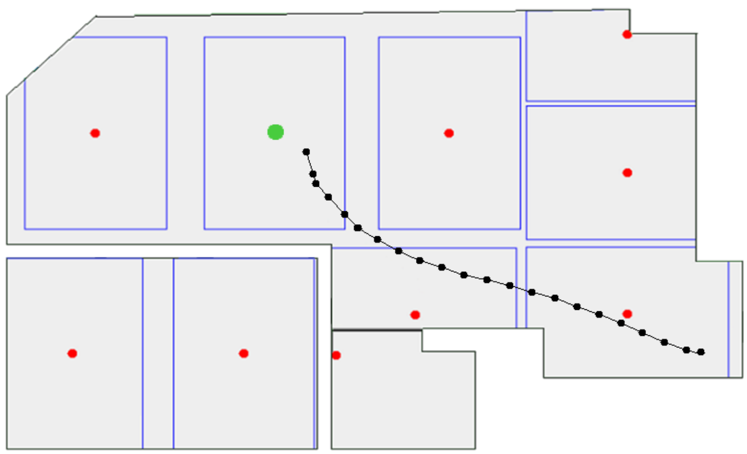
Figure 3.3: The Smart-Condo™ floorplan (grey walls, black doors).

sensor is represented with both a dot at the position of the actual sensor and a polygon depicting a sensor footprint. An example of an artificial trace is shown in Figure 3.4b; each black dot is a <*timestamp, location*> tuple. This sequence of tuples is used as the ground truth at the evaluation stage.

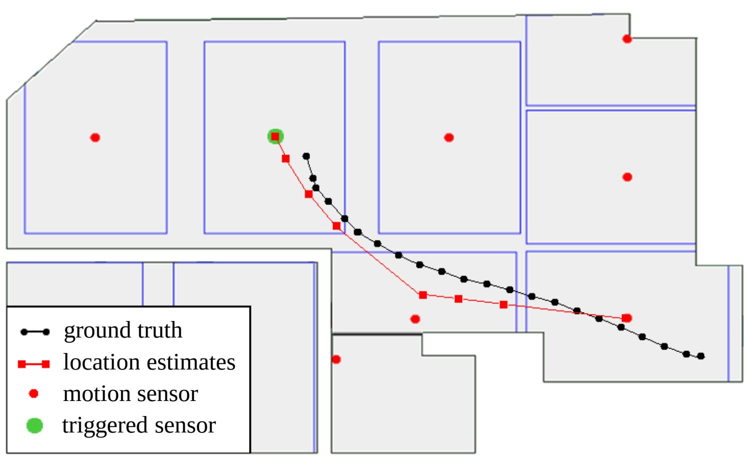
Next, we test *location* from each tuple against the sensors' boundaries, thus determining which sensors should be triggered. This way we generate a stream of artificial sensor events and feed them into the bridge component as if they were real sensor readings. *i.e.,* the localizing software receives a binary vector of sensor readings (1/0 at the n^{th} position representing ON/OFF status of the n^{th} sensor) for each *timestamp*. The location estimates generated by the localization component are directly compared to the original avatar trace from the ground truth log file (Figure 3.4c). Based on their differences, we can assess the accuracy of our monitoring infrastructure.



(a) Model of the space integrated with sensor-coverage map.



(b) A fragment of an artificial trace.



(c) Comparison of the ground truth and estimated locations.

Figure 3.4: Simulation workflow.

Chapter 4

Experiments: Simulation and Practice

We systematically evaluate the localization accuracy of alternative sensor placements through simulation. For each <*timestamp, location*> tuple, as logged by the simulator, the error is defined as the Euclidean distance, in meters, between the avatar's position and the position estimate inferred by the localization component for the given timestamp. To summarize performance, we report the mean of the localization error, standard deviation, and error distribution.

4.1 Simulation

In preparation for the EvAAL competition, we used the architectural diagram of the Living Lab located in Madrid, Spain (Figure 4.1). We consider three placements (a) overlapping with 30 sensors, (b) non-overlapping with 22 sensors, and (c) non-overlapping with significant gaps in coverage with 13 sensors (Figure 4.2).

In the first two placements we attempt to uniformly cover the space with higher and lower density respectively. The third placement arises as we consider the case when the budget of sensors is not sufficient to achieve full coverage. Such a situation may occur if during the competition some sensors fail to function properly and there are no backup devices. With all three placements we would like to test whether there exists a monotonic relation between the localization accuracy and the density of the deployed sensors.

As our methodology suggests, we generated a number of artificial traces; each

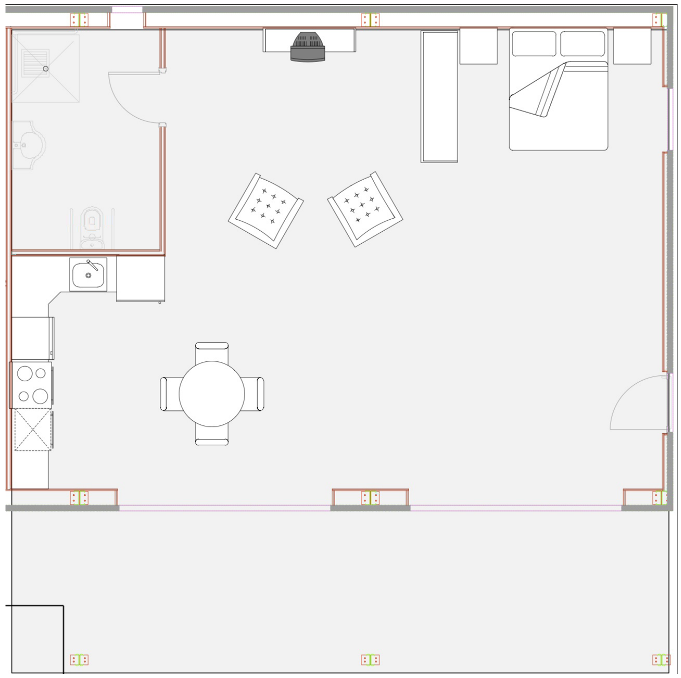


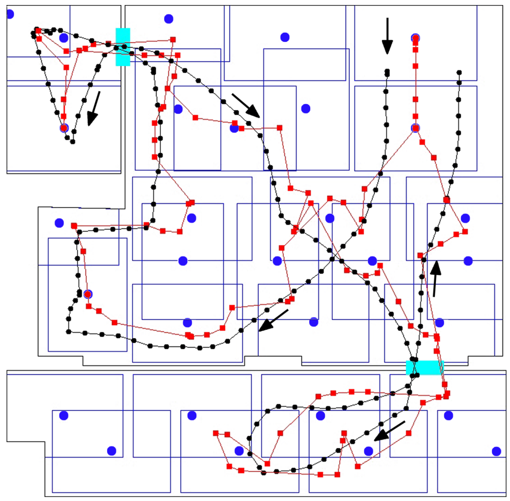
Figure 4.1: The floorplan of the Living Lab.

Table 4.1: Descriptive statistics for three types of sensor placement tested in preparation for the EvAAL competition.

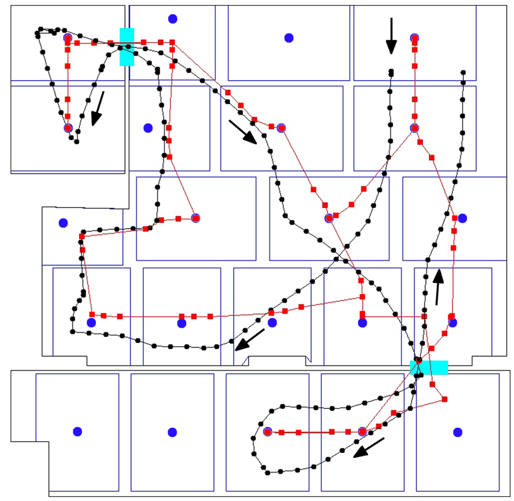
Description	# of sensors	Average error, m	SD, m
Overlapping, full coverage	30	0.5286	0.3155
Non-overlap., full coverage	22	0.6127	0.3269
Non-overlap., partial coverage	13	1.1619	0.9232

trace has been tested against the three placements under identical conditions; the results of our simulations are presented in Table 4.1. Figure 4.3 displays the error distribution for each of the tested placements. Overall we observe that the overlapping placement shows better results as expected. However, this improvement, compared to the second type of placement, is not proportional to the increase in the number of sensors (13% of performance improvement compared to 36% of increase in the number of sensors). Similarly, the second type of placement has 69% more sensors than the third placement but its performance is improved by 47%. This analysis suggests that the relation between the number (density) of sensors and the localization accuracy follows the law of diminishing returns.

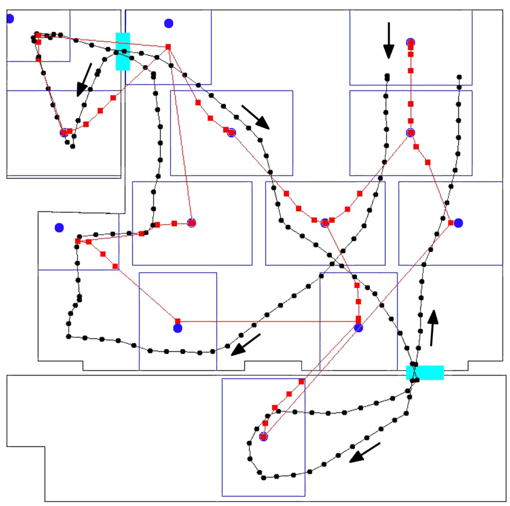
Close inspection of Figure 4.2 suggests possible improvement both in terms of sensor placement and tweaking the localization algorithm. Note Figure 4.2a:



(a) Overlapping placement, 30 sensors.



(b) Non-overlapping placement, 22 sensors.



(c) Non-overlapping placement, 13 sensors.

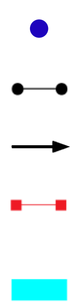


Figure 4.2: Three alternative placements used for simulations.

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 larger section causing erroneous estimates when the sensor is triggered from the polygon's "slim" part. On the contrary, Figure 4.2b shows a generally smoother calculated trajectory (although with a larger 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.

Although the third type of placement was anticipated to perform poorly, it can be

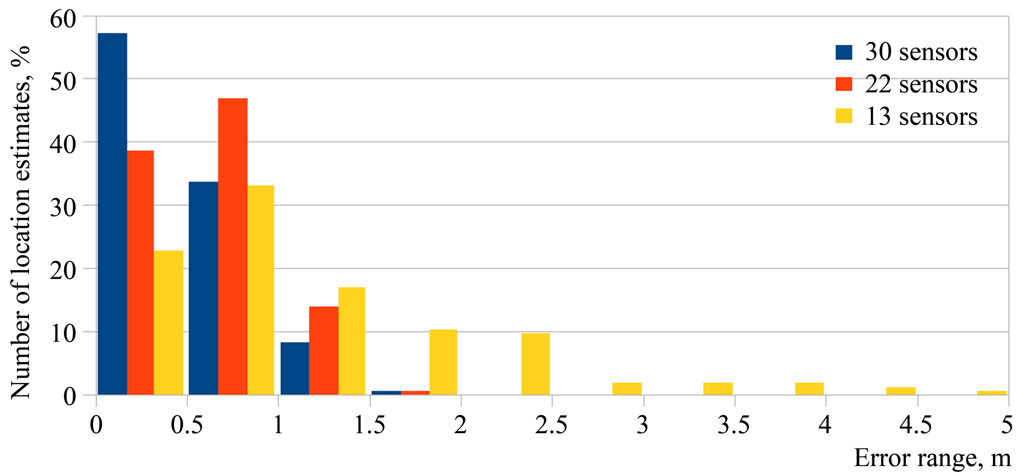


Figure 4.3: Error distribution for three types of placement used in simulations.

seen that the coverage map is not regular, thus suggesting that performance can be improved even with the given number of sensors. One glaring issue in Figure 4.2c is that the localization algorithm does not take into account the walls and doors (based on trace transitions from the living room to the porch or the bathroom).

An important factor that influenced our final sensor-strategy decision was the number of sensors that needed to be transported to the remote location, reinforced by the limited time allowed 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 require 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 unambiguously distinguish between adjacent sensor footprints and to fairly easily detect anomalies in sensor readings signatures. That is, the placement most closely meeting our requirements is the second type with 22 sensors. However, the simulation results for the third placement assured us that, even if a fairly large number of sensors are not working, our system will still be able to generate fairly good results.

4.2 Practical Results

The actual EvAAL competition deployment was identical to the third placement considered in the previous section. According to the competition protocol, there

Table 4.2: Descriptive statistics of the EvAAL competition results.

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**

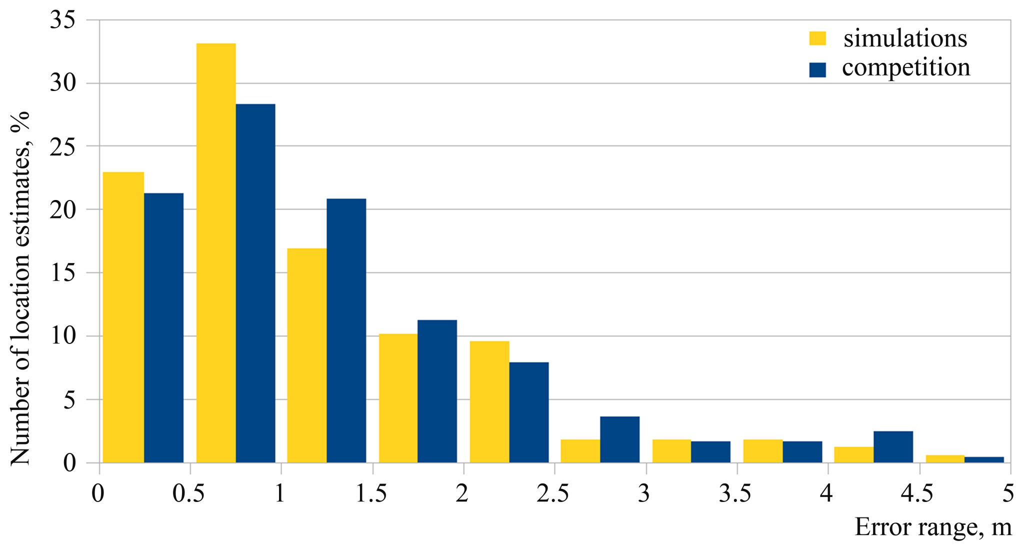


Figure 4.4: Error distribution for identical placements in simulation and competition.

were 8 tests overall, 4 with one person, 4 with two persons, and 2 more with one person that assessed the ability of our system to detect the presence of the person in a number of predefined areas of interest (AoI). In this thesis, 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 5 trials (out of the total of 6 trials for "one person" and "AoI detection" without one clear outlier, which is deemed to occur during a period of system hardware malfunction) for which we are reporting the average error and standard deviation in Table 4.2.

For a total of 1321 location estimates from 5 valid traces, the overall average error is 1.2704m and standard deviation is 1.0193m. It is worth noting that the aver-

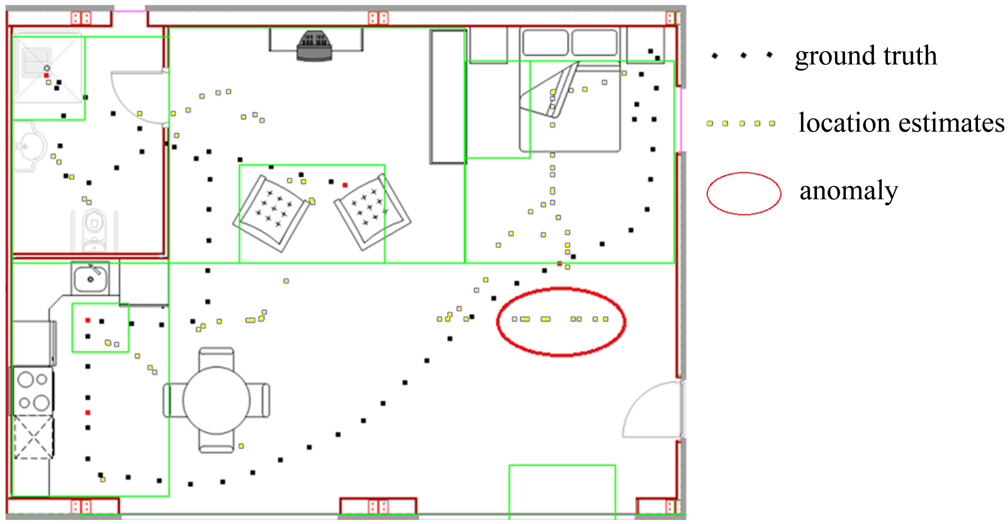


Figure 4.5: Anomaly in predicting location estimates due to the detached sensor (image generated by the competition organizers).

age error of the experimental results exceeds the average error of simulations with identical placement only by 9%. Figure 4.4 compares the error distributions of the simulation experiment and the competition tests with identical sensor placements. Note that 92% of all location estimates generated by our system during the competition lie within the 3m-error range. The distributions resemble each other both quantitatively and qualitatively and, therefore, to a certain extent demonstrate the value of our methodology.

One factor that could have influenced our competition results unfavorably (*i.e.,* 9%-difference with the simulation results) is imperfections in sensor installation. In particular, adhesive materials that we used for attaching sensors to the boxes with wireless nodes proved unreliable. The 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 became unglued from the box and freely hung on the ceiling causing a lot of misfiring. Figure 4.5 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 three traces of type "one person" (first three rows in Table 4.2).

Having learned ample lessons during the EvAAL competition, we see plenty of opportunity for improvement for the next competition-like deployment. However, our current results suffice to show that our simulation-based testing methodology can reliably estimate the localization accuracy achievable with a particular sensor placement, which was our main motivation for participation in this event.

