The Smart-Condo: Optimizing Sensor Placement for Indoor Localization

Iuliia Vlasenko, Ioanis Nikolaidis, *Member, IEEE,* and Eleni Stroulia

***Abstract*—The Smart-Condo is a hardware/software platform that aims to support and assist an individual in performing a variety of everyday tasks within his/her living space. The key to achieving this goal is being able to recognize the individual's general activities in real-time, without impeding these activities or compromising privacy. Since location and movement constitute meaningful evidence for many everyday tasks (e.g., presence in the bathroom correlates with personal hygiene activities), we are motivated to develop an efficient, accurate, and noninvasive occupant-localization method. To this end, we propose a methodology for planning the deployment of an array of privacy-respecting binary motion sensors. In particular, given the geometric constraints of the deployment space, we generate a model of indoor mobility patterns typical for a single occupant. We then use this model as the basis for a specific optimization problem: maximizing a measure of how well the frequently-visited areas of the living space are covered by a number of sensors, subject to a cardinality constraint on this number. We argue this optimization objective is a good surrogate for maximizing localization accuracy, and prove that it bears exploitable properties that make it receptive to a simple optimization routine. As a result, we obtain sensor configurations with localization accuracy superior to that achievable with the same number of sensors placed manually or randomly in the same environment.**

***Index Terms*—Indoor localization, optimization, sensor placement, smart homes, wireless sensor networks (WSNs).**

I. 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 [1]. Sensor-based systems are a common means of nonintrusively monitoring a person's activity and providing this person, and his/her formal and informal caregivers, with useful information for making decisions regarding his/her care [2]. In this paper on the Smart-Condo project [3], we have been developing an integrated hardware-software platform for addressing this broad research problem.

The Smart-Condo was conceived as a multipurpose platform for a variety of services such as location- and activity-recognition, alert generation, home automation, etc. Many of these services rely on accurate occupant localization and movement analysis. The importance of understanding the occupant's mobility cannot be underestimated as, on a long-term basis, mobility data can be mined for patterns useful for the diagnosis and assessment of progressive chronic conditions [4]. As a short-term benefit, a system with an accurate location-recognition method can substantially improve the occupant's living experience as it can anticipate and provide various intelligent services (by controlling home actuators) based on the occupant's location and movement trajectory.

The effectiveness of a localization method depends on multiple factors, including the underlying tracking technology, the localization algorithm, and the geometric and structural properties of the deployment space. Our platform employs passive infrared (PIR) motion sensors due to the privacy-respecting nature of their operation, i.e., the only information captured is whether motion occurred within a sensor footprint. Such a technology is more likely to be accepted by users sensitive to video-surveillance techniques, but this benefit comes at the cost of lower localization accuracy.

One way to improve accuracy is to use a larger number of sensors along with a data fusion mechanism. Recognizing that a major factor hindering the adoption of smart-home technologies is cost, we have focused on investigating the tradeoff between the number of sensors (and the consequent cost of the system) and the localization accuracy achievable with a particular sensor placement. In addition to costs associated with equipment and manual labor required for installation and maintenance, each new deployment implies a rather time- and resource-consuming, and thus costly, deployment-planning phase. That is, an expert has to analyze the requirements and design a sensor placement that satisfies the geometric specifications of the new space and guarantees a desirable performance level. Our past experience proves this task challenging since it may require trial runs on the fully deployed system until an acceptable sensor configuration is found.

To enable informed decision making on the part of potential adopters, we have developed a systematic process for simulating and evaluating the performance of the system under particular deployment conditions [5]. Through this process, we can estimate the localization accuracy of candidate sensor placements in the predeployment phase thus virtually eliminating costly trial runs with human participants.

To date, the Smart-Condo platform has been deployed in three different spaces [6]–[8]. The first two deployments

Manuscript received March 7, 2014; revised May 25, 2014; accepted July 20, 2014. Date of publication October 2, 2014; date of current version February 12, 2015. This work was supported by IBM, the National Science and Engineering Research Council, Alberta Innovates Technology Futures, and OlsoNet. This paper was recommended by Associate Editor S. Das.