Part II

Sensor Placement Optimization

Chapter 5

Background and Problem Formulation

In this chapter, we introduce background concepts and definitions necessary for the sensor placement problem formulation. We review general approaches for sensor placement optimization and build upon one of the reviewed studies with a closely related problem formulation [33]. This study proposes a theoretical framework for finding placements that optimize *information-oriented* coverage. We extend this framework with a domain-specific method for *information utility* assignment based on the model of anticipated mobility patterns.

5.1 Approaches for Sensor Placement Optimization

An extensive review of various strategies and techniques for node placement in WSNs with respect to their application domains and problem formulations has been conducted by Younis *et al.* [37]. The authors claim that optimal node placement has been proved NP-hard for most proposed formulations of the sensor deployment problem. Most studies, therefore, suggest various heuristics for finding sub-optimal solutions. Another survey [36] is more narrowly focused on indoor monitoring, thus, the authors select a single problem formulation, list a number of applicable optimization criteria, and then review all relevant approaches. Most approaches are iterative (to name just the most popular ones): sequential approaches greedily place one node at a time, simulated annealing probabilistically selects a variable for permutation in each iteration, genetic algorithms generate better fitting populations of candidate solutions. For a better grasp of the state-of-the-art in optimal sensor placement, we refer to the first, broader, survey in the next paragraphs.

Static and dynamic WSNs. Two large groups of techniques can be identified with respect to their application to (i) static WSNs, and (ii) WSNs whose nodes may be dynamically relocated during the network operation [37]. The first group implies that the sensor placement is computed once before the network is deployed and all the network parameters are assumed static throughout its operation time. The second group, besides networks with mobile nodes, also includes networks with generally stationary nodes that may need to relocate if the observed environment changes or a number of nodes fails. Although we recognize the importance of the network's ability to adapt and self-recover, we focus on the static networks as the most suitable and minimally invasive approach for a typical smart-home scenario.

Within the group of strategies for static deployment, Younis *et al.* discern three application-specific classifying criteria: (a) methodology for initial deployment, (b) optimization objective, and (c) roles of nodes in the network.

Methodology for initial deployment. The initial deployment strategy can be randomized or controlled and depends heavily on the scale of the network and properties of the observed environment. A random distribution of nodes is applicable to large-scale networks where careful placement of nodes appears infeasible, *e.g.,* forest fire detection. In this case, such parameters of deployment as node density or redundancy for fault-tolerance can be optimized. An example of a density-oriented study is presented by Toumpis *et al.* [29]. The authors assume massively dense networks and suggest optimizing node density with respect to macroscopic parameters such as information density and traffic flow. It is often the case in such studies that the final optimal set of nodes is a subset of the initial randomized deployment. Thus, the initially unused nodes may later be used to increase fault-tolerance of the network.

In the smart-home type of deployment, we can only afford a reasonably small number of sensors and hence opt for the controlled deployment option and corresponding techniques. In this thesis, we do not address fault-tolerance achievable at the expense of node redundancy as we impose a strong constraint on the number of nodes. However, we would like to investigate this direction in the future.

Optimization objective. The most commonly considered optimization objectives are network connectivity and/or longevity, area coverage, and data fidelity [37]. The category of techniques dedicated to network operation requirements (*i.e.,* connectivity and longevity) is concerned with such problems as communication costs (often in a form of energy conservation), efficient routing techniques, nodes with varying communication range, avoiding traffic bottlenecks, etc. Note that the majority of aforementioned problems apply to multi-hop networks, which intrinsically have nodes with various roles (*e.g.,* base-stations, relay-nodes), and therefore, we prefer to merge this category with the last classification criterion, based on the role assignment of the nodes. On its own, the latter group of techniques focuses on clustering (determining cluster-head nodes) and balancing traffic with respect to node roles and therefore, in our view, heavily overlaps with the rest of strategies for multi-hop static networks.

The most typical coverage problem formulation is cost minimization under coverage constraints, where the surveillance region is approximated by a finite set of grid points [5]. The data fidelity objective can be approached as a data fusion problem and, hence, implies multiple sensors in the vicinity of the monitored phenomenon so that their placement guarantees the fused data to be of some desired quality. Such problems often involve probabilistic models of sensing. For example, for target detection problems, each sensor may be assigned a detection probability, often a function of distance between the sensor and the target. Under these conditions, Wu *et al.* [35] formulated a combinatorial optimization problem with the objective of maximizing the overall detection probability within a given deployment cost. They showed the problem to be NP-complete and proposed an approximate solution based on a two-dimensional genetic algorithm.

A study by Krause *et al.* [18] captures several optimization objectives; they aim to maximize information while minimizing communication costs. The probabilistic models are defined for both predictive quality of the placement and the quality of links between sensors using the data collected during a pilot (non-optimized) deployment. The proposed iterative algorithm first identifies clusters of nodes and then greedily finds a sub-optimal (but with proven approximation guarantees) placement within each cluster.

To relate the aforementioned optimization objectives to our own goals, we remind that the WSN used in the current implementation of the Smart-Condo^{TM} project follows a single-hop model. Two arguments justify this model: (i) the total area of the potential surveillance region is fairly small (in the current setup, a one bedroom apartment of 10.6 m x 6.3 m) so that all the nodes lie within the communication range of each other; and (ii) we are interested in real-time updates, whereas a multi-hop communication model may introduce additional delays due to processing and retransmissions at the relay-nodes. For these reasons, we confine our optimization attempts to maximizing data fidelity, *i.e.,* localization accuracy under a number of nodes constraint, deliberately disregarding communication costs and other network-operation-related concerns due to the simplistic communication model used.

Problems with obstacles and preferential coverage. In the narrower indoor sensor placement realm, most interesting to us are placement problems that assume various obstacles affecting the sensing range of assumed nodes. Dhillon *et al.* [10] suggest to incorporate information about obstacles into sensor probabilistic detection models. They also integrate a model of preferential coverage for areas of high importance. Eventually, they apply an iterative greedy algorithm that places one sensor at a time to the grid point with the lowest confidence level of detection. David *et al.* [9] consider a more specific instance of the problem with sensors whose sensing range is defined by line of sight (*e.g.,* video cameras, pyroelectric infrared sensors) and, therefore, the actual range is derived from application of the raytracing algorithm to the sensor-obstacle geometry. A number of sensor-placement candidates is used to train a genetic algorithm, which finds a (sub)optimal candidate satisfying the coverage constraint.

Perhaps, most closely related to our formulation of optimization problem is work by Wang *et al.* [33]. They propose an algorithm that exploits the fact that in many applications the importance of sensed information varies across the sensing field. Thus, they define *points of interest* (similar to preferential coverage discussed earlier) which they wish to cover optimally. Such a strategy may not yield full coverage but it does maximize the sensed information utility. Similarly, we would like to sparsely distribute sensors in the condo but still be able to sense the most valuable data; in other words, we want to determine prioritized points of interest. We propose a methodology that identifies areas of high interest (mobility) and inherently incorporates geometric properties of the space and information about obstacles. We take into consideration the space floorplan and furniture placement and make realistic assumptions about most common paths used by the condo occupant in his/her daily routine.

Some similarity with our approach can also be seen in the *MavHome* location-aware predictive framework [25], which is another example of a smart-home technology that aims to anticipate the occupant's desires and provide pro-active resource management and on-demand operation of actuators. A critical assumption here is that the occupant travels along most typical path segments between rooms, thus, mobility data can be learned over time and used to predict the occupant's location. In this framework, the sensor placement is assumed from the existing infrastructure; the authors are not concerned with the cost of deployment. Our own speculation based on analysis of this framework is that after sufficiently long learning it might not need immediate updates from the whole array of sensors; a small subset of sensors in most critical zones may be sufficient for a successful prediction. We, however, do not want to rely on a learning phase of system operation due to the history of short-term deployments (*e.g.,* trials with patients at the Glenrose lasted two days). That is, we are strongly motivated to generate high-quality location estimates at any point in time. To achieve this goal we attempt to model mobility patterns anticipated in the indoor environment prior to the actual data collection and propose an algorithm that optimizes sensor placement with respect to the inputted mobility model.

5.2 Introductory Background

Mobility patterns in an indoor environment can be expected to resemble typical road traffic with its bottlenecks, conjunctions, more and less travelled segments. If we have a limited budget of sensing devices and cannot achieve full coverage of the space, a natural solution is to place sensors in the most travelled locations. Essentially, we are not interested in greater coverage but more specifically in maximizing the *utility* of the sensed information. This objective is similar to a problem formulation by Wang *et al.* [33], which we present here and build upon further.

Wang *et al.* use a probabilistic sensing model of target detection that is expressed as a function of distance between the sensor and the target. Every *point of interest* can be covered by multiple sensors and, therefore, if $p_{s_j \to i}$ denotes probability of target detection by sensor s_j at the i^{th} point, then the probability of detection at that point covered by m sensors is calculated as follows:

$$p_i^{(m)} = 1 - \prod_{j=1}^{m} (1 - p_{s_j \to i}) \tag{5.1}$$

Having N points of interest each with *information utility* v_i and a budget of $k < N$ sensors, Wang *et al.* define the optimal placement for information-oriented coverage as such that maximizes the following function:

$$f(k) = \sum_{i=1}^{N} (p_i^{(k)} \cdot v_i) \tag{5.2}$$

Our work extends the proposed theoretical framework by introducing a domain-specific methodology for information utility assignment. In the following sections we define v_i and p_i more specifically fitted for indoor localization purposes.

5.3 Mobility Modeling for Utility Assignment

The Smart-Condo™ localization component relies on knowledge of (a) the architectural drawing of the deployment space, (b) volumetric coverage models of the motion sensors, and (c) the coordinates and mounting angles of where the motion sensors have been placed to construct a special-purpose map of sensor-coverage regions. In a typical scenario, the first two elements are given to the designer of a new deployment in a form of a floorplan and sensor datasheets, whereas the third element is derived as an application of a designer's practical knowledge to the sensor-floorplan geometry.

Passive infrared sensors detect motion in the line-of-sight, hence, the shape of sensor coverage is defined by sensor orientation, obstruction effects of the walls, and furniture. If sensors are mounted on the ceiling, the impact of furniture is negligible (as long as it does not reach the ceiling). Essentially, a geometrically accurate 2D representation of sensor coverage can be generated without any notion of the implied furniture. Such a placement strategy can be seen in the first part of this thesis as well as in a number of other studies focusing on localization with passive infrared sensors [19, 21, 27]. Here, we want to take advantage of contextual information that can be inferred from positions of furniture and basic amenities depicted on the floorplan. Next we apply a number of transformations to the floorplan image that let us retrieve such information.

5.3.1 Floorplan Color-Coding

Consider the floorplan of the Smart-Condo™ (Figure 3.3). We make the following assumption about the occupant movement in this space: the occupant's daily routine consists of a number of short path segments between arbitrary pairs of objects depicted in the floorplan (*e.g.,* bed – toilet, stove – dining table, entrance door – recliner). We first identify all objects of interest (*e.g.,* bed, fridge, stove, sink, toilet) and convert the floorplan into a color-coded image. This conversion is the only manual step of the procedure (easily performed using any graphics painting software). Figure 5.1 shows the result of the conversion and contains five types of colored-coded objects (1) walls (black), (2) doorways/sliding doors (strips of yellow), (3) impassable obstacles that have fixed positions, *i.e.,* all large pieces of furniture (grey), (4) obstacles that change positions frequently, *i.e.,* chairs next to the dining table (green), and (5) areas of interest associated with every object of interest (narrow strips of red next to the articles of furniture).

The fourth type of object (henceforth addressed as *the movables*) has particular

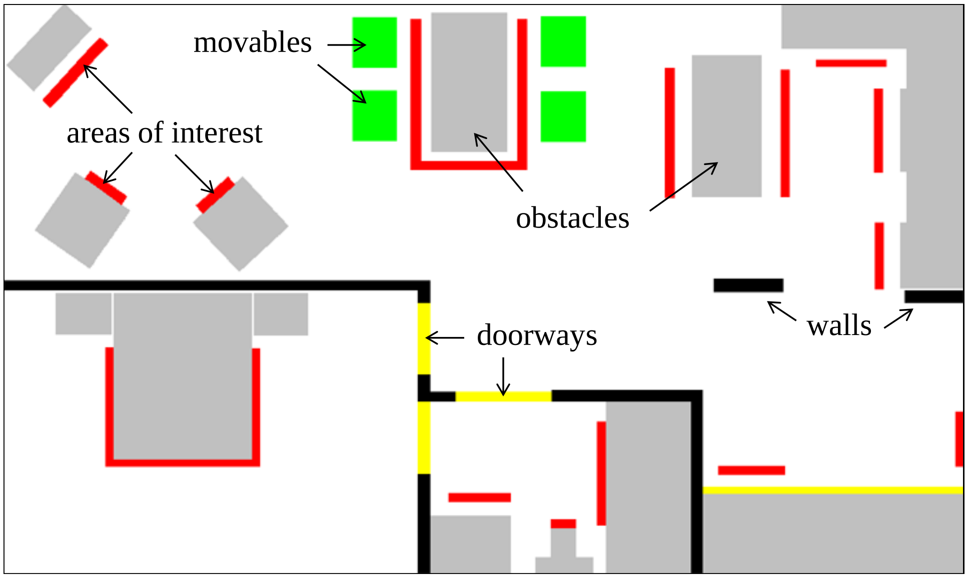


Figure 5.1: The color-coded floorplan.

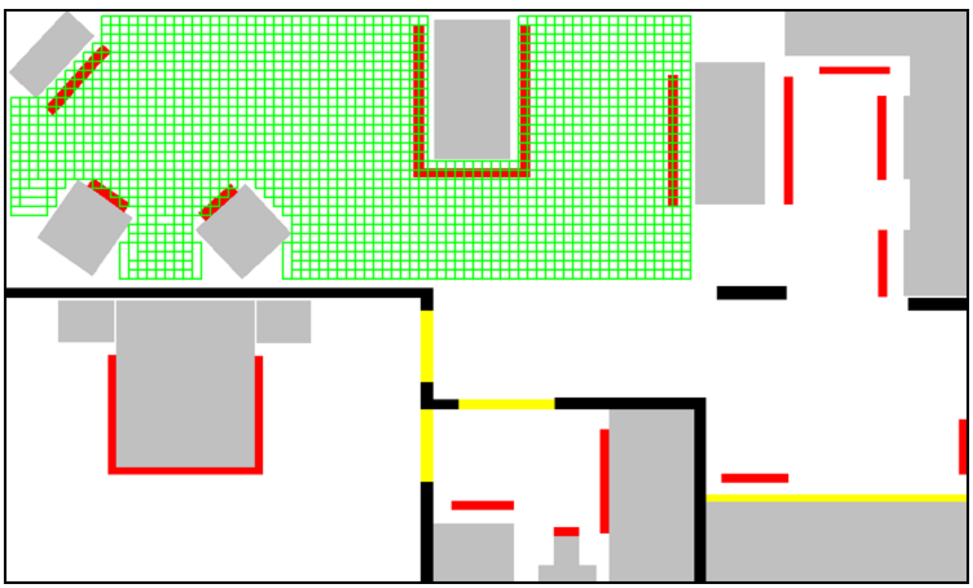
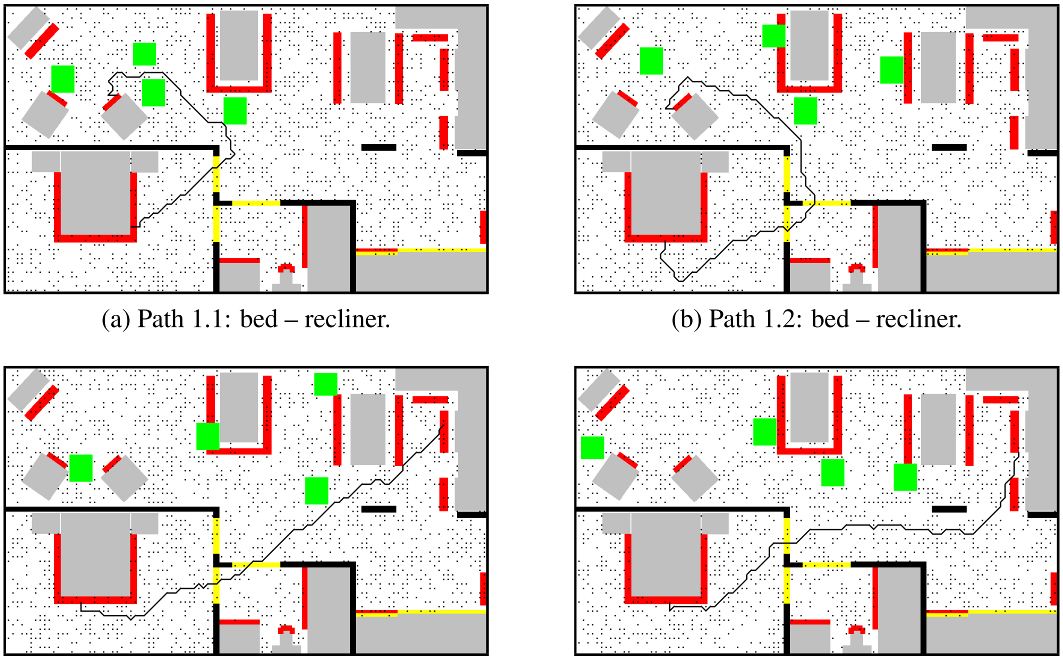


Figure 5.2: The grid of all positions that can be potentially occupied by the movable obstacles.

properties. The only reasonable assumption regarding these objects is that they remain within a certain area of the condo but will occasionally be moved to random locations. To model this behavior of the movables, we define borders of the area that confines their displacement, and then identify all possible locations within that area that satisfy the dimensions of the movables. Figure 5.2 depicts such positions within the user-defined area.