The authors are with the Department of Computing Science, University of Alberta, Edmonton, AB, T6G 2E8, Canada (e-mail: vlasenko@ualberta.ca; nikolaidis@ualberta.ca; stroulia@ualberta.ca).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TSMC.2014.2356437

served more as validations of the robustness of the platform and its potential usefulness for care providers. Our most recent deployment at the indoor-localization competition [8] was the first time we were able to showcase our platform's more mature occupant-localization feature. While preparing for the competition, we ran extensive simulations to help us decide on a deployment for the competition. Despite its usefulness, this simulation process was not systematic in designing candidate placements; rather it relied on experts, i.e., our team, to suggest intuitively good placements. In this paper, we advance our methodology by introducing a semi-automated approach for generating sensor placements optimized for localization.

Our approach relies on the basic observation that "the relative sensor-target geometry can significantly affect the potential performance of any particular localization algorithm" [9]. We hypothesize that the geometry of sensor placement can be optimized with respect to the measurements being taken, i.e., localization of a single occupant of the indoor environment, thereby improving the localization accuracy of our system without modifying the existing localization algorithm.

To achieve this goal we propose the following methodology. Given an architectural drawing of the deployment space, we extract its geometric properties and the locations of various objects of interest. Using this information, we generate a model of anticipated mobility patterns. Next, we provide specifications of motion sensors that are in turn translated into deployment-specific sensor models. Finally, we obtain a sensor placement as a solution to an optimization problem formulated using the generated mobility and sensor models.

Such an enhanced methodology may become the first step toward self-configuration of our system for deployments in new spaces. The crucial implications of this paper are the following.

1) Our method eliminates the need for either expert knowledge or intuitive guesses which are typically required for manual sensor placement.
2) Our method is able to identify how a reduced number of sensors may be best placed to achieve a desired performance level.

Therefore, this paper makes a promising case for the reduction of the overall cost of a new deployment and, consequently, toward greater proliferation of smart-home technology in general.

In the rest of this paper, we review recent research in the broad field of optimal sensor placement for wireless sensor networks (WSNs) and relate various optimization problem formulations to our objectives in Section II. We propose a mobility-modeling methodology in Section III and describe application-specific sensor models in Section IV. We formulate a sensor placement optimization problem, comment on its complexity, and justify our selection of an approximation routine in Section V. We then discuss the effects of parameter selection in Section VI. Finally, we present our simulation platform (Section VII), report on our experimental evaluation (Section VIII), and conclude with a summary of results achieved in this paper and plans for future work (Section IX).

II. Related Work

An extensive review of various strategies for sensor placement in WSNs with respect to their application domains and problem formulations has been conducted by Younis and Akkaya [10]. The authors claim that optimal node placement is proved NP-hard for most proposed formulations of the sensor-deployment problem. Most studies, therefore, suggest heuristics for finding sub-optimal solutions. Another survey [11] 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: 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. We review the first survey in the next paragraphs.

Younis and Akkaya [10] identify three criteria by which to categorize sensor deployment strategies: 1) the methodology for initial deployment; 2) the optimization objective; and 3) the roles of nodes in the network. 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., inaccessible natural habitats. In this case, the objective becomes to optimize node density or redundancy for fault-tolerance. Toumpis and Gupta [12] present a density-oriented study. The authors assume massively dense networks and suggest optimizing node density with respect to macroscopic parameters such as information density and traffic flow. In a smart-home type of deployment, we can only afford a reasonably small number of sensors, and hence opt for the controlled deployment option. We do not address fault-tolerance achievable by means of node redundancy as we impose a strong constraint on the number of nodes for cost and aesthetics purposes. However, we will investigate this direction in the future.

The most commonly considered optimization objectives are area coverage, network connectivity and/or longevity, and data fidelity [10]. A typical coverage problem is cost minimization under coverage constraints, where the surveillance region is approximated by a finite set of grid points; this problem can be solved using integer linear programming [13].

Techniques dedicated to network operation requirements (i.e., connectivity and longevity) are relevant primarily to multihop networks, and therefore on the periphery of our study since the Smart-Condo project follows a single-hop model, i.e., every node deployed in the condo communicates directly with the sink node (base-station). This is because: 1) the total area of the potential surveillance region is fairly small (in the current setup, a one bedroom apartment of 10.6×6.3 m) so that all the nodes lie within the communication range of each other and 2) we are interested in real-time updates, whereas a multihop 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.

The data fidelity objective can be approached as a data fusion problem. Multiple sensors are placed in the vicinity of the monitored phenomenon in such a way that guarantees that the data obtained from the combined sensor readings are 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.* [14] formulate a combinatorial optimization problem with the objective of maximizing the overall detection probability under a deployment cost constraint. They showed the decision version of the problem to be NP-complete and used a 2-D genetic algorithm to generate an approximate solution. Krause *et al.* [15] capture several optimization objectives; they aim to maximize "informativeness" of a sensor placement while minimizing communication costs. They use the data collected during a pilot (nonoptimized) deployment to define probabilistic models for both predictive quality of the placement, i.e., the ability to predict values at locations where no sensors are placed, and the quality of communication links between sensors. 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.

In the narrower research area of indoor sensor placement, we focus on placement problems that consider obstacles affecting the sensing range of assumed nodes. Dhillon and Chakrabarty [16] incorporate information about obstacles into 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.* [17] consider 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 ray-tracing 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 optimization formulation is the work of Wang *et al.* [18]. They exploit the fact that in many applications the importance of sensed information varies across the sensing field. Thus, they define points of interest which should be optimally covered. Such a strategy may not yield full coverage but it does maximize the sensed information utility. We too would like to sparsely distribute sensors in the indoor environment and yet be able to sense the most valuable data. 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 in the occupant's daily routine.

Some similarities to our approach can also be seen in the MavHome location-aware predictive framework [19], which is another example of a smart-home technology that aims to anticipate the occupant's desires and provide proactive resource management and on-demand operation of actuators. A critical assumption 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, given our experience with short-term deployments (e.g., some past trials lasted only 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 resulting mobility model.

III. Mobility Modeling

Mobility patterns in an indoor environment resemble typical road traffic with its bottlenecks, conjunctions, and more and less traveled 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 traveled locations. Even if the budget of devices suffices for full coverage, but the space is not uniformly utilized, localization errors occurring in more traveled areas will have greater impact on the overall accuracy than localization errors in the rest of the space. Following this reasoning, we aim at optimizing sensor placement with respect to how frequently the occupant visits various regions.

Typically, information about the occupant mobility patterns is not available in the predeployment phase, and to have a pilot deployment to collect such data is costly and impractical. Therefore, we propose a framework for mobility modeling that relies on mere knowledge of the architectural drawing of the deployment space, i.e., a floorplan. We also want to take advantage of contextual information that can be inferred from positions of furniture and amenities depicted on the floorplan. Next we apply a number of transformations to the floorplan that let us extract this information.

A. Floorplan Color Coding

Consider the floorplan of the Smart-Condo (Fig. 1). We assume that 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 $\rightarrow$ toilet, stove $\rightarrow$ dining table, entrance door $\rightarrow$ recliner). We first identify all objects of interest (e.g., bed, fridge, stove, sink, toilet) and augment the floorplan using a custom color coding scheme. This color coding procedure is the only manual step of our methodology (easily performed using graphics software).

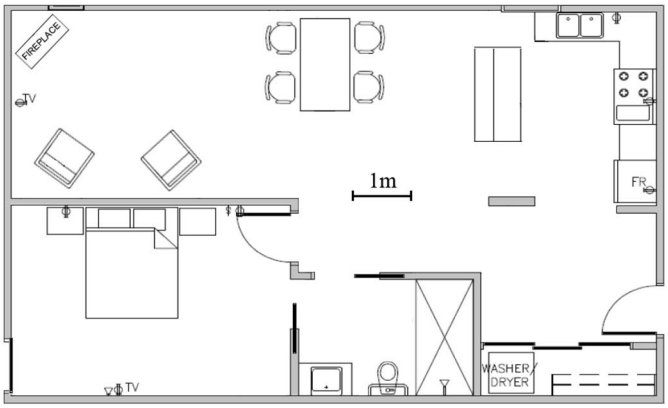

Fig. 1. Smart-Condo floorplan (gray walls, black doors).