(a) Path 1.1: bed – recliner.

(b) Path 1.2: bed – recliner.

(c) Path 2.1: bed – stove.

(d) Path 2.2: bed – stove.

Figure 5.3: Effects of randomization on the results of pathfinding (black dots are excluded grid points).

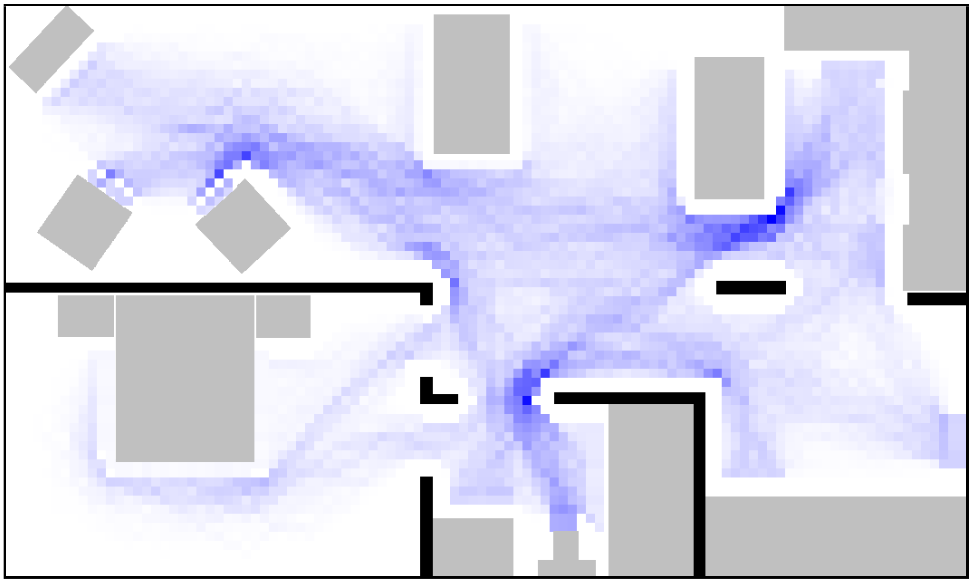
The last type of object represents areas where the occupant is likely to end up while trying to approach a piece of furniture or facility. Note, for instance, the recliners in the living room: they have only one side that can be sat on, therefore we place the corresponding area of interest next to that side only. It is the case with the bed that we cannot predict which side is likely to be preferred by the occupant for getting into the bed, therefore we define its area of interest around the rim of the bed. Other areas of interest (next to the fridge, stove, etc.) are intentionally not adjacent to the respective objects since the person typically reaches those at arm's length. A crucial distinction between the areas of interest and their respective objects is that the former are considered walkable and are used for constructing paths between otherwise impassable objects.

Having defined all the necessary objects, the image serves as input for our custom floorplan parser. The parser renders the matrix of image pixels into a uniform grid (with configurable grid step size) and outputs various data structures corresponding to the groups of objects.

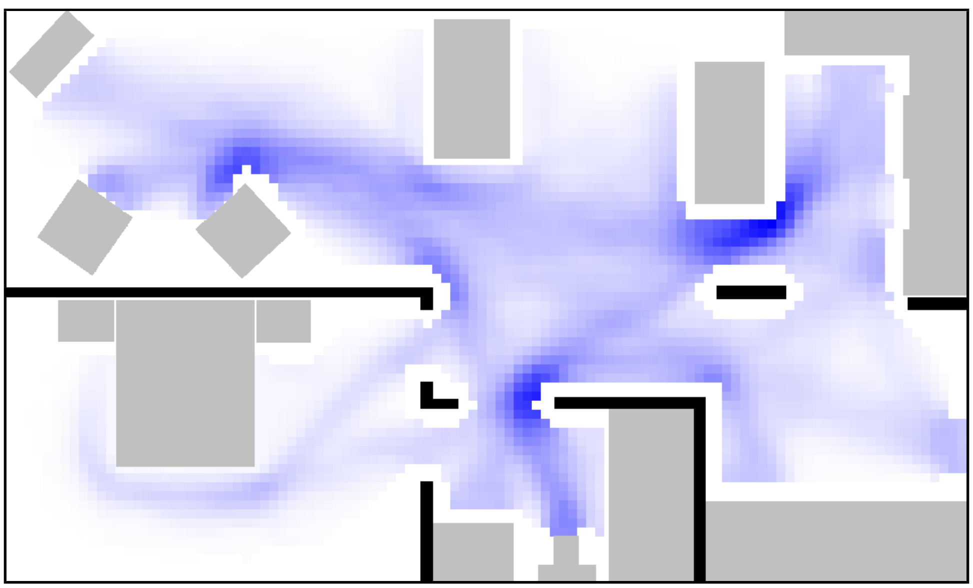
5.3.2 Mobility Modeling

We want to model a variety of realistic paths existing for all pairs of objects of interest. Once the original image is approximated to a square grid, each area of interest turns into a set of grid points. The procedure of constructing a path between a pair of objects entails choosing a point from each set of points corresponding to the objects and inputting two such points as a start and a goal into a pathfinding algorithm (PFA), *i.e.,* a generic implementation of A*. All the grid points except walls and impassable obstacles are considered walkable (assuming the doors can always be opened if needed). Given the PFA's property to always produce the shortest path, we may end up with a number of recurring straight-line paths that do not accurately represent the actual paths a human would choose. To alleviate this undesirable effect we exclude a number of random walkable points during each execution of the PFA (empirically chosen as 30% of the walkable grid due to visually satisfactory variability of generated paths). Figure 5.3 illustrates how different the paths generated for the same pair of objects can be due to three types of randomized input: (1) start and goal points representing objects of interest, (2) positions of the movables, and (3) exclusion of a number of walkable points. This model also takes into account a body diameter of the walking agent, which helps to smooth out otherwise sharp turns the PFA tends to make around obstacles.

To obtain a general picture of mobility patterns, we run the PFA over all pairwise combinations of objects multiple times. The result of this procedure greatly depends on the number of times a particular object participates in path construction; this number has to be indicative of importance of the object in a typical daily routine of the tracked occupant. For example, we may hypothesize that the occupant uses the bed twice a day, visits the kitchen three times a day, and uses the washing machine once a week. Respectively, those objects can be assigned weights at a ratio 14:21:1. Thus, their occurrences during multiple runs of the PFA are controlled to conform to this ratio. To assign weights we have to make certain assumptions about the daily routine of the occupant. Perhaps, one way to obtain this information is from a survey about typical usage of appliances, filled out by the person to be tracked. Our mobility modeling module is designed to properly handle weight



(a) Raw heatmap.



(b) “Smoothed” heatmap.

Figure 5.4: Heatmaps of anticipated mobility patterns before/after “smoothing”.

assignment, although in this thesis we keep the weights equal for simplicity. We intend to investigate the impact of different weights (tailored to real patients) on the placement optimization in the future real-world trials.

The resulting mobility model can be seen as a heatmap of frequencies with which the walking agent visits respective grid points (Figure 5.4a). The number of paths generated for this image is the minimal number that guarantees that every point from a set representing an object has at least one path to every other object (a total of 18,060 paths in our case). This heatmap, however, has visible defects: some grid points become bottlenecks while others never get visited, although located in "hot" areas, due to imperfections in grid approximation. Such artifacts may have a negative impact on generation of sensor placements, and we therefore eliminate them by "smoothing" the heatmap. Figure 5.4b depicts the final heatmap further used for finding placements optimized for coverage of the "hottest" points. Essentially, the "heat" score of a grid point can be treated as its information utility value v_i, therefore, N is the number of walkable points, in correspondence with notation used in the introductory problem formulation in Section 5.2.

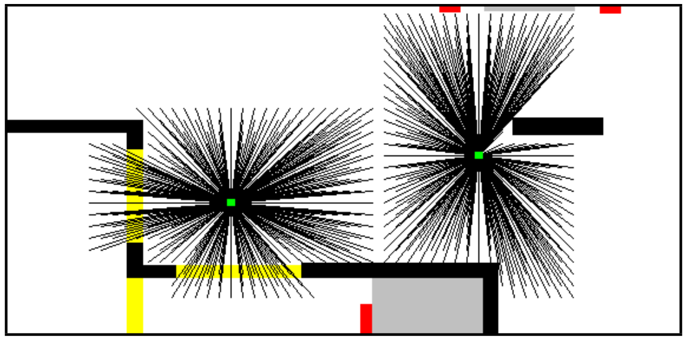
5.4 Sensing Model

As mentioned in Section 2.1, our motion sensors come in a variety of 3D coverage shapes (a cone, a pyramid with a rectangular base), ranges, and detection areas. To reduce the search space of candidate sensor positions/orientations, we restrict the possible sensor orientations to orthogonal with respect to the floor plane, *e.g.,* a cone projects into a circle, and a pyramid projects into a rectangle. We are less interested in the circular projections since a great amount of previous research has been concerned with this particular sensing model [19, 21, 35]. We also lean towards sensors with a smaller coverage area as the deployment space we use for experiments is 10.6 m x 6.3 m and sensors with larger coverage will not be able to provide localization accuracy comparable with the scale of the space. Therefore, we use sensors with the rectangular footprint of 2 m x 1.4 m in cross-section at 2m-height. Technically, these metrics can be expressed in relative or abstract units for the sake of easily generalizable results but we prefer to keep the real-world reference in order to relate our results in this thesis to the previous practical experience.

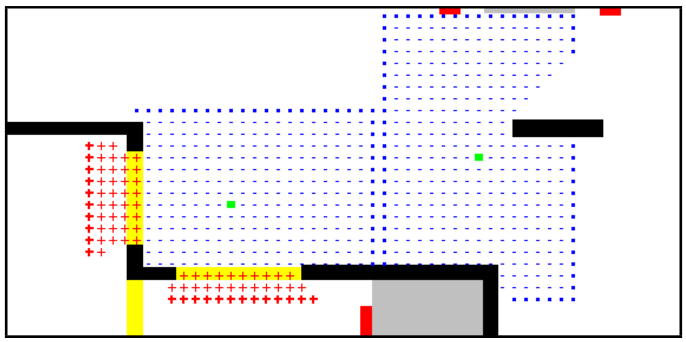
Although we have limited the possible orientation of sensors in the vertical plane to a single option, we consider two orientations in the horizontal plane: the longer side of the projected rectangle can be be either parallel or perpendicular to the longer side of the floorplan. Thus, nominally we have two types of projections of the same sensor type, whereas in fact our framework treats them as two different sensor types. Therefore, a variety of sensors properly handled by our framework is not limited to only the two listed types. We may consider various coverage shapes or a whole variety of (discretized) orientations but do not do so in view of practical concerns (time complexity, ease of mounting associated with the two selected orientation types). Moreover, the obstruction effects of the walls and doors greatly impact sensor projections to the extent that we have to deal with polygons of arbitrary shapes and sizes.

To avoid complicated geometric calculations in continuous space we discretize sensor projections using the grid points defined during the mobility modeling phase. After this step we can easily determine the approximate 2D coverage shape of a sensor resulting from obstruction by applying a ray tracing technique to every grid point within the borders of the default sensor projection. Figure 5.5a visualizes the ray tracing procedure; Figure 5.5b shows the grid approximation used in further calculations instead of the actual polygons (Figure 5.5c). Essentially, our approach can be applied to arbitrary sensor projections, *e.g.,* elliptic or, more generally, any convex/concave shapes whose borders may be defined as a list of connected line segments and arcs).

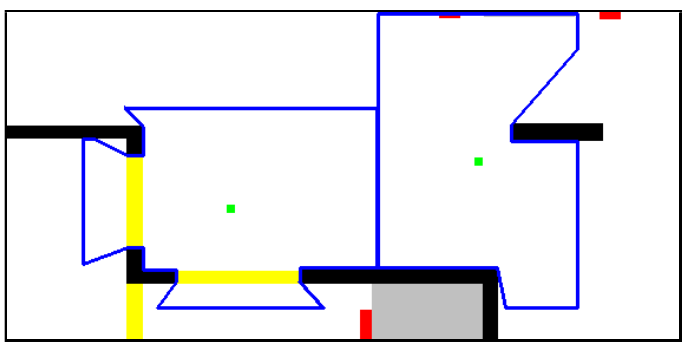
Finally, once the shape of sensor projections is determined, we can define a probabilistic model of target detection for our sensors. In contrast with other studies that use variations of Gaussian probabilistic detection models [33, 35], we assume that the probability of detecting a true positive motion event by a passive infrared sensor is uniform within its footprint and approaches 1 (assuming the sensor is always on during system runtime, failures of sensors or wireless nodes are not considered). This assumption is based on the principles of sensor operation [24] and supported by a great number of empirical studies we have performed. Our framework, however, is not limited to uniform probability distributions; more complex sensing models can also be applied, *e.g.,* non-uniform models of RFID readers. In fact, we further extend the proposed simplistic sensing model by incorporating information about sensor-obstacle geometry into it, in a way similar to that described by Dhillon *et al.* [10].



(a) Ray tracing applied to grid points.



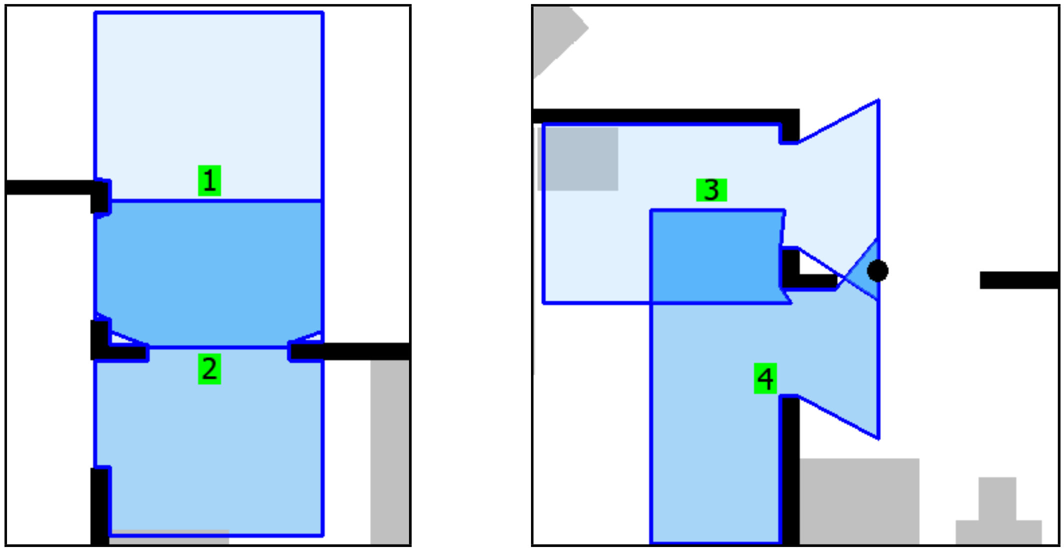
(b) Grid approximation of sensor coverage (crosses indicate points seen through a doorway).



(c) Real shape of sensor coverage projections.

Figure 5.5: Various sensor coverage representations.

To explain the impact of sensor-obstacle geometry on our probabilistic model we refer to Figure 5.5: a footprint of one of the sensors reaches beyond the boundaries of the room through the doorways. Such a footprint holds true as long as the doors are open. Effectively, the probability of doors being open over the time of system operation directly translates into the probability of detection in those "seen-through-doorway" points over the same period. Therefore, we ultimately define the



(a) (b)

Figure 5.6: Specifics of sensor footprints reaching through doorways.

probabilistic sensing model of sensor s_j as follows:

$$p_{s_j \to i} = \begin{cases} 1 & \text{if } i \in \mathbb{C}_j \setminus \mathbb{D}_j \\ p_{\text{od}} & \text{if } i \in \mathbb{D}_j \\ 0 & \text{else} \end{cases} \tag{5.3}$$

where $i = 1, \ldots, N$ are indices of points of interest, $\mathbb{C}_j$ is a set of grid points covered by the sensor after ray tracing has been applied (*e.g.*, Figure 5.5b), and $\mathbb{D}_j \subsetneq \mathbb{C}_j$ is a set of points seen by the sensor through a doorway (the crosses in Figure 5.5b), and p_{od} is the probability that the doors are open. If no statistics about doors usage are available, we may assume that they are open half of the time, *i.e.,* $p_{\text{od}} = 0.5$.

Let us consider two sensors s_1 and s_2 such that the footprint of s_1 fully belongs to a single room and s_2 penetrates into adjacent rooms (Figure 5.6a). Let us also assume that $\sum_{i \in \mathbb{C}_1} v_i = \sum_{j \in \mathbb{C}_2} v_j$, *i.e.,* the total information utility covered by each sensor is equal. Using (5.1) and (5.2), we can calculate information-utility scores for each sensor placed independently as $\sum_{i=1}^{N} (p_{s_j \to i} \cdot v_i)$, assuming no other sensors have been placed on the grid yet. Plugging probabilities (5.3) for each sensor respectively, sensor s_2 will receive a remarkably lower score than s_1 due to $p_{\text{od}} < 1$. In other words, by assigning lower probabilities of detection to seen-through-doorway areas, we effectively penalize placement candidates that yield unreliable doorway coverage.

The penalizing side-effect of such a probability assignment can be exploited for avoiding undesirable artifacts shown in Figure 5.6b. Sensors 3 and 4 belong to the same room and overlap in two disjoint regions (the darkest shade in the image): (i) one lies within the same room and (ii) the other is seen through the doorways by both sensors. If all the doors are open, the sensors will produce identical signals for motion detected in either of the regions. That is, if the moving target crosses the second region (black dot in the image), both sensors get triggered, and those signals may be mistakenly interpreted as if the target is in the first region. This type of error might not be as detrimental for localization accuracy but greatly impacts the quality of contextual information. That is, we cannot distinguish between target presence in the bathroom or bedroom if no other sensory input is available. However, such information is crucial for clinicians observing a patient about to be discharged. To avoid placements with regions of ambiguous room designation, $p_{\text{od}} = 0.1$ is used in our experiments as a means of penalizing sensors covering areas beyond the respective room boundaries.