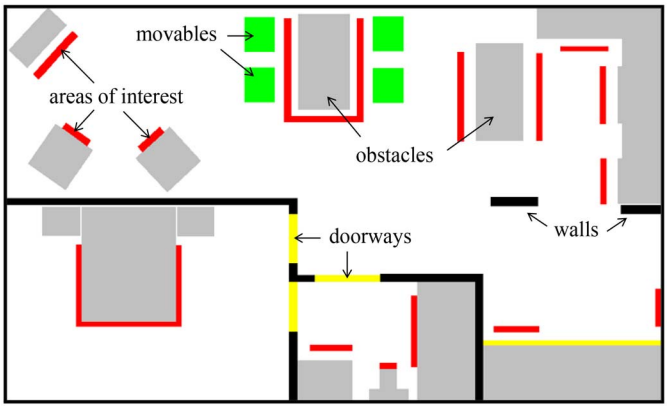

Fig. 2. Augmented floorplan.

Fig. 2 shows the result of the procedure and contains five types of objects: 1) walls; 2) doorways/sliding doors; 3) impassable fixed-position obstacles, i.e., all large pieces of furniture; 4) obstacles that may frequently change positions, e.g., chairs; and 5) areas of interest associated with every object of interest (narrow strips next to the articles of furniture).

The fourth type of object (henceforth referred to as the movables) is of particular interest. We assume that these objects remain within a certain area of the space but may occasionally be moved to random locations. To model this behavior of the movables, we define the borders of the area that confines their displacement, and then determine all possible locations within that area that satisfy the dimensions of the movables. Fig. 3 depicts the grid of such positions.

The last type of object represents areas where the occupant is likely to end up while trying to approach a piece of furniture. 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 harder to predict which side of the bed will be preferred by the occupant; therefore we define the respective 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 corresponding objects is that the former

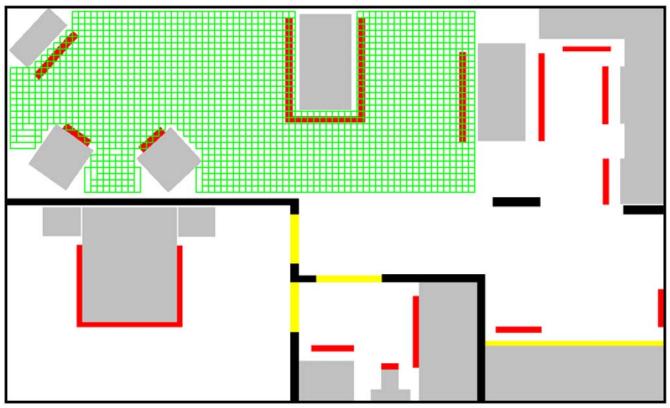

Fig. 3. Grid of all positions that can be potentially occupied by the movable obstacles.

are considered walkable and are used for constructing paths between otherwise impassable objects.

Having defined all the necessary objects, the augmented floorplan serves as input to 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.

B. Calculating Visitation Frequencies

We want to model a variety of realistic paths between all pairs of objects of interest. Once the augmented floorplan 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 path finding algorithm (PFA), i.e., a generic implementation of A* [20]. All the grid points except walls and impassable obstacles are considered walkable (assuming the doors can always be opened if needed).

Given that the PFA always produces the shortest path, we may end up with recurring straight-line paths that do not accurately represent the actual paths a human would choose. To mitigate this effect, we blockade a number of walkable points at random for each execution of the PFA. The number of blocked points is configurable; more blockage generally leads to greater variance in the resulting paths but also increases the PFA running time. We tested a range of values and found blocking 30% of the walkable points gave a good tradeoff.

Fig. 4 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) blockage of a number of walkable points. This model also takes into account a body diameter of the assumed occupant, which helps to smooth out the 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 depends on the number of times a particular object participates in path construction. This number has to represent the object's importance in a typical daily

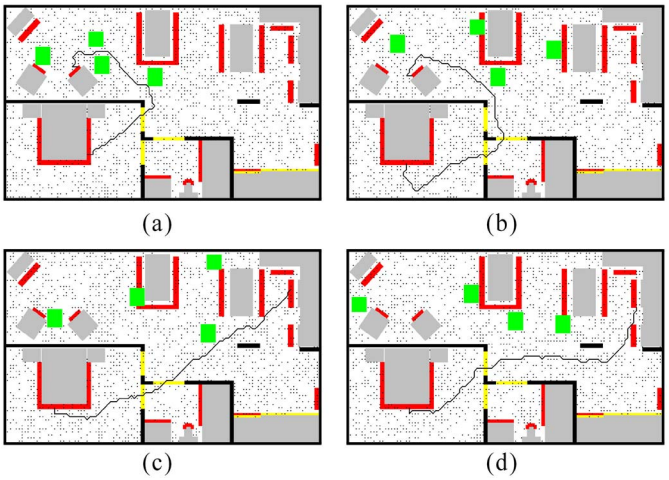

Fig. 4. Effects of randomization on the results of path-finding (black dots are blocked grid points). (a) Path 1.1: bed—recliner. (b) Path 1.2: bed—recliner. (c) Path 2.1: bed—stove. (d) Path 2.2: bed—stove.

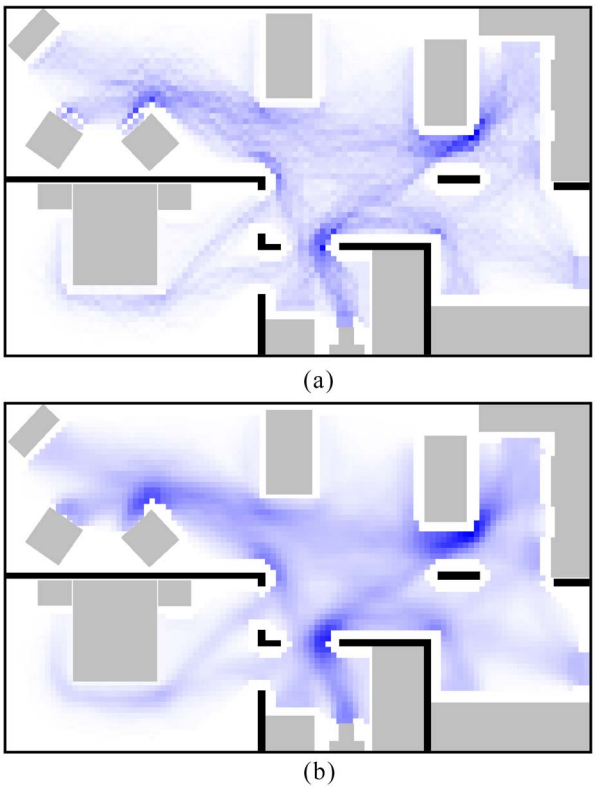

Fig. 5. Heatmaps of anticipated mobility patterns before/after "smoothing." (a) Raw heatmap. (b) Smoothed heatmap.

routine of the occupant. For example, if the occupant uses the bed twice a day, visits the kitchen three times a day, and uses the washing machine once a week, then those objects can be assigned weights at a ratio 14:21:1, respectively, and their occurrences during multiple runs of the PFA will 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 assignment, although in this paper 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 future real-world trials.

The resulting mobility model can be represented as a heatmap of the counts of the times that the assumed occupant visits respective grid points [Fig. 5(a)]. 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 the examples presented here). This heatmap, however, suffers from a visible defect: some grid points become bottlenecks while others never get visited, although located in "hot" areas, due to imperfections in grid approximation. This phenomenon may have a negative impact on the generation of sensor placements, and we therefore eliminate it by "smoothing" the heatmap. Fig. 5(b) depicts the final heatmap further used for finding placements optimized for coverage of the "hottest" points.

To formalize a definition of the mobility model, let N denote the number of walkable grid points, and h_i denote the "heat" score of the ith grid point, i.e., equal to the visitation frequency from the smoothed heatmap. The mobility model is therefore defined as a set of values $\{h_1, \ldots, h_N\}$.