Chapter 6

Sensor Placement Algorithm

Let $\mathbb{S}$ be a set of sensor candidates $\{s_j\}$ identified by tuples (x_j, y_j, o_j) where x_j and y_j denote a pair of coordinates on the ceiling and o_j denotes the horizontal orientation with respect to the longer side of the floorplan (0° or 90°). In the proceeding calculations we assume that the ceiling height is fixed (2.5m) and therefore omit the z-coordinate. However, variable height (*e.g.,* sloped ceiling) can also be handled under the assumption that the sensor main optical axis is still orthogonal to the floor plane. To reduce the search space of (x, y) pairs, the ceiling coordinate plane is discretized with the same grid step as the floor plane. Thus, sensor coordinates are aligned with coordinates of the points of interest. We will further treat a set of sensor positions as a superset of the set of points of interest since sensors can be placed above the area occupied by obstacles, which is excluded from the points of interest as non-walkable. Hence, $|\mathbb{S}| > 2N$ where the factor 2 is due to two types of sensor orientation. In preparation for the next steps of the algorithm we determine $\mathbb{C}_j$ and $\mathbb{D}_j$ for all $s_j \in \mathbb{S}$.

In the design of our placement algorithm we follow Wang *et al.* [33] whose objective function is defined in (5.2). They propose the following greedy algorithm: at each iteration the algorithm adds a sensor that maximizes information utility gain with respect to already covered points. The incremental information utility gain of placing the m^{th} sensor is therefore expressed as:

$$\Delta(m) = \sum_{i=1}^{N} v_i \cdot \left(\prod_{j=1}^{m-1} (1 - p_{s_j \to i}) \right) \cdot p_{s_m \to i} \tag{6.1}$$

where $j = 1, 2, \ldots, m-1$ are indices of sensors placed thus far. According to this expression, if a set of points has already been covered with probability of detection 1, then information utility gained by placing another sensor over the same set of points will be 0. This definition of information utility gain applied to our sensing model will lead to the majority of points of interest to be covered by non-overlapping sensors. However, merely detecting a target with a single sensor does not typically translate into high localization accuracy. In the indoor localization realm it is often essential for successful localization that the target is sensed by multiple sensors simultaneously, especially if such popular techniques as triangulation or trilateration are used. Similarly to these techniques, we may achieve better localization accuracy at the points of high interest if several sensor footprints partially overlap in that particular area. On the other hand, if we simply follow this intuition, and the range of the heatmap values is large (several orders of magnitude between the lower and upper values), we will end up overlapping too many sensors over the high-priority areas while sacrificing coverage in the rest of the space. Such situations are especially undesirable if the budget of devices is very limited. Therefore, we need to find a tradeoff between redundant coverage of the most travelled areas and sufficient coverage of the less travelled areas.

6.1 Sensor Coverage Models

The first step towards solving our problem is to replace probabilistic sensing models with models of coverage. We introduce a measure of *coverage units* per grid point, which basically indicates how well a particular point is covered. Let $c_i^{(\{s_j\})}$ denote the amount of coverage units allocated to the arbitrary i^{th} grid point by sensor s_j and let $c_i^{(\{s_j\})}$ be equal to $p_{s_j \to i}$ from (5.3). The equality of coverage units and probabilities in this case is merely quantitative. That is, if a point is covered by multiple sensors, the total amount of coverage units allocated to that point is calculated as a sum of coverage units as opposed to multiplying the respective probabilities. Therefore, a cumulative coverage score of the i^{th} point, covered by sensors $s_1, \ldots, s_k$, is

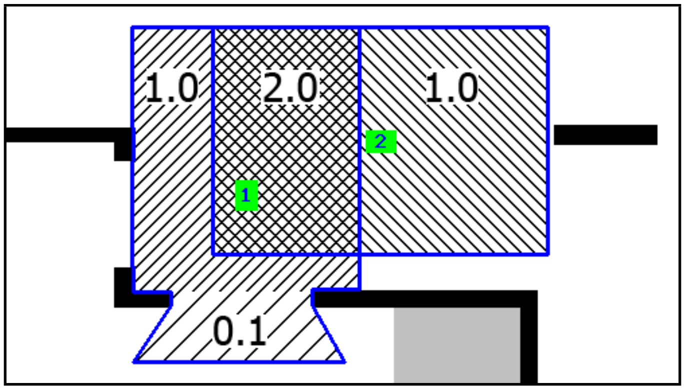


Figure 6.1: Allocation of coverage units.

defined as follows:

$$c_i^{(\{s_1,\ldots,s_k\})} = \sum_{j=1}^{k} c_i^{(\{s_j\})} \tag{6.2}$$

In the most trivial case, when m sensors cover the i^{th} point with $c_i^{(\{s_j\})} = 1$, we obtain $c_i^{(\{s_1,\ldots,s_m\})} = m$. In other words, the coverage score is indicative of the number of sensors overlapping at a point. Figure 6.1 illustrates how coverage units are calculated for sensors s_1 and s_2:

$$\begin{aligned}
&\forall i \in \mathbb{D}_1,\ c_i^{(\{s_1\})} = 0.1\\
&\forall i \in \mathbb{C}_1 \setminus \mathbb{D}_1,\ c_i^{(\{s_1\})} = 1.0\\
&\forall i \in (\mathbb{C}_1 \setminus \mathbb{D}_1) \cap \mathbb{C}_2,\ c_i^{(\{s_1,s_2\})} = 2.0
\end{aligned}$$

Since every sensor is associated with a set of values $c_i^{(\{s_j\})}$ defined for all N points of interest (recall (5.3)), we will further use N-dimensional vectors $\mathbf{s_j} = (c_1^{(\{s_j\})}, \ldots, c_N^{(\{s_j\})})$ to denote sensor coverage models, and vectors $\mathbf{c}^{(\{s_1,\ldots,s_k\})} = (c_1^{(\{s_1,\ldots,s_k\})}, \ldots, c_N^{(\{s_1,\ldots,s_k\})})$ to denote a cumulative coverage model of the entire space after sensors $s_1, \ldots, s_k$ have been placed.

6.2 Redefining Utility Through Coverage

The next step involves mapping the whole range of heatmap values into a small range of *coverage utility* values. Coverage utility is analogous to information utility in the sense that it prioritizes points of interest for greedy selection. On the other hand, we can view coverage utility values as indicators of *preferable* coverage score, *i.e.,* such a cumulative coverage score per grid point that we would like to achieve by sensor placement. That is, if c_i^* is the coverage utility of the i^{th} point, then by placing k sensors we would like to achieve $c_i^{(\{s_1,\ldots,s_k\})} \to c_i^*$. In other words, points of higher importance should be covered by more sensors than less important points. We further hypothesize that a greater number of sensors overlapping at some point will translate into lower error rates when localizing a moving object at that particular point. This hypothesis, however, fails once the overlapping sensors are clustered too closely together, *e.g.,* if the area of their footprints' intersection approaches the area of a single sensor footprint. The proposed here mapping addresses this issue.

There are different ways to define a heatmap-to-coverage mapping, thus, it is yet another configurable application-specific parameter of our framework. We define it as follows:

$$c_i^* = \left\lceil \frac{v_i}{t(c_{\text{max}})} \right\rceil \tag{6.3}$$

where c_{max} is the maximum preferable coverage score per grid point (also configurable), and $t(c_{\text{max}})$ is a threshold value:

$$t(c_{\text{max}}) = \frac{\max_{i \in \{1,\ldots,N\}} v_i}{c_{\text{max}}} \tag{6.4}$$

Essentially, we put all v_i values into $c_{\text{max}} + 1$ buckets with labels $0, 1, \ldots, c_{\text{max}}$. The main purpose of this mapping is to balance the gap between the lowest and highest values of the heatmap while retaining relative gradation. Thus, the ratio of c_i^* scores assigned to the hottest and least hot (but non-zero v_i) points is guaranteed to be c_{max}. This can be interpreted as if the most frequently travelled points are restricted to be covered by at most c_{max} sensors (under the simplifying assumption that they are covered by sensors with $c_i^{(\{s_j\})} = 1$), whereas the least frequently travelled points will still get a chance to be covered by at least one sensor (if they have been traversed by the PFA at least once, meaning they are not in unreachable places, *e.g.,* an empty corner behind the fireplace).

The result of this mapping can be seen as an alternative heatmap with fewer shades of color. Consider Figure 6.2a: the heatmap of coverage utility values is colored in 4 different shades of a base color (due to $c_{\text{max}} = 4$), to represent corresponding values $c_i^* > 0$, and white for all grid cells with $c_i^* = 0$. The value of c_{max} is chosen as such that allows us to segregate several small groups of high-priority points, which emphasize the most travelled areas, and one large group of low-priority points, which simply outlines the borders of the overall walkable area. When choosing c_{max} it is important to obtain a mapping where the clusters of highest-priority points are represented by relatively small sectors of the map. Essentially, as long as these clusters are significantly smaller than a sensor footprint, our placement algorithm will avoid placing sensors, overlapping over the high-priority points, too close to each other. Therefore, the success of the proposed methodology greatly depends on the selected mapping scheme in general, and on the c_{max} value in our particular case.

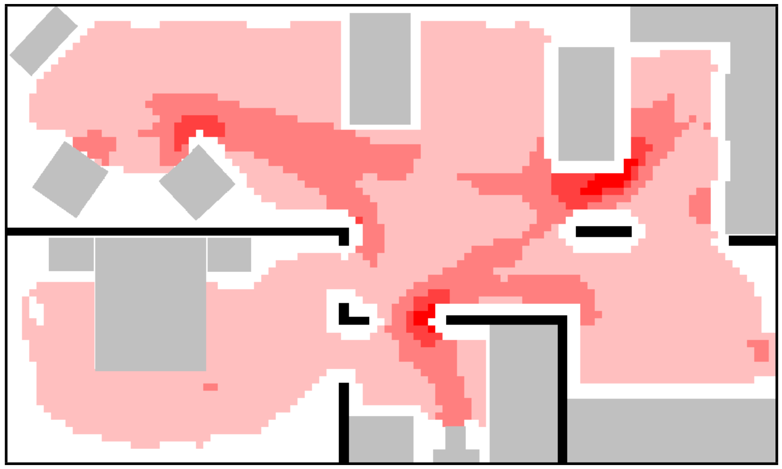
Similarly to vector $\mathbf{c}^{(\{s_1,\dots,s_k\})}$ representing a cumulative coverage model, we define an N-dimensional vector $\mathbf{c}^* = (c_1^*, \dots, c_N^*)$ to denote a coverage utility model of the space.

6.3 Objective Function and Placement Algorithm

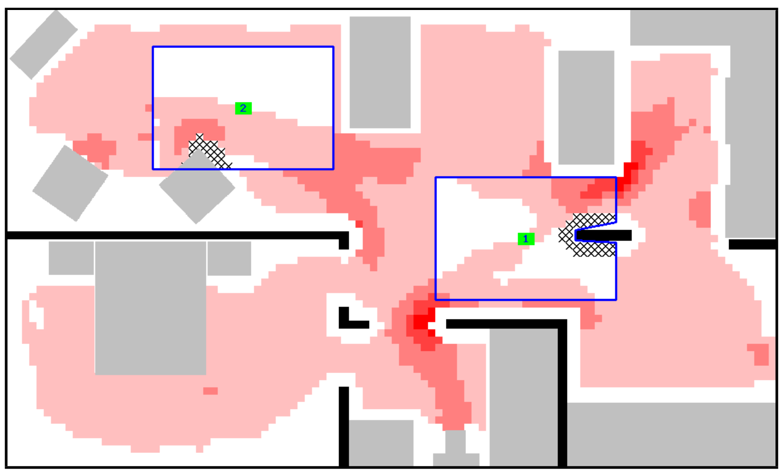
Our goal is to define an objective function in terms of coverage utility and sensor coverage models. To do so, we will express the utility score of an individual sensor as a function over a set of already placed sensors. Let us first consider the case when no sensors have been placed yet; the utility of sensor s_1 is defined as a function of s_1 and the empty set:

$$\delta_{s_1}^{(\varnothing)} = \sum_{i=1}^{N} (c_i^{(\{s_1\})} \cdot c_i^*) \tag{6.5}$$

That is, a sensor that most effectively covers (*i.e.,* with higher $c_i^{(\{s_j\})}$ values) points with higher coverage utility is considered more useful than, for example, a sensor with the same amount of coverage units but covering points with lower c_i^* values. This expression is equivalent to calculating $\Delta(1)$ from (6.1) if v_i and $p_{s_1 \to i}$ are replaced with c_i^* and $c_i^{(\{s_1\})}$ respectively. Expression (6.5) can also be rewritten in vector terms defined in Subsections 6.1 and 6.2 as $\delta_{s_1}^{(\varnothing)} = \mathbf{s_1} \cdot \mathbf{c}^*$.



(a) Heatmap of coverage utility values, $c_{\text{max}} = 4$.



(b) Updated coverage utility values with two sensors placed, cross-hatched areas indicate negative values.

Figure 6.2: Coverage utility heatmap before and after having placed sensors.

Note that once sensor s_1 is placed, another sensor placed in the same location cannot be considered as useful as the already placed one. In other words, if sensor s_2 is placed *after* sensor s_1 and their footprints overlap, then the posterior utility of sensor s_2 is lower than its anterior equivalent, *i.e.*, $\delta_{s_2}^{(\{s_1\})} < \delta_{s_2}^{(\varnothing)}$ given $\mathbb{C}_1 \cap \mathbb{C}_2 \neq \varnothing$. Therefore, we express the utility decrease of each newly added sensor in terms of the mutual overlap with the already placed sensors. One way to quantify the amount of overlap between sensors s_1 and s_2 is a dot product of coverage models of two sensors $\mathbf{s_1} \cdot \mathbf{s_2}$. The posterior utility score of sensor s_2, given that sensor s_1 has been placed on the grid, is defined as follows:

$$\begin{aligned}\delta_{s_2}^{(\{s_1\})} &= \delta_{s_2}^{(\varnothing)} - \mathbf{s_2} \cdot \mathbf{s_1} \\ &= \mathbf{s_2} \cdot \mathbf{c}^* - \mathbf{s_2} \cdot \mathbf{s_1} \\ &= \mathbf{s_2} \cdot (\mathbf{c}^* - \mathbf{s_1}) \end{aligned} \tag{6.6}$$

If we continue adding sensors, then the expression for the posterior utility score of sensor s_3 should account for overlap with the two previously placed sensors:

$$\begin{aligned}\delta_{s_3}^{(\{s_1,s_2\})} &= \delta_{s_3}^{(\varnothing)} - \mathbf{s_3} \cdot \mathbf{s_1} - \mathbf{s_3} \cdot \mathbf{s_2} \\ &= \mathbf{s_3} \cdot \mathbf{c}^* - \mathbf{s_3} \cdot \mathbf{s_1} - \mathbf{s_3} \cdot \mathbf{s_2} \\ &= \mathbf{s_3} \cdot (\mathbf{c}^* - \mathbf{s_1} - \mathbf{s_2}) \\ &= \mathbf{s_3} \cdot (\mathbf{c}^* - \sum_{1 \le j \le 2} \mathbf{s_j}) \end{aligned} \tag{6.7}$$

We finally formalize a closed form expression for the utility score of the m^{th} sensor after $m-1$ sensors have been placed:

$$\delta_{s_m}^{(\{s_1,\ldots,s_{m-1}\})} = \mathbf{s_m} \cdot \left(\mathbf{c}^* - \sum_{1 \le j \le m-1} \mathbf{s_j} \right) \tag{6.8}$$

Note that the sum of sensor vectors $\sum_{1 \le j \le m-1} \mathbf{s_j}$ is another vector whose i^{th} component is $\sum_{1 \le j \le m-1} c_i^{(\{s_j\})}$. Using (6.2) and replacing scalar values with vector notation we may rewrite expression (6.8) as:

$$\delta_{s_m}^{(\{s_1,\ldots,s_{m-1}\})} = \mathbf{s_m} \cdot \left(\mathbf{c}^* - \mathbf{c}^{(\{s_1,\ldots,s_{m-1}\})} \right) \tag{6.9}$$

where a set of sensors $\{s_1, \ldots, s_{m-1}\}$ turns into $\varnothing$ if $m = 1$.

We may interpret (6.9) as a reduction in coverage utility values for each sensor placed after s_1. That is, placing a sensor over a cluster of high-priority points can be seen as reducing the "heat" in that area, which shifts priority to other areas for the next iteration of the algorithm. This *utility update* mechanism balances redundant coverage of high-priority points and satisfactory coverage of the rest of the space. Figure 6.2b illustrates how once a sensor is placed, the colors of the heatmap, representing c_i^* values, change accordingly.

Note the cross-hatched areas: these are the points with zero coverage utility, $c_i^* = 0$. Having placed a sensor over a set of zero-utility points, their utility appears negative for all the subsequent sensors according to (6.9). Zero-utility points are typically points which a walking agent was not able to traverse (a) due to a body-diameter constraint, (b) because they are unreachable, or (c) they lie off the typical paths (*e.g.,* room corners). Thus, when allocating coverage units to zero-utility points, we technically waste those units. Negative utility in this case can be seen as a penalty that lets us avoid further waste of coverage units.

According to (6.9) it is generally possible to obtain a negative utility score for sensor s_m. However, the notion of negative utility is counterintuitive in a domain where additional sensors simply provide extra information. Therefore, we apply positive thresholding to expression (6.9) so that the utility score of any sensor can never be negative, and redefine the utility score function:

$$\delta_{s_m}^{(\{s_1,\ldots,s_{m-1}\})} = \max\left\{0;\ \mathbf{s_m}\cdot\left(\mathbf{c}^* - \mathbf{c}^{(\{s_1,\ldots,s_{m-1}\})}\right)\right\} \tag{6.10}$$

Note that if we reach a state where we have placed so many sensors that the total number of allocated coverage units $\sum_{i=1}^{N}\sum_{1\le j\le m-1} c_i^{(\{s_j\})}$ exceeds the total amount of coverage utility $\sum_{i=1}^{N} c_i^*$ but the budget of sensors has not been exhausted yet, then this indicates that the range of c_i^* values is inadequate for the given number of sensors. Although we resort to positive thresholding in (6.10), it is preferable to avoid this situation by adjusting the range of c_i^*, *i.e.,* by increasing c_{max} parameter (Subsection 6.2).

We finally define an optimal solution to our problem as one that maximizes the total coverage utility obtained by placing k sensors, *i.e.,* a function of a set $\{s_1,\ldots,s_k\}$ expressed as a sum of utility scores of each individual sensor:

$$\Phi(\{s_1,\ldots,s_k\}) = \sum_{i=1}^{k} \delta_{s_i}^{(\mathbb{P}_{i-1})} \tag{6.11}$$

where $\mathbb{P}_{i-1}$ is $\varnothing$ for $i = 1$ or a set of sensors $\{s_1,\ldots,s_{i-1}\}$ for $1 < i \le k$. This is a discrete combinatorial optimization problem with k N-dimensional variables and a number of possible solutions equal to the number of k-combinations of $\mathbb{S}$ (a set of sensor candidates of cardinality $|\mathbb{S}| > 2N$). Finding a globally optimal solution in such a huge combinatorial space is computationally expensive, therefore, we use an approximation algorithm that finds a near-optimal solution in a reasonable time.

We propose a greedy algorithm that adds one sensor at a time and maximizes utility gain at each iteration. The algorithm takes k iterations. At the m^{th} iteration ($m = 1, \ldots, k$) we recalculate utility scores of all sensor candidates from $\mathbb{S}$ using (6.10) and add a sensor with the maximum utility score. If multiple sensor candidates evaluate to the maximum score value, we choose the one whose coverage of the original heatmap of visitation frequencies (before discretization in Section 6.2) is most efficient (*i.e.,* measured as $\sum_{i=1}^{N} \left(v_i \cdot c_i^{\{s_j\}} \right)$ for arbitrary sensor $s_j \in \mathbb{S}$, precalculated once at the beginning of the procedure and never updated). The tie-breaking mechanism is not captured in our objective function as it does not affect its quantitative value. However, this mechanism allows for additional expert knowledge to be encoded into the algorithm. It can be modified depending on the purpose of the deployment. The algorithm terminates after the k^{th} sensor is placed.

6.4 Near-Optimality Guarantee

Following the theoretical framework for sensor placement optimization by Krause *et al.* [18], we show that our greedy algorithm has a strong near-optimality[1] guarantee (proved by Nemhauser *et al.* [23]) due to two important properties exhibited by our objective function: *submodularity* and *monotonicity*.

Intuitively, submodularity can be described as a property of diminishing returns: adding a sensor to a small set of already placed sensors is more beneficial (*i.e.,* generates larger utility gain) than adding a sensor to a large set of sensors. This property is formalized for a set function F defined on subsets of $\mathbb{V}$ as follows:

$$F(\mathbb{A} \cup \{s\}) - F(\mathbb{A}) \geq F(\mathbb{B} \cup \{s\}) - F(\mathbb{B}) \tag{6.12}$$

for all $\mathbb{A} \subseteq \mathbb{B} \subseteq \mathbb{V}$ and $s \in \mathbb{V} \setminus \mathbb{B}$. We may rewrite this property for subsets

[1]Near-optimal solution is typically defined as such that lies within a factor of two of the optimal solution for minimization problems, or such that is $\geq 50\%$ of optimal for maximization problems.

$\mathbb{A} = \{s_1, \ldots, s_{m-1}\}$ and $\mathbb{B} = A \cup \{s_m\}$ and our objective function as follows:

$$\begin{aligned} &\Phi(\{s_1, \ldots, s_{m-1}, s'\}) - \Phi(\{s_1, \ldots, s_{m-1}\}) \\ &\geq \Phi(\{s_1, \ldots, s_{m-1}, s_m, s'\}) - \Phi(\{s_1, \ldots, s_{m-1}, s_m\}) \end{aligned} \tag{6.13}$$

where $s_1, \ldots, s_{m-1}, s_m, s' \in \mathbb{S}$ and $s' \notin \{s_1, \ldots, s_m\}$. To prove that this property holds, we apply several transformations. Plugging the initial definition of the objective function from (6.11) into the left and right sides of (6.13) respectively we obtain:

$$\Phi(\{s_1, \ldots, s_{m-1}, s'\}) - \Phi(\{s_1, \ldots, s_{m-1}\}) \tag{6.14}$$

$$= \left(\sum_{i=1}^{m-1} \delta_{s_i}^{(\mathbb{P}_{i-1})} + \delta_{s'}^{(\{s_1,\ldots,s_{m-1}\})} \right) - \sum_{i=1}^{m-1} \delta_{s_i}^{(\mathbb{P}_{i-1})} \tag{6.15}$$

$$= \delta_{s'}^{(\{s_1,\ldots,s_{m-1}\})} \tag{6.16}$$

$$\Phi(\{s_1, \ldots, s_m, s'\}) - \Phi(\{s_1, \ldots, s_m\}) \tag{6.17}$$

$$= \left(\sum_{i=1}^{m} \delta_{s_i}^{(\mathbb{P}_{i-1})} + \delta_{s'}^{(\{s_1,\ldots,s_m\})} \right) - \sum_{i=1}^{m} \delta_{s_i}^{(\mathbb{P}_{i-1})} \tag{6.18}$$

$$= \delta_{s'}^{(\{s_1,\ldots,s_m\})} \tag{6.19}$$

Plugging results (6.16) and (6.19) into inequality (6.13) is equivalent to the following:

$$\delta_{s'}^{(\{s_1,\ldots,s_m\})} \leq \delta_{s'}^{(\{s_1,\ldots,s_{m-1}\})} \tag{6.20}$$

To prove this inequality, we transform the left term using the utility score function definition (6.10) (omitting positive thresholding for clarity):

$$\delta_{s'}^{(\{s_1,\ldots,s_m\})} = \mathbf{s}' \cdot \left(\mathbf{c}^* - \mathbf{c}^{(\{s_1,\ldots,s_m\})} \right) \tag{6.21}$$

$$= \mathbf{s}' \cdot \left(\mathbf{c}^* - \sum_{1 \leq j \leq m} \mathbf{s_j} \right) \tag{6.22}$$

$$= \mathbf{s}' \cdot \left(\mathbf{c}^* - \sum_{1 \leq j \leq m-1} \mathbf{s_j} - \mathbf{s_m} \right) \tag{6.23}$$

$$= \mathbf{s}' \cdot \left(\mathbf{c}^* - \mathbf{c}^{(\{s_1,\ldots,s_{m-1}\})} \right) - \mathbf{s}' \cdot \mathbf{s_m} \tag{6.24}$$

$$= \delta_{s'}^{(\{s_1,\ldots,s_{m-1}\})} - \mathbf{s}' \cdot \mathbf{s_m} \tag{6.25}$$

Hence, inequality (6.20) holds as long as $\mathbf{s}' \cdot \mathbf{s_m} \geq 0$. Since an arbitrary sensor vector $\mathbf{s_j}$ is specified using $c_i^{(\{s_j\})} \geq 0$, a dot product of any two vectors is also always ≥ 0. Thus, $\delta_{s'}^{(\{s_1,\ldots,s_m\})}$ is always $\leq \delta_{s'}^{(\{s_1,\ldots,s_{m-1}\})}$ (regardless of positive thresholding) and our objective function is submodular for all $\mathbb{A} \subseteq \mathbb{B} \subseteq \mathbb{S}$ such that $|\mathbb{B} \setminus \mathbb{A}| = 1$. This argument can be applied to arbitrary $\mathbb{A} \subseteq \mathbb{B}$ inductively and will hold true if our objective function is monotonic, which we prove next.

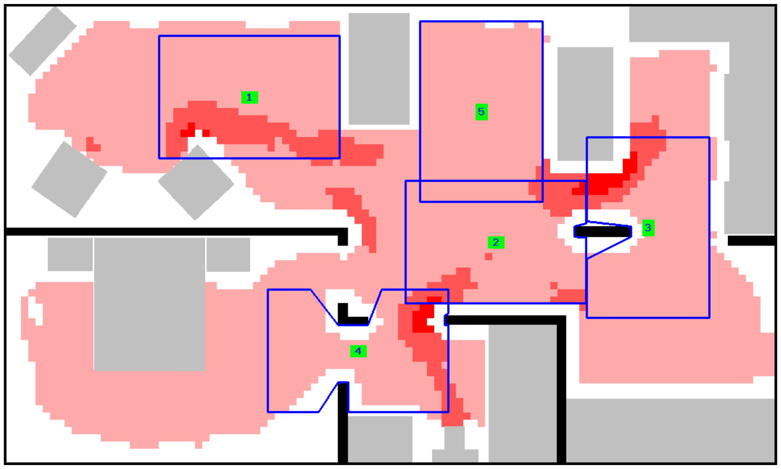
A set function F is considered monotonic if $F(\mathbb{A}) \leq F(\mathbb{B})$ for all $\mathbb{A} \subseteq \mathbb{B} \subseteq \mathbb{V}$. Thus, we want to prove that $\Phi(\{s_1,\ldots,s_{m-1}\}) \leq \Phi(\{s_1,\ldots,s_m\})$. We may rewrite this inequality as $\Phi(\{s_1,\ldots,s_m\}) - \Phi(\{s_1,\ldots,s_{m-1}\}) \geq 0$ and use the result from (6.16) by replacing s' with s_m, *i.e.,* $\Phi(\{s_1,\ldots,s_m\}) - \Phi(\{s_1,\ldots,s_{m-1}\}) = \delta_{s_m}^{(\{s_1,\ldots,s_{m-1}\})} \geq 0$. The last inequality is true due to our definition of a utility score as non-negative in (6.10).

Having proved that our objective function is submodular and monotonic, we claim that the proposed greedy algorithm is guaranteed to find a near-optimal solution. In particular, if $\mathbb{S}_{\text{opt}}$ is a set of k sensors that yield maximum total utility, and $\mathbb{S}_{\text{g}}$ is a set of k sensors found by greedy selection, than $\Phi(\mathbb{S}_{\text{g}}) \geq (1 - 1/e) \cdot \Phi(\mathbb{S}_{\text{opt}}) \approx 0.63 \cdot \Phi(\mathbb{S}_{\text{opt}})$. For more details on the proof of this bound for greedy selection applied to submodular monotonic functions please refer to the fundamental work by Nemhauser *et al.* [23].

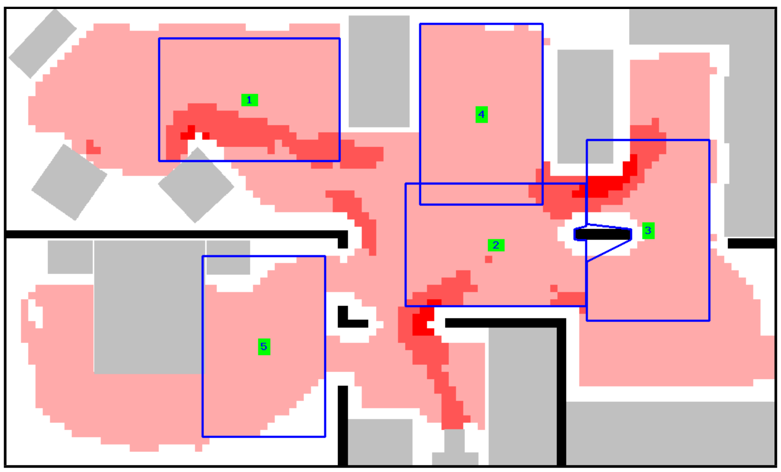
6.5 Illustrative Results

In this section we discuss how the placements generated by our algorithm are affected by the selection of the parameters c_{max} (maximum preferable coverage utility) and p_{od} (probability of doors being open). Figure 6.3 illustrates three placements generated for a budget of 5 sensors and different c_{max} and p_{od} values. The background of each image is colored according to the corresponding c_{max} value: Figure 6.3a and Figure 6.3b have 3 shades of color, and Figure 6.3c has 4.

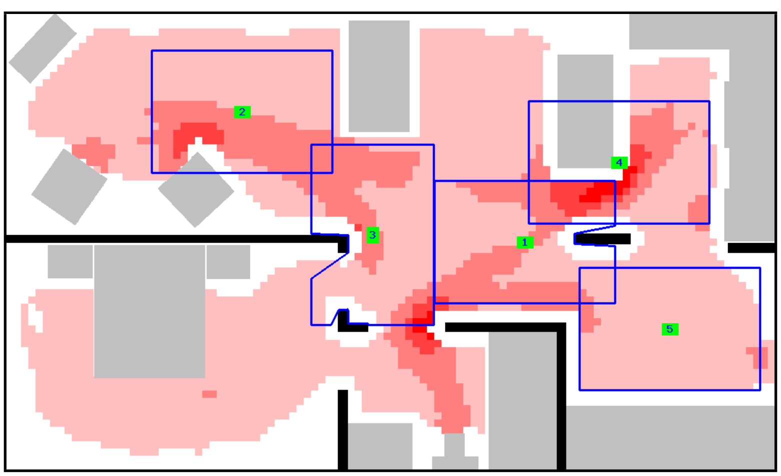
We tested the effect of the value of p_{od} on the resulting placement by generating two placements with $p_{\text{od}} = 1$ and $p_{\text{od}} = 0.1$. The difference between the two placements is apparent: without a strict penalty for crossing the boundaries of the



(a) $c_{\max} = 3$, $p_{\text{od}} = 1$.



(b) $c_{\max} = 3$, $p_{\text{od}} = 0.1$.



(c) $c_{\max} = 4$, $p_{\text{od}} = 0.1$.

Figure 6.3: Examples of placements with different values of $c_{\max}$ and p_{od}.

rooms, about half of the footprint area of the sensor placed in the bathroom in Figure 6.3a reaches into the adjacent rooms. Once the penalty is enabled (Figure 6.3b), the same sensor is “pushed” out of the bathroom into the bedroom. The placement in Figure 6.3c has been generated with the same p_{od} as in Figure 6.3b but with $c_{\max} = 4$. The difference is also remarkable: the sensors are more tightly placed in the areas of intensive color shades. Distinctive in this placement is that two sensors were overlapped very precisely over the area of the highest color intensity, which is one of the algorithm’s main objectives.

Overall, these examples suggest that the algorithm may work as expected, *i.e.,* improve localization accuracy when compared to unoptimized placements with the same number of sensors, if used with well-chosen parameters. One difficulty we encountered is that the selection of $c_{\max}$ proved non-trivial. Intuitively, there should be a correlation between this parameter and the number of sensors given for placement. However, practically we did not observe a reliable pattern and therefore resorted to choosing this parameter based on the simulation results.

Chapter 7

Experimental Evaluation

In this chapter we highlight the difference between the simulation framework used in the first part of the thesis and the simulation framework used for testing optimized sensor placements. Next, we perform an extensive performance evaluation of the optimized *vs.* manual *vs.* randomized placements.

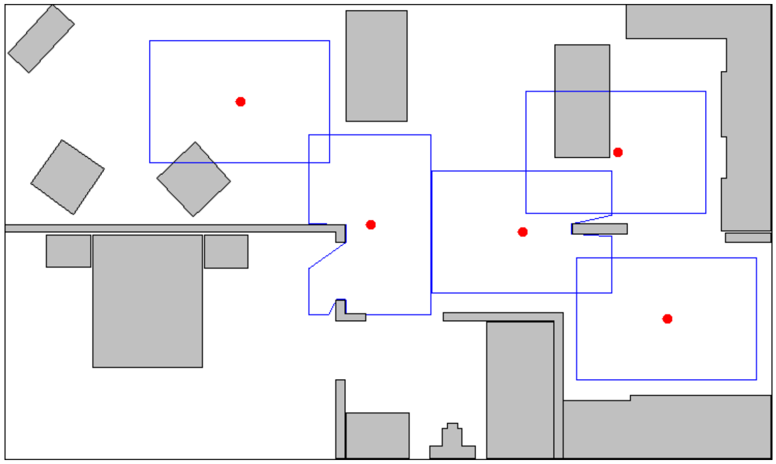
7.1 Modified Simulation Framework

The simulation is performed similarly to how it has been described in Section 3.2, however, with one crucial modification. We eliminate a notion of disjoint rooms represented by polygons as our updated model of the space allows for arbitrary walls (including inner columns).

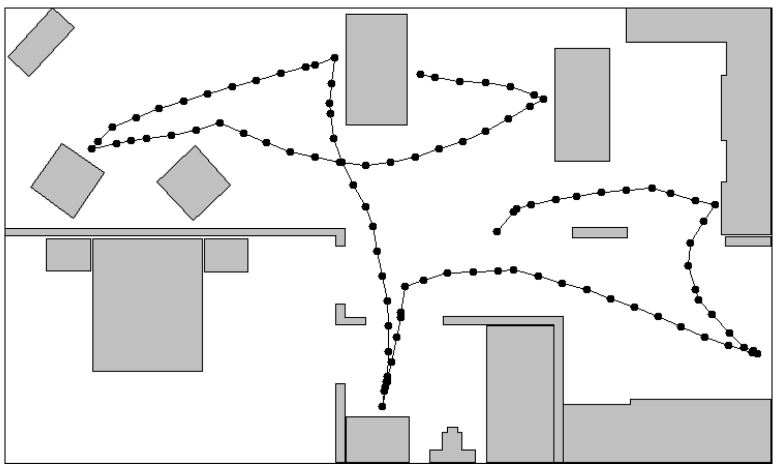
We recreate all critical objects from the floorplan and sensor-coverage map in the 2D continuous geometric space, *i.e.,* objects such as obstacles and sensor footprints are represented as closed chains of vertices forming arbitrary polygons. The resulting model of the space integrated with the sensor-coverage map is shown in Figure 7.1a.