In the following section, we define the sensor-coverage model, which together with the mobility model will be used in the formulation of the optimization problem.

IV. Sensor-Coverage Model

To define the sensor-coverage model, we first review the PIR sensor operating principles. Next, we touch upon indoor localization strategies and propose such a coverage model that works toward achieving higher localization accuracy.

A. PIR Sensor Operating Principles

The motion sensors used in our platform are commercially available PIR (pyroelectric) sensors chosen for their miniature size, reliable human presence detection and low energy consumption [21]. The sensor collects incident infrared radiation from within their coverage area. Its output is binary: 0 for no motion, and 1 when the sensor detects an increase in the amount of radiation due to a moving object entering the coverage area. Given a single sensor, the position of the moving object cannot be discerned with any higher precision than the "radius" of the sensor footprint.

One great advantage of these sensors over the majority of alternative localization technologies is that the person being tracked does not need to carry additional devices and may remain unaware of the surrounding sensor infrastructure. Therefore, they have been widely used in a number of indoor localization studies [22]–[24].

B. Sensor Footprint

The PIR sensors detect motion in line of sight, hence, the shape of sensor footprint is defined by the mounting position, orientation, and obstruction effects of the walls and

furniture. To reduce the search space of candidate sensor positions/orientations, we assume that the sensors are mounted on the ceiling and restrict the possible sensor orientations to orthogonal with respect to the floor plane, e.g., a cone projects into a circle. We are less interested in the circular projections since, despite a great amount of previous research concerned with this particular sensing model [14], [22], [23], exact circular coverage is a rarity. Instead, we choose to model sensors whose volumetric coverage shape is a rectangular pyramid with the base of 2×1.4 m in cross section at 2 m-height [21]. We consider two orientations of a ceiling-mounted sensor: the longer side of the base of the pyramid is either parallel or perpendicular to the longer side of the floorplan. Technically, the PIR sensor specifications 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 paper to our previous practical experience.

Note that 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 sensor footprint resulting from obstruction by applying a ray tracing technique to every grid point within the borders of the default sensor projection. Fig. 6(a) visualizes the ray tracing procedure; Fig. 6(b) shows the grid approximation used in further calculations instead of the actual polygons [Fig. 6(c)]. This implies that our sensor-placement method 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).

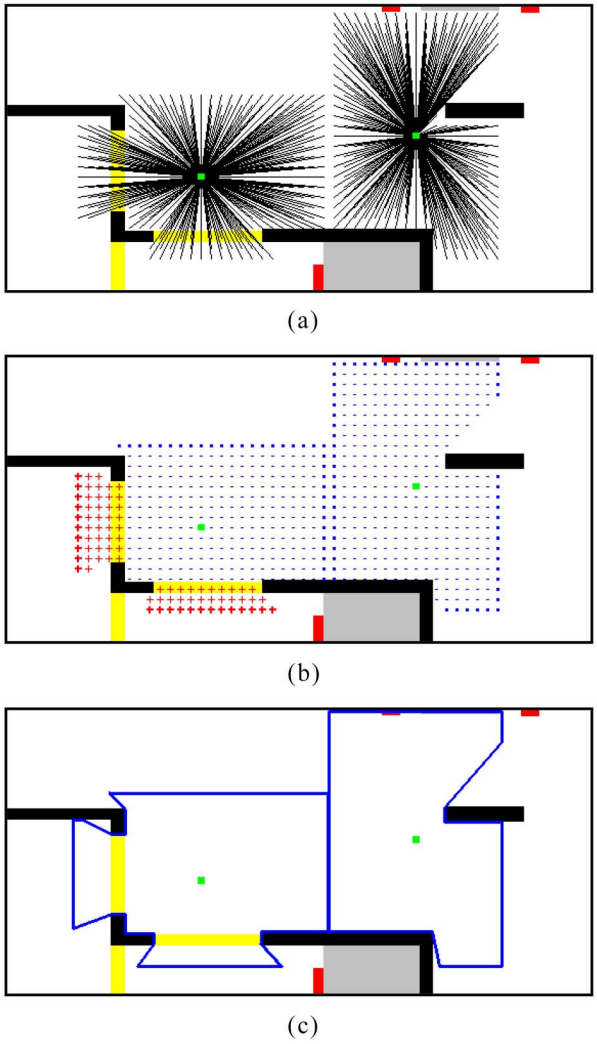

Fig. 6. Various sensor-coverage representations. (a) Ray tracing applied to grid points. (b) Grid approximation of sensor coverage (crosses indicate points seen through a doorway). (c) Real shape of sensor-coverage projections.