We use the simulator software to generate realistic avatar traces representing the occupant's daily routine. An example of one such routine is shown in Figure 7.1b. Importantly, the avatar trace is also defined in the continuous geometric space. Thus, our placements will be tested on continuous data points as opposed to discrete points used for mobility modeling.

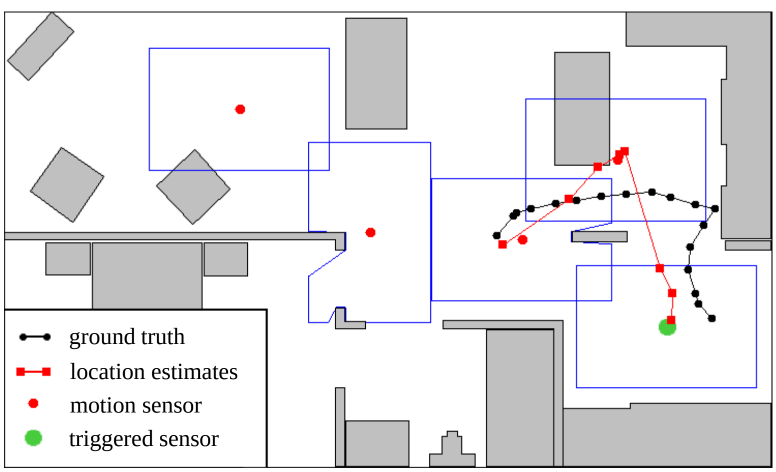
Figure 7.1c also illustrates location predictions made by the localization algo-



(a) Model of the space integrated with sensor-coverage map.



(b) A fragment of an artificial “daily-routine” trace.



(c) Comparison of the ground truth and estimated locations.

Figure 7.1: Modified simulation.

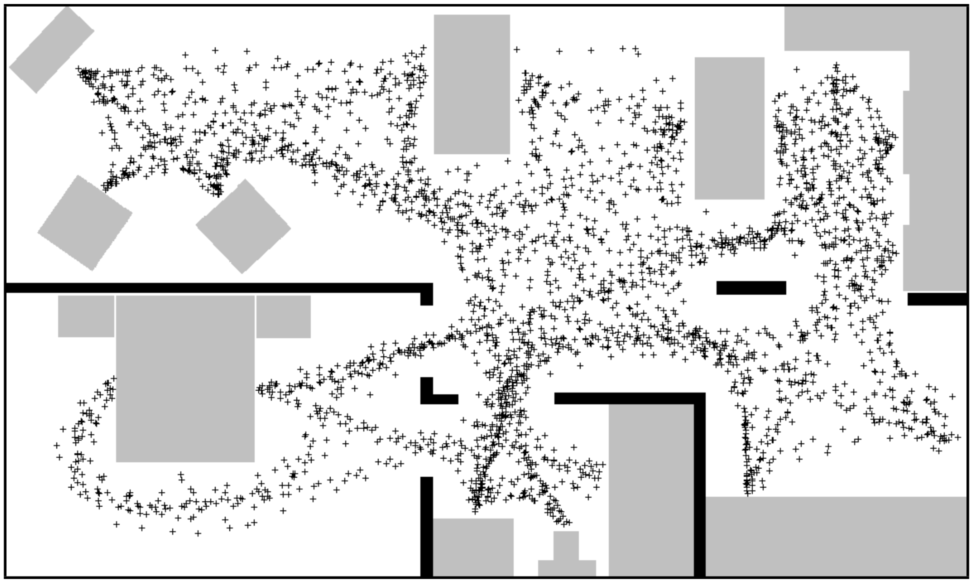


Figure 7.2: Combined footprint of location tuples from10 daily-routine traces.

rithm. At the moment, the algorithm does not exclude the area occupied by obstacles from possible locations of the moving target. However, we are not interested in changing this behavior as we want to show that the localization accuracy can be increased by *merely optimizing the placement* in contrast with attempts to improve the localization algorithm.

7.2 Experimental Setup

We compare three placement strategies: (a) placements generated by our algorithm, *i.e., optimized*, (b) placements crafted manually, and (c) randomized placements. Each strategy is represented by a number of placements varying in cardinality. Based on the dimensions of the deployment space and sensor footprints, we focus on cardinality in a range from 5 to 20 sensors. We have generated 10 artificial traces of various length representing typical daily routines, as described in Section 7.1 and illustrated in Figure 7.1b. The combined footprint of all 10 traces is shown in Figure 7.2. For each experiment we report the mean of the localization error, standard deviation, standard error, and three quartiles of error values. The following subsections describe the placement generation in more detail.

7.2.1 Optimized placements

As mentioned at the end of Chapter 6, the success of optimized placements greatly depends on the selected c_{max} value (maximum preferable coverage utility). Having performed a number of visual tests on the boundary cases, *i.e.,* placements of 5 and 20 nodes, we narrowed down a possible range of c_{max} values to a set of $\{3, 4, 5\}$. This choice is based on the following observations: with $c_{\text{max}} < 3$ the impact of coverage utility assignment on the results of the placement algorithm is hardly noticeable; with $c_{\text{max}} > 5$ we observed oversaturation of sensors in high-priority areas while some low-priority areas did not get covered at all even with the significant budget of 20 sensors.

Next, we generate three placements, *i.e.,* with $c_{\text{max}} = 3, 4, 5$, for each $k = 5, \ldots, 20$ and test them against the artificial traces.[1] Figure 7.3 shows the average localization error for each placement in relation to the c_{max} value. Although in some cases the data points coincide (*e.g.,* . $c_{\text{max}} = 3$ and $c_{\text{max}} = 4$ for 11 and 17-20 sensors), we observe a general trend: lower error rates are achieved with $c_{\text{max}} = 4$ for $k \leq 11$ sensors and with $c_{\text{max}} = 5$ for $11 < k \leq 20$. This trend confirms our intuition that c_{max} grows with k. It is worth noting that $k = 11$ is an important threshold: this is the largest number of sensors that can cover the majority of the space (excluding furniture) *without overlap*, *i.e.,* this number is obtained by dividing the total walkable area (area of the floorplan without non-walkable obstacles) by the area of a sensor footprint. Once we have a budget of more than 11 sensors we cannot avoid overlapping. Hence, it is natural to increase the value of c_{max} for all $k > 11$. In all subsequent experiments we use the optimized placements chosen according to the observed trend, *i.e.,* those generated with $c_{\text{max}} = 4$ for $k \leq 11$ and $c_{\text{max}} = 5$ for $11 < k \leq 20$. Examples of optimized placements for $k = 9$ and $k = 20$ are shown in Figure 7.4.

[1] Note the parameter p_{od} is set to 0.1 for all the conducted experiments.

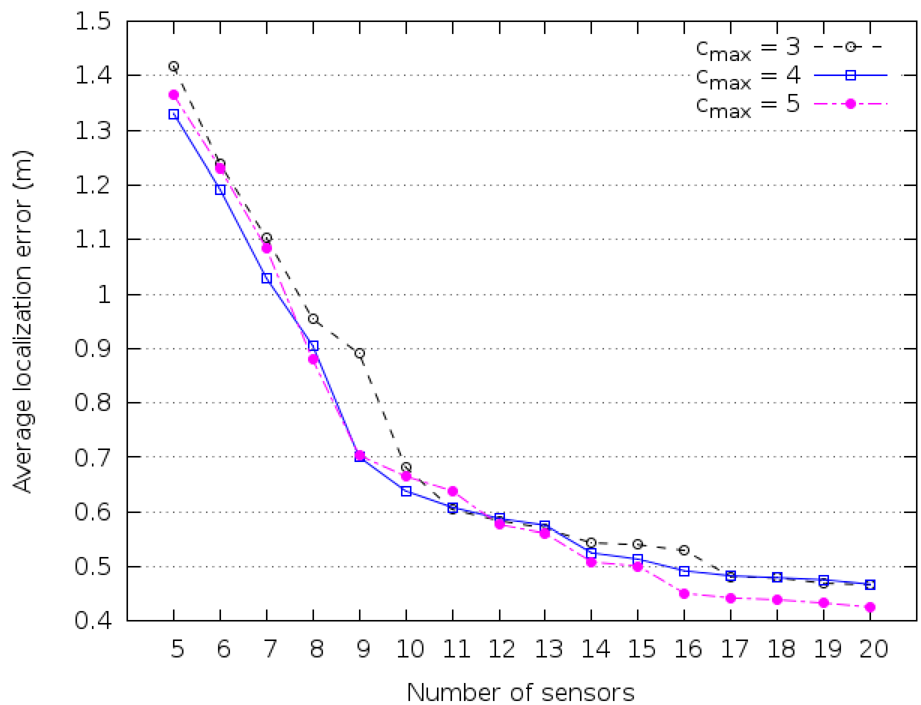
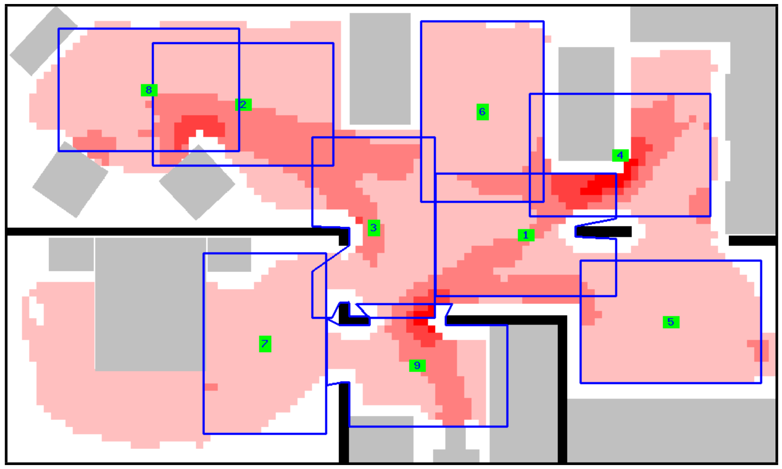


Figure 7.3: Localization error with respect to $c_{\max}$.

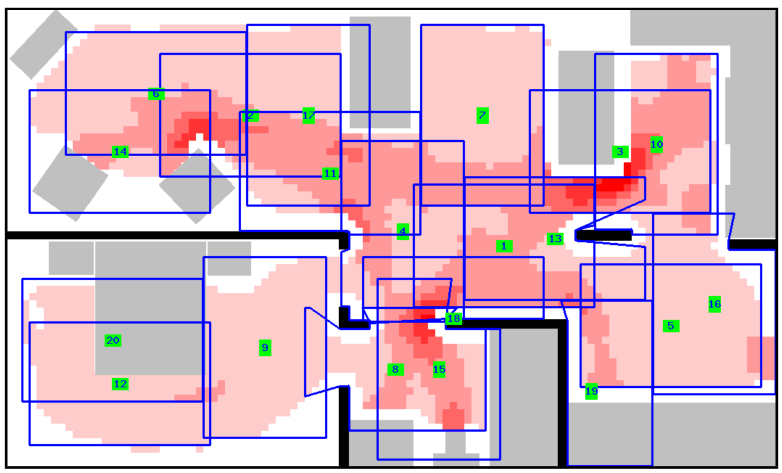
7.2.2 Manual placements

We assume that the expert designing a placement is inclined to maximize total coverage without specific considerations for the area occupied by furniture. Typically, we want to distribute sensors uniformly so that the amount of *uncertainty* about the occupant's location is also uniformly distributed across the space, even if the budget of sensors is not sufficient for full coverage. Given more sensors than necessary for full coverage, we attempt to design placements with a fairly regular "grid" of overlapping regions. Figure 7.5 shows two examples of manually crafted placements: 9 sensors, uniformly covering the space with gaps between sensors, and 20 sensors, regularly overlapping.

We have manually designed 15 placements for each $k = 5, \ldots, 20$ following the aforementioned considerations. Our final goal is to compare performance of the optimized against manual placements on the traces of the typical daily routine. But first we would like to show that our manual placements are designed without a bias towards a particular data set. We generate a number of traces that systematically



(a) 9 sensors, $c_{\max} = 4$.

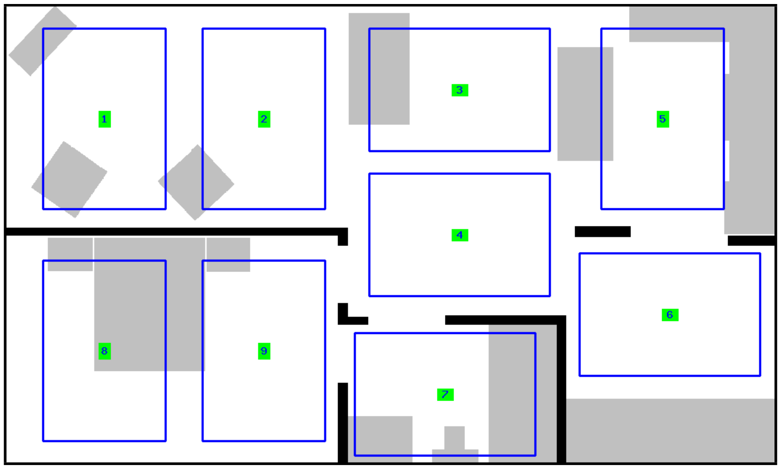


(b) 20 sensors, $c_{\max} = 5$.

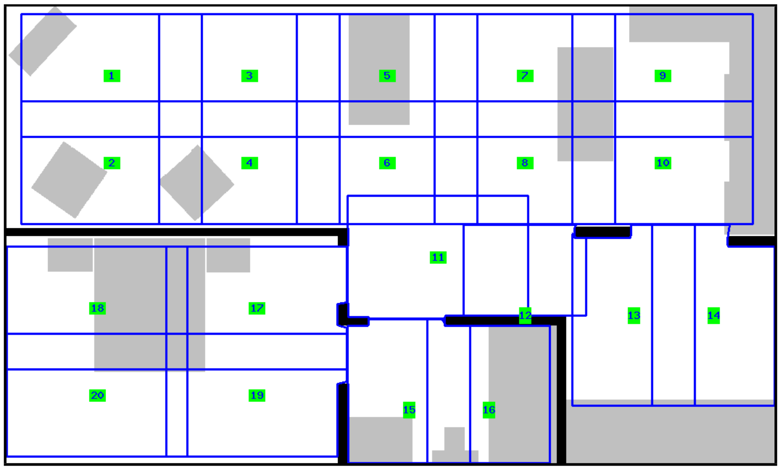
Figure 7.4: Examples of optimized placements.

cover the entire space in a zigzag-like fashion (*e.g.,* Figure 7.6a). The assumed model of the space used for these traces is different from the initial model: it consists of the walls solely (no furniture). Essentially, we attempt to generate traffic of uniform density across the whole space and show that the optimized placements have no advantage over the manual placements on the uniformly-distributed mobility data.

Figure 7.7 compares performance of manual against optimized placements on "zigzag" traces. We can see that the manual placements outperform the optimized ones for all $k = 5, \ldots, 20$. The unpaired t-test shows that the difference between the two types of placements is statistically significant for all k except $k = 7$. This

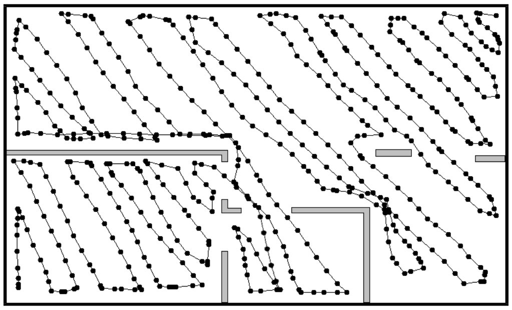


(a) 9 sensors.

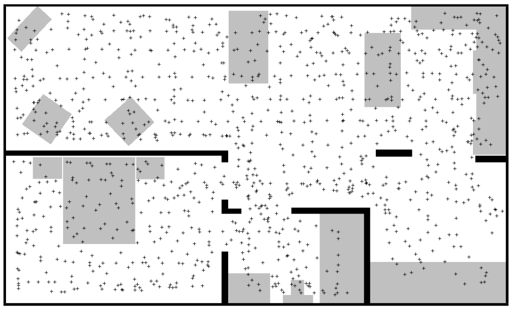


(b) 20 sensors.

Figure 7.5: Examples of manual placements.



(a) Zigzag trace that systematically covers the space and ignores furniture.



(b) Combined footprint of uniform density traces.

Figure 7.6: Uniform density traces.

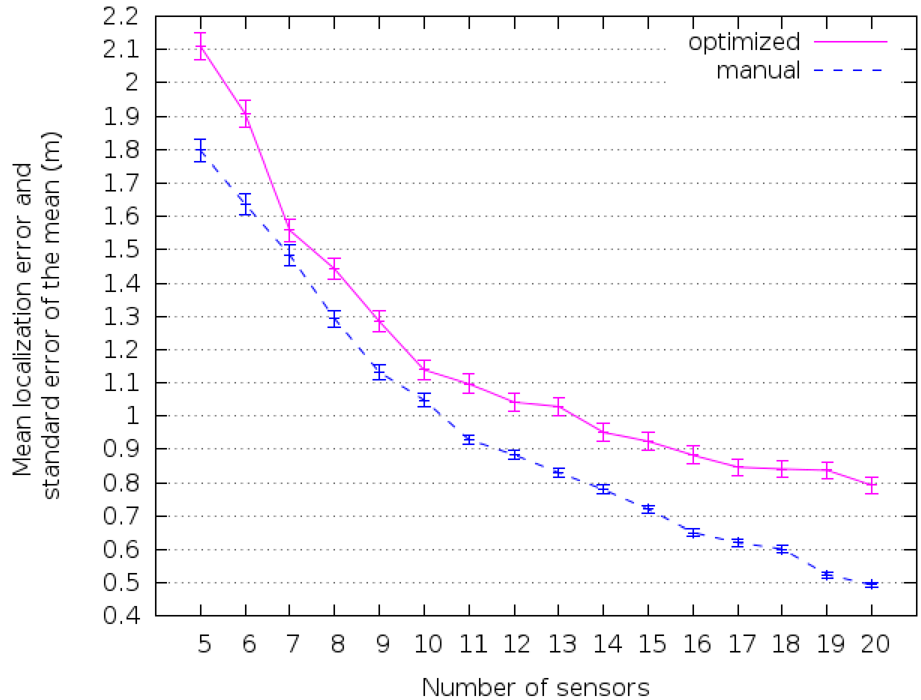


Figure 7.7: Localization error of manual *vs.* optimized placements tested on zigzag traces.

exercise suggests that the manual placements are well-designed and can be considered good candidates for a comparison with the optimized placements when tested on the daily-routine traces.

7.2.3 Randomized placements

We generate 5 randomized placements for each $k = 5, \ldots, 20$ in the following fashion: we split the spatially ordered set $\mathbb{S}$ of sensor candidates into k equally-sized subsets and randomly select one candidate from each subset. This way we avoid occurrences of all k sensors being clustered too close to each other. One example of a randomized placement generated this way is shown in Figure 7.8. In the subsequent sections we report localization error of the k^{th} randomized placement as the average of 5 placements generated for every k, each tested on 10 daily-routine traces.

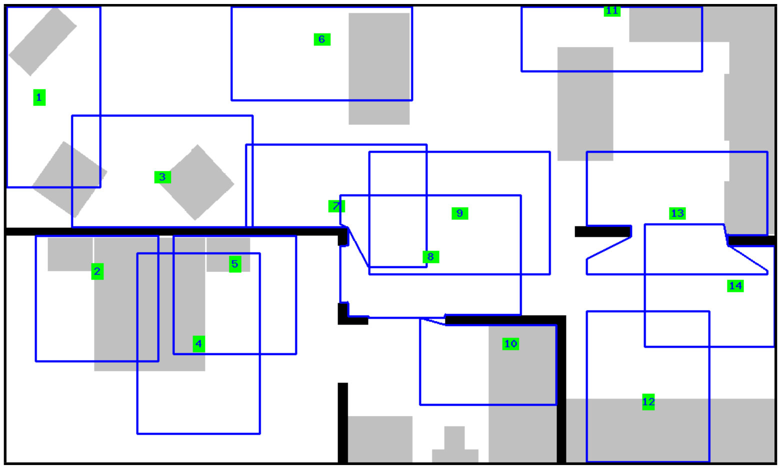


Figure 7.8: Randomized placement with 14 sensors.

7.3 Performance Evaluation

Having selected 15 optimized placements, 15 manual placements and averaging the results of 5 randomized placements for each $k = 5, \ldots, 20$, we evaluate the performance of the three placement strategies with respect to each other. Figure 7.9 depicts the average localization error achieved by each placement in simulation experiments with 10 daily-routine traces (a total of 3943 location tuples). We do not show error bars since the values of the standard error of the mean are so small as to be visually indistinguishable. Please refer to Table 7.1 for full descriptive statistics.

It is clear that both manual and optimized placements significantly outperform randomized placements. An interesting observation based on the performance of the randomized placements is that the localization error generally decreases as the number of sensors increases (although not strictly monotonically) regardless of the placement strategy. That is, manual and optimized placements simply emphasize this trend by monotonically improving localization accuracy with each additional sensor. We also observe that the optimized placements' performance is either as good as that of the manual placements or better in the majority of cases. That is, the unpaired t-test shows that there is no significant difference between the two placement strategies for $k \in \{5, 6, 8, 19\}$, and that the optimized placements are statistically better than the manual ones for all other k except $k = 20$, the only case

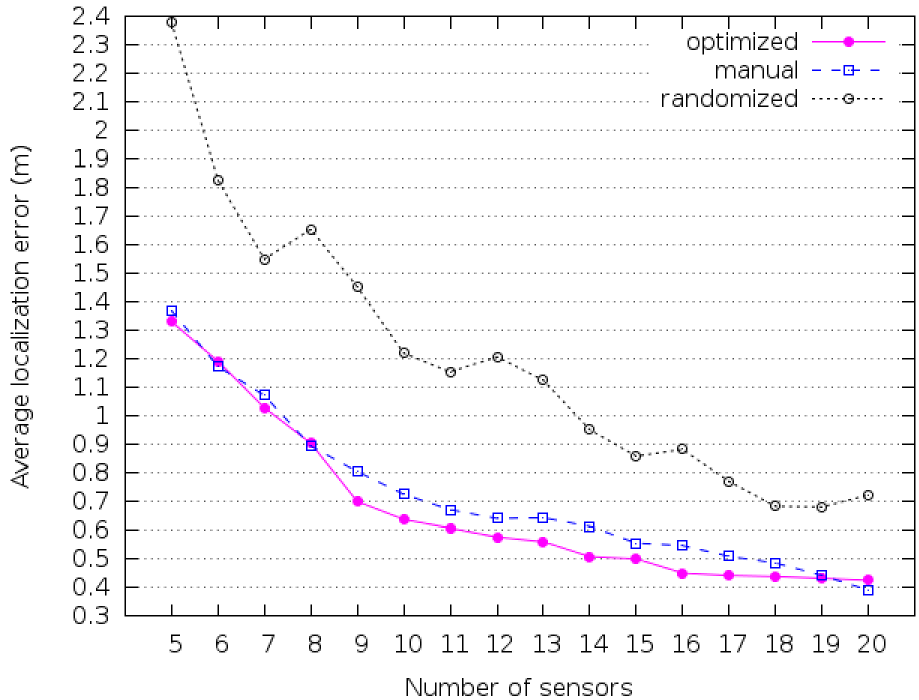


Figure 7.9: Localization error of optimized, manual, and randomized placements tested on daily-routine traces.

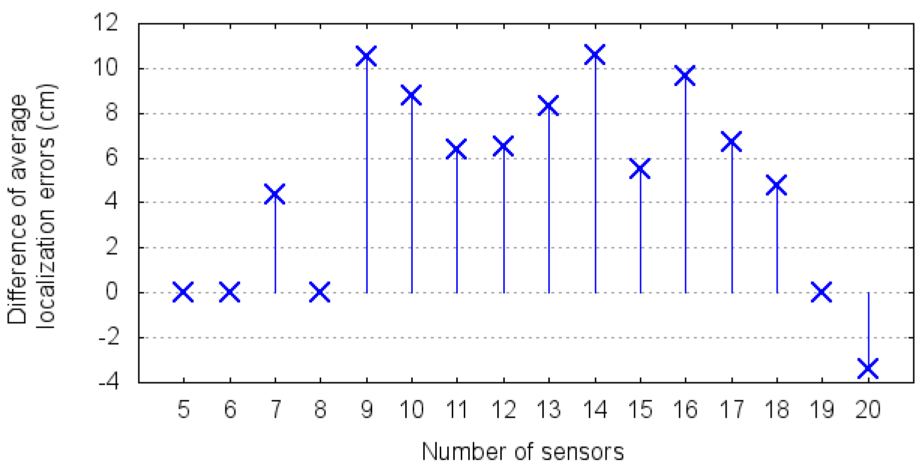


Figure 7.10: Difference in average localization errors of the optimal and manual placements.

when the manual placement statistically outperforms the optimized placement. The difference in average localization error values for all statistically different cases is shown in Figure 7.10.

Note the drastic change in results of the optimized placements between $k =$

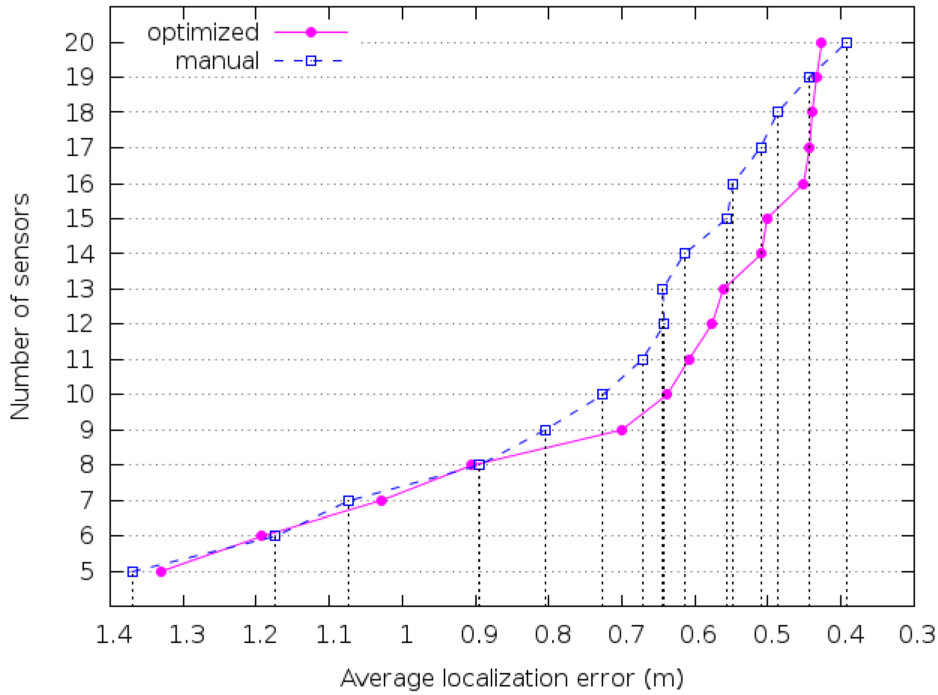


Figure 7.11: Correspondence between the localization error and the number of sensors.

8 and $k = 9$. One notable detail distinguishing the placement with 9 sensors is that it is the first placement (in the order of ascending k) that has a sensor in the washroom (sensor #9 in Figure 7.4a). That is, our algorithm renders sensors placed in the washroom as fairly low utility gain due to two reasons: (i) the walkable area inside the washroom is very limited and (ii) any sensor reaching outside of the washroom is penalized due to $p_{\text{od}} = 0.1$. Therefore, such sensor candidates are disfavored by our algorithm until the 9^{th} iteration. However, when manually designing placements we included at least one washroom sensor in all placements with $k \geq 6$. Moreover, our daily-routine traces contain a significant amount of mobility inside the washroom. Our algorithm, however, can address this problem if we encode expert knowledge about especially important areas into the coverage utility assignment by artificially inflating the coverage utility values of grid cells in those areas.

Aside from optimizing localization accuracy under a cardinality constraint, we are also interested in the reverse problem formulation, *i.e.,* reducing the number of sensors while achieving a desired performance level. Assuming that the localization accuracy achieved with the manual placements is acceptable, we compare the number of sensors required for a certain performance level by both manual and optimized placements. Figure 7.11 is plotted from the same data as Figure 7.9 but with the axes swapped. Consider data points of the curve representing manual placements within a span from 10 to 19 sensors. For each manual placement with $k = 10, \ldots, 19$ we may achieve the same or better level of performance with an optimized placement with $(k - \Delta)$ sensors where Δ is between 1 and 3 sensors for different k. For example, the localization error of the manual placement with 14 sensors is achievable with the optimized placement comprising only 11 sensors. Such a reversed interpretation of our results may eventually reduce the cost of a new deployment. Essentially, we are not as much interested in improving the localization accuracy with the given number of sensors (considering the fact that the performance improvement is relatively marginal, ranging between 4 cm and 11 cm in individual cases) as in exploring the opportunity to reduce the number of sensors without sacrificing performance.

If the localization data are used by online services (*e.g.,* virtual world visualization), then the percentiles of error distribution become more consistent quality indicators than the average error. That is, while visualizing the data in real-time we prefer a placement that generates more accurate predictions for a larger fraction of system runtime, rather than a placement whose overall average performance is better. Therefore, we report three quartiles of error values produced by manual and optimized placements in Figure 7.12. It is clear that all optimized placements with $k < 20$ outperform the respective manual placements in the two lower quartiles. In other words, by using optimized placements instead of manual ones we can produce more accurate location predictions *at least* 50% of the system runtime.

These experimental results suggest that the optimized placements have a number of advantages over the other two approaches. They also confirm that our objective function reliably estimates the quality of a sensor placement with respect to eventual localization accuracy. In summary, our placement algorithm has the potential to replace manual sensor placement, thereby reducing the cost of future

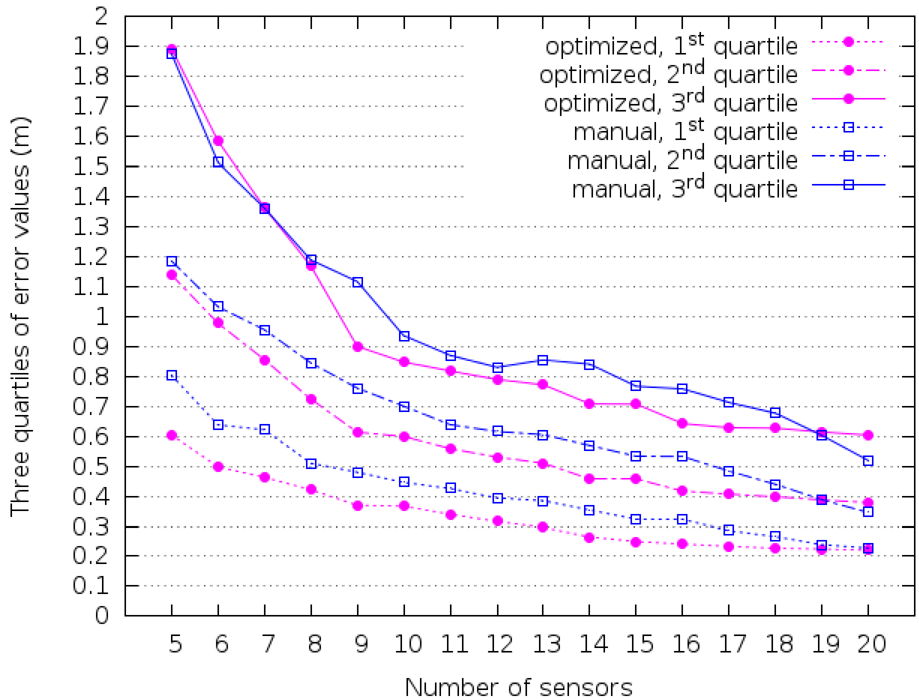


Figure 7.12: Three quartiles of errors produced by optimized and manual placements.

deployments of the Smart-Condo™ system.

Table 7.1: Descriptive statistics of experimental results achieved with optimized, manual, and randomized placements (meters).

k	Type of placement	Mean	SD	SEM	1^{st} qrt.	2^{nd} qrt.	3^{rd} qrt.
	optimized	1.3306	0.9567	0.0152	0.6039	1.1380	1.8919
5	manual	1.3692	0.8060	0.0128	0.8028	1.1845	1.8739
	randomized	2.3772	1.7249	0.0123	0.9974	1.9718	3.3922
	optimized	1.1921	0.9380	0.0149	0.4974	0.9769	1.5837
6	manual	1.1736	0.7185	0.0114	0.6375	1.0317	1.5124
	randomized	1.8266	1.2996	0.0093	0.8109	1.5181	2.5953
	optimized	1.0292	0.7127	0.0113	0.4625	0.8551	1.3629
7	manual	1.0728	0.6591	0.0105	0.6215	0.9552	1.3615
	randomized	1.5508	1.2574	0.0090	0.6398	1.1597	2.1408
	optimized	0.9050	0.6692	0.0107	0.4212	0.7238	1.1678
8	manual	0.8954	0.5120	0.0082	0.5097	0.8447	1.1880
	randomized	1.6549	1.3013	0.0093	0.6812	1.2761	2.2968
	optimized	0.7006	0.4789	0.0076	0.3692	0.6132	0.8994
9	manual	0.8058	0.4312	0.0069	0.4798	0.7605	1.1162
	randomized	1.4538	1.0738	0.0076	0.6345	1.1435	2.0643
	optimized	0.6383	0.3493	0.0056	0.3662	0.5990	0.8472
10	manual	0.7263	0.3697	0.0059	0.4456	0.7002	0.9354
	randomized	1.2211	0.9559	0.0068	0.4929	0.9712	1.7082
	optimized	0.6085	0.3492	0.0056	0.3397	0.5583	0.8176
11	manual	0.6723	0.3461	0.0055	0.4251	0.6384	0.8700
	randomized	1.1554	0.9573	0.0068	0.4813	0.8501	1.5684
	optimized	0.5770	0.3353	0.0053	0.3153	0.5302	0.7892
12	manual	0.6418	0.3346	0.0053	0.3939	0.6156	0.8298
	randomized	1.2077	1.0034	0.0071	0.4596	0.8687	1.7005
	optimized	0.5611	0.3363	0.0054	0.2962	0.5096	0.7713
13	manual	0.6443	0.3426	0.0055	0.3846	0.6049	0.8541
	randomized	1.1263	0.9266	0.0066	0.4635	0.8482	1.5035
	optimized	0.5089	0.3121	0.0050	0.2647	0.4569	0.7076
14	manual	0.6148	0.3418	0.0054	0.3531	0.5687	0.8403
	randomized	0.9523	0.7172	0.0051	0.4217	0.7479	1.3107
	optimized	0.5015	0.3133	0.0050	0.2496	0.4569	0.7067
15	manual	0.5565	0.2911	0.0046	0.3227	0.5329	0.7662
	randomized	0.8601	0.7091	0.0051	0.3478	0.6514	1.1564
	optimized	0.4514	0.2603	0.0041	0.2412	0.4164	0.6419
16	manual	0.5480	0.2779	0.0044	0.3223	0.5319	0.7576
	randomized	0.8834	0.6999	0.0050	0.3879	0.6801	1.1841
	optimized	0.4430	0.2589	0.0041	0.2335	0.4065	0.6282
17	manual	0.5103	0.2751	0.0044	0.2852	0.4831	0.7123
	randomized	0.7711	0.6146	0.0044	0.3312	0.5900	1.0208
	optimized	0.4396	0.2606	0.0042	0.2270	0.3977	0.6265
18	manual	0.4870	0.2750	0.0044	0.2660	0.4392	0.6765
	randomized	0.6838	0.5646	0.0040	0.3088	0.5302	0.8780
	optimized	0.4336	0.2591	0.0041	0.2246	0.3871	0.6139
19	manual	0.4439	0.2634	0.0042	0.2392	0.3878	0.6023
	randomized	0.6814	0.5832	0.0042	0.2940	0.5042	0.8832
	optimized	0.4262	0.2529	0.0040	0.2206	0.3784	0.6035
20	manual	0.3919	0.2207	0.0035	0.2280	0.3450	0.5186
	randomized	0.7238	0.5965	0.0042	0.3160	0.5434	0.9431

Chapter 8

Discussion and Future Work

This chapter is dedicated to a discussion of known limitations of the proposed deployment planning methodology and directions for future work.