C. Modeling Sensor Coverage and Obstacles

Having determined the shape of the sensor projections, we can proceed to define the sensor-coverage model through which to capture the probability of a motion event being detected by a particular sensor. Based on published sensor operating principles [21] and a number of empirical studies we have performed, the probability of detecting a true positive motion event by a PIR sensor is uniform within its footprint and approaches 1. Therefore, a naive model would represent the coverage of a given sensor by assigning 1 s to the grid points within the sensor footprint and 0 s outside.

We further extend the sensor-coverage model by incorporating information about obstacles, in a way similar to that described by Dhillon and Chakrabarty [16]. To explain the impact of obstacles on our sensor model we refer to Fig. 6(b): the 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 motion detection in those "seen-through-doorway" points over the same period. This directly impacts the quality of coverage in such areas, and is reflected in our sensor-coverage model. More formally, let $c_i^{\{s_j\}}$ denote the amount of coverage units, i.e., an abstract measure of coverage quality, allocated to the arbitrary ith point by placing sensor s_j on the grid. We ultimately define the coverage model of sensor s_j through a set of values $\{c_1^{\{s_j\}}, \ldots, c_N^{\{s_j\}}\}$ such that

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

where $i = 1, \ldots, N$ are indices of grid points, $\mathbb{C}_{s_j}$ is a set of grid points covered by the sensor after ray tracing has been applied [see Fig. 6(b)], $\mathbb{D}_{s_j} \subsetneq \mathbb{C}_{s_j}$ is a set of points seen by the sensor through a doorway [the crosses in Fig. 6(b)], and p_{od} is the probability that the doors are open. If no statistics about doors usage are available, we will simply assume that they are open 50% of the time, i.e., $p_{\text{od}} = 0.5$.

D. Sensor Overlap and Localization Accuracy

Note that merely detecting motion with a single sensor does not typically translate into high localization accuracy.

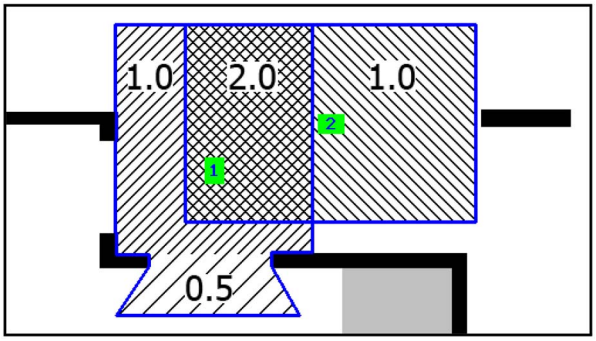

Fig. 7. Allocation of coverage units.

For accurate indoor localization it is essential that the target is sensed by multiple sensors simultaneously, especially if such popular techniques as triangulation or trilateration are used. As with these techniques, we may achieve better localization accuracy with PIR sensors if several sensor footprints overlap in high-interest areas. That is, upon fusing the readings from the overlapping sensors, we will be able to infer that motion occurred within the intersection region that is just a fraction of a single sensor footprint, thus allowing us to predict the location of the motion event more accurately.