Furniture displacement. One source of criticism applicable to our methodology is that it heavily relies on knowledge of furniture placement and assumes that most of the articles of furniture are stationary (except the movables). However, even very heavy and generally stationary items, such as a bed or recliners, can be eventually moved if the occupant of the space desires so. On the other hand, new items can also appear in the space. To address this issue we consider two scenarios.

First, we consider the case when some of the objects that were assumed stationary from the floorplan drawing (*e.g.,* recliners, dining table) in fact turn out mobile, *e.g.,* recliners on a wheeled base. Typically, such details become evident only upon visiting the physical deployment space. Currently, our framework does not stipulate that an object of interest (*i.e.,* an object that participates in pathfinding for mobility modeling) can also be movable. However, inclusion of this type of object can be easily implemented within the existing framework. That is, we can define a group of *movable objects of interest* that combines properties of the two existing groups. For example, a rectangle representing the dining table would be linked with respective areas of interest (red strips in Figure 5.1); the boundaries of the table's possible displacement could be defined in a similar fashion as they are defined for the movables (Figure 5.2); for each execution of the PFA, the linked object of the table with the areas of interest would be placed into random positions within the predefined boundaries.

A more complicated scenario emerges when the occupant of the space decides to remove/add an article of furniture that cannot be considered movable under any circumstances, *e.g.,* a wardrobe. Removing such an item would reveal a potential “blind spot” area (uncovered by the existing sensors), and adding one would affect the existing mobility patterns. Unfortunately, within our framework this scenario is equivalent to a new deployment planning. On the other hand, such modifications are unlikely to occur in a controlled (possibly clinical) environment unless all the interested parties (clinicians, relatives) are properly informed in advance. Therefore, we assume that the sensor placement can be adjusted by the caregivers accordingly.

Blind spots outside the daily routine. Sensor placements optimized with respect to the proposed mobility model are susceptible to the following side-effect: they tend to avoid coverage of presumably “useless” areas (*e.g.,* corners, small patches of unvisited areas between the walls and some furniture). These areas can be seen as *blind spots*, which can eventually be locations of “exceptional events” (*e.g.,* falls).

Our objective function is specifically designed to sacrifice full coverage in favor of improved localization accuracy. However, if the sensor placement has to address the issue of blind spots and yet be able to generate high-quality location predictions, we may just slightly modify the tie-breaking mechanism (explained in the end of Section 6.3) without changing the objective function. That is, when choosing between multiple sensor candidates with equal utility scores, we may give preference to the one whose footprint covers the most *uncovered* area of the floorplan.

Inaccurate door 2D representation. This limitation is related to a transition of 3D sensor coverage models onto the 2D floorplan. In particular, the doorways are represented as gaps between the walls and are assumed to have the same height with the walls. In reality, there is a wall segment between the top of the door frame and the ceiling, which also causes obstruction for a passive infrared sensor. If the height of such wall segments is known, we can more accurately determine the shape of sensor footprints that reach through the doorways. This detail is critical for real-world deployments. Another “door” limitation is related to assignment of p_{od}. In the current implementation this parameter is set globally for all the doors. Ideally, our framework should be able to handle different values of p_{od} for different doors.

Future work. We would like to investigate the fault-tolerance of the optimized placements, *i.e.,* how well they perform under circumstances when one or more sensors stop functioning. Considering the fact that the most frequently travelled areas of the deployment space receive redundant coverage (overlapping sensors) in the optimized placements, we predict that the optimized placements should be more fault-tolerant than placements with uniformly placed sensors.

Another important future direction in which to expand this work is to test the proposed placement methodology in the real-world trials. In short-term test trials, we would like to compare the localization accuracy predicted by our simulation framework to the accuracy achieved with a physically deployed system. In long-term trials with real patients, we want to compare mobility patterns modeled within our framework prior to deployment with the collected mobility data. In particular, we would like to study robustness of our placements with respect to different sources of uncertainty in mobility data, *e.g.,* inaccurate mobility model, unpredicted furniture displacement, localization of more than one person, etc. We also want to investigate whether long-term learning of mobility patterns can be exploited to adjust the placement for better localization performance or to reduce the number of deployed/maintained sensors. Moreover, the number of motion sensors can be even further reduced if we include models of pressure sensors, reed switches, and RFID readers, which we already use in the real-world deployments but have not yet had their impact on deployment planning investigated.

In addition to more placement-related studies, another avenue for future work arises as we consider limitations of the localization algorithm. Currently it disregards furniture placement when predicting locations. We may incorporate the mobility model generated by our framework into the localization algorithm in two different ways. First, we may replace the center-of-mass calculation based on mere geometric considerations with the center-of-mass of the “heat” scores of the grid cells, thus, implicitly excluding areas occupied by furniture from location predictions as those areas are never visited ("heat" score is 0). Second, we may treat our mobility model as a probabilistic model of plausible trajectories and as such use it for calculations of refined location estimates.

Chapter 9

Conclusion

The Smart-Condo^{TM} project aims to provide supportive services to help people with chronic conditions to live independently. Our approach emphasizes a privacy-respecting and non-intrusive monitoring infrastructure comprised of sensors, which collect an occupant's activity data, and actuators, which control home ambience with the goal of improving the occupant's living quality. One of the most valuable sources of data for caregivers is the occupant's mobility profile. Mobility data can be used for early diagnosis of chronic conditions or simply to better deliver intelligent services spatially distributed in the apartment. Thus, an accurate localization method is crucial to our project.

Localization accuracy greatly depends on the sensor placement, which is typically designed manually for each new deployment and whose quality for localization purposes is reliant on designer's expertise. As such, this thesis investigated the problem of deployment planning, which was approached as two subproblems. First, we demonstrated the value of our simulation-based testing methodology, *i.e.,* that it reliably estimates the localization accuracy achievable with a particular sensor placement. Second, we proposed an automated approach for generating sensor placements optimized for localization in the ambient assisted living environments. In particular, we formulated an optimization problem under a cardinality constraint (*i.e.,* a limited budget of sensors). In doing so, we digressed from a typical sensor placement objective – to achieve maximum coverage of the sensing field – and proposed to maximize an application-specific utility-score function, defined on a set of points of interest. We then proposed a methodology for assigning utility scores to the points of interest based on the model of anticipated mobility patterns. That is, the utility of a point is proportional to the likelihood of the occupant visiting that particular point as part of his/her daily routine. We then proposed a greedy placement algorithm that generates near-optimal solutions for our optimization problem, and evaluated the localization performance of the generated placements using the aforementioned simulation framework.

To give empirical support to our methodology, we compared the optimized placements against both manually designed and randomized placements. The optimized placements significantly outperformed all randomized placements and performed better than manual placements in the majority of cases. Overall, our experimental results suggest that our mobility modeling methodology and proposed placement algorithm may eventually eliminate the time-consuming and tedious task of manually selecting the best placement strategy for indoor localization, as well as reduce the cost of new deployments.

Bibliography

[1] MQ Telemetry Transport (MQTT) v3.1 protocol specification. [Online]. Available: http://www.ibm.com/developerworks/webservices/library/ws-mqtt/index.html

[2] Technical annex for indoor localization and tracking, EvAAL competition. [Online]. Available: http://evaal.aaloa.org/current-competition/annex2012-track-localization

[3] E. Akhmetshina, P. Gburzynski, and F. Vizeacoumar, "PicOS: A tiny operating system for extremely small embedded platforms," in *Proceedings of ESA*, vol. 3, 2003, pp. 116–122.

[4] A. N. Bishop, B. Fidan, B. D. Anderson, K. Doğançay, and P. N. Pathirana, "Optimality analysis of sensor-target localization geometries," *Automatica*, vol. 46, no. 3, pp. 479 – 492, 2010.

[5] K. Chakrabarty, S. Iyengar, H. Qi, and E. Cho, "Grid coverage for surveillance and target location in distributed sensor networks," *Computers, IEEE Transactions on*, vol. 51, no. 12, pp. 1448–1453, 2002.

[6] M. Chan, E. Campo, D. Estève, and J. Fourniols, "Smart homes – current features and future perspectives," *Maturitas*, vol. 64, no. 2, pp. 90–97, 2009.

[7] D. Cook, "Health monitoring and assistance to support aging in place," *Journal of Universal Computer Science*, vol. 12, no. 1, pp. 15–29, 2006.

[8] M. Cypriani, P. Canalda, and F. Spies, "OwlPS: A self-calibrated fingerprint-based Wi-Fi positioning system," in *Evaluating AAL Systems Through Competitive Benchmarking. Indoor Localization and Tracking*, ser. Communications in Computer and Information Science, S. Chessa and S. Knauth, Eds. Springer Berlin Heidelberg, 2012, vol. 309, pp. 36–51.

[9] P. David, V. Idasiak, and F. Kratz, "A sensor placement approach for the monitoring of indoor scenes," *Smart Sensing and Context*, pp. 110–125, 2007.

[10] S. Dhillon and K. Chakrabarty, "Sensor placement for effective coverage and surveillance in distributed sensor networks," in *Wireless Communications and Networking*, vol. 3. IEEE, Mar. 2003, pp. 1609–1614.

[11] N. Fet, M. Handte, S. Wagner, and P. J. Marrón, "LOCOSmotion: An acceleration-assisted person tracking system based on wireless LAN," in *Evaluating AAL Systems Through Competitive Benchmarking*, ser. Communications in Computer and Information Science, S. Chessa and S. Knauth, Eds. Springer Berlin Heidelberg, 2013, vol. 362, pp. 17–31.

[12] R. T. Fielding, "Architectural styles and the design of network-based software architectures," Ph.D. dissertation, Univ. of California, Irvine, 2000. [Online]. Available: http://www.ics.uci.edu/ fielding/pubs/dissertation/top.htm

[13] V. Ganev, D. Chodos, I. Nikolaidis, and E. Stroulia, "The Smart Condo: integrating sensor networks and virtual worlds," in *Proceedings of the 2nd Workshop on Software Engineering for Sensor Network Applications*. ACM, 2011, pp. 49–54.

[14] WebSphere Sensor Events. IBM. [Online]. Available: http://www.ibm.com/software/integration/sensor-events

[15] A. Jiménez Ruiz, F. Seco Granja, J. Prieto Honorato, and J. Guevara Rosas, "Accurate pedestrian indoor navigation by tightly coupling foot-mounted IMU and RFID measurements," *Instrumentation and Measurement, IEEE Transactions on*, vol. 61, no. 1, pp. 178–189, 2012.

[16] O. Kaltiokallio, M. Bocca, and N. Patwari, "Follow@grandma: long-term device-free localization for residential monitoring," in *Local Computer Networks Workshops (LCN Workshops), 2012 IEEE 37th Conference on*. IEEE, 2012, pp. 991–998.

[17] S. Knauth, L. Kaufmann, C. Jost, R. Kistler, and A. Klapproth, "The iLoc ultrasound indoor localization system at the EvAAL 2011 competition," in *Evaluating AAL Systems Through Competitive Benchmarking. Indoor Localization and Tracking*, ser. Communications in Computer and Information Science, S. Chessa and S. Knauth, Eds. Springer Berlin Heidelberg, 2012, vol. 309, pp. 52–64.

[18] A. Krause, C. Guestrin, A. Gupta, and J. Kleinberg, "Near-optimal sensor placements: Maximizing information while minimizing communication cost," in *Proceedings of the 5th international conference on Information processing in sensor networks*. ACM, 2006, pp. 2–10.

[19] S. Lee, K. Ha, and K. Lee, "A pyroelectric infrared sensor-based indoor location-aware system for the Smart Home," *Consumer Electronics, IEEE Transactions on*, vol. 52, no. 4, pp. 1311–1317, 2006.

[20] H. Liu, H. Darabi, P. Banerjee, and J. Liu, "Survey of wireless indoor positioning techniques and systems," *Systems, Man, and Cybernetics, Part C: Applications and Reviews, IEEE Transactions on*, vol. 37, no. 6, pp. 1067–1080, 2007.

[21] X. Luo, B. Shen, X. Guo, G. Luo, and G. Wang, "Human tracking using ceiling pyroelectric infrared sensors," in *Control and Automation, 2009. ICCA 2009. IEEE International Conference on*. IEEE, 2009, pp. 1716–1721.

[22] A. V. Medina, J. A. Gómez, J. A. Ribeiro, and E. Dorronzoro, "Indoor position system based on a Zigbee network," in *Evaluating AAL Systems Through Competitive Benchmarking*, ser. Communications in Computer and Information Science, S. Chessa and S. Knauth, Eds. Springer Berlin Heidelberg, 2013, vol. 362, pp. 6–16.

[23] G. L. Nemhauser, L. A. Wolsey, and M. L. Fisher, "An analysis of approximations for maximizing submodular set functions," *Mathematical Programming*, vol. 14, no. 1, pp. 265–294, 1978.

[24] MP motion sensor NaPIOn, datasheet. Panasonic Electric Works. [Online]. Available: http://pewa.panasonic.com/assets/pcsd/catalog/napion-catalog.pdf

[25] A. Roy, S. Das, and K. Basu, "A predictive framework for location-aware resource management in smart homes," *Mobile Computing, IEEE Transactions on*, vol. 6, no. 11, pp. 1270–1283, 2007.

[26] C. Scanaill, S. Carew, P. Barralon, N. Noury, D. Lyons, and G. Lyons, "A review of approaches to mobility telemonitoring of the elderly in their living environment," *Annals of Biomedical Engineering*, vol. 34, no. 4, pp. 547–563, 2006.

[27] B. Song, H. Choi, and H. Lee, "Surveillance tracking system using passive infrared motion sensors in wireless sensor network," in *Information Networking, 2008. ICOIN 2008. International Conference on*. IEEE, 2008, pp. 1–5.

[28] E. Stroulia, D. Chodos, N. Boers, J. Huang, P. Gburzynski, and I. Nikolaidis, "Software engineering for health education and care delivery systems: The Smart Condo project," in *Software Engineering in Health Care, 2009. SEHC'09. ICSE Workshop on*. IEEE, 2009, pp. 20–28.

[29] S. Toumpis and G. A. Gupta, "Optimal placement of nodes in large sensor networks under a general physical layer model," in *Proc. IEEE Communications Society Conference on Sensor and Ad Hoc Communications*, vol. 4, 2005.

[30] I. Vlasenko, M. Vosoughpour Yazdchi, V. Ganev, I. Nikolaidis, and E. Stroulia, "The Smart-Condo^{TM}infrastructure and experience," in *Evaluating AAL Systems Through Competitive Benchmarking*, ser. Communications in Computer and Information Science, S. Chessa and S. Knauth, Eds. Springer Berlin Heidelberg, 2013, vol. 362, pp. 63–82.

[31] M. Vosoughpour Yazdchi, "Indoor localization with passive sensors," Master's thesis, University of Alberta, Edmonton, 2013. [Online]. Available: http://hdl.handle.net/10402/era.30236

[32] J. Wang, R. Ghosh, and S. K. Das, "A survey on sensor localization," *Journal of Control Theory and Applications*, vol. 8, no. 1, pp. 2–11, 2010.

[33] Q. Wang, K. Xu, G. Takahara, and H. Hassanein, "WSN04-1: Deployment for information oriented sensing coverage in wireless sensor networks," in *Global Telecommunications Conference, 2006. IEEE*. IEEE, 2006, pp. 1–5.

[34] K. Woo, V. Ganev, E. Stroulia, I. Nikolaidis, L. Liu, and R. Lederer, "Sensors as an evaluative tool for independent living," in *Advances in Human Aspects of Healthcare*, V. G. Duffy, Ed. CRC Press, 2012, pp. 612–621.

[35] Q. Wu, N. S. Rao, X. Du, S. S. Iyengar, and V. K. Vaishnavi, "On efficient deployment of sensors on planar grid," *Computer Communications*, vol. 30, no. 14–15, pp. 2721–2734, 2007.

[36] T. Yi and H. Li, "Methodology developments in sensor placement for health monitoring of civil infrastructures," *International Journal of Distributed Sensor Networks*, vol. 2012, 2012.

[37] M. Younis and K. Akkaya, "Strategies and techniques for node placement in wireless sensor networks: A survey," *Ad Hoc Networks*, vol. 6, no. 4, pp. 621–655, 2008.