We propose to express the increase of localization accuracy due to overlapping sensors through accumulation of coverage units in grid points covered by multiple sensors. That is, a cumulative coverage score of the ith point, covered by a set of sensors $\{s_1, \ldots, s_k\}$, is defined as follows:

$$c_i^{\{s_1,\ldots,s_k\}} = \sum_{j=1}^{k} c_i^{\{s_j\}}. \tag{2}$$

In the most trivial case, when m sensors cover the ith 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. Fig. 7 illustrates how coverage units are calculated for sensors s_1 and s_2 (assuming $p_{\text{od}} = 0.5$)

$$\begin{aligned} c_i^{\{s_1\}} &= 0.5,\ \forall i \in \mathbb{D}_{s_1} \\ c_i^{\{s_1\}} &= 1.0,\ \forall i \in \mathbb{C}_{s_1} \setminus \mathbb{D}_{s_1} \\ c_i^{\{s_1,s_2\}} &= 2.0,\ \forall i \in (\mathbb{C}_{s_1} \setminus \mathbb{D}_{s_1}) \cap \mathbb{C}_{s_2}. \end{aligned}$$

Since every sensor is associated with a set of values $c_i^{\{s_j\}}$ defined for N grid points, 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.

Having defined both the mobility model and the sensor-coverage model, we proceed with the formulation of an optimization problem and a placement algorithm.

V. Sensor Placement Optimization

Let $\mathbb{S}$ be a set of potential sensor locations $\{s_1, \ldots, s_{|\mathbb{S}|}\}$ identified by tuples (x_j, y_j, z_j, o_j) for $j = 1, \ldots, |\mathbb{S}|$, where x_j and y_j denote a pair of coordinates on the ceiling, z_j denotes the ceiling height, and o_j denotes the sensor projection horizontal orientation (0° or 90°). We assume that no two sensors can occupy the same grid cell. Note that, for the purpose of simplicity, we also assume that the ceiling height is fixed (at 2.5 m) and therefore omit the z-coordinate. Variable height (e.g., sloped ceiling) can also be accommodated by making the z-coordinate explicitly part of the calculations, under the assumption that the sensor main optical axis is 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.[1] Therefore, sensor coordinates are aligned with coordinates of the walkable grid points. In preparation for the next steps of the algorithm we determine $\mathbb{C}_{s_j}$ and $\mathbb{D}_{s_j}$ for all $s_j \in \mathbb{S}$, thus generating the necessary sensor-coverage models.

Given the mobility model as a set of heat-score values $\{h_1, \ldots, h_N\}$, and $|\mathbb{S}|$ sensor-coverage models, we are interested in identifying a subset of sensor locations $\mathbb{P} \subsetneq \mathbb{S}$ that yields a set of coverage score values $\{c_1^{\mathbb{P}}, \ldots, c_N^{\mathbb{P}}\}$ proportional to the respective heat-score values. In other words, we would like to allocate more coverage units to more frequently visited areas, i.e., with higher heatmap values, thus ensuring overall higher localization accuracy according to the line of reasoning in Section IV-D. Unfortunately, if we simply follow this intuition, and the range of the heatmap values is large (orders of magnitude between the lower and upper values), too many sensors may end up overlapping over the high-priority areas while sacrificing coverage in the rest of the space. Such situations are especially undesirable, and are likely to occur when the device budget is limited. Therefore, we need to strike a trade-off between redundant coverage of the most traveled areas and sufficient coverage of the less traveled areas.

A. Coverage Utility

To address the issue of unbalanced coverage, we map the whole range of heatmap values into a small range of coverage utility values. Coverage utility is a measure of the importance of a grid point and is indicative of a preferable coverage score, i.e., such a cumulative coverage score per grid point that we would like to achieve with sensor placement. That is, if c_i^* denotes the coverage utility of the ith point, then by placing k sensors we would like to achieve $c_i^{\{s_1,\ldots,s_k\}} \to c_i^*$.

There are different ways to define a heatmap-to-coverage-utility mapping. In our framework, we define a parameter $c_{\max}$ as the maximum preferable coverage score per grid point. We map all h_i values into $c_{\max} + 1$ buckets with labels $0, 1, \ldots, c_{\max}$, thus translating highly varied h_i values into a limited range of c_i^* values. This way we balance the gap between the lowest and highest values of the heatmap, while retaining relative gradation. Essentially, the ratio of c_i^* scores assigned to the hottest and least hot (but nonzero h_i) points is guaranteed to be $c_{\max}$. This can be interpreted as if the most frequently traveled 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

[1] Practically, the grid of sensor positions is likely to have coarser granularity than the mobility model grid, e.g., to align sensors with the tiles of a drop ceiling, thus even further reducing the search space.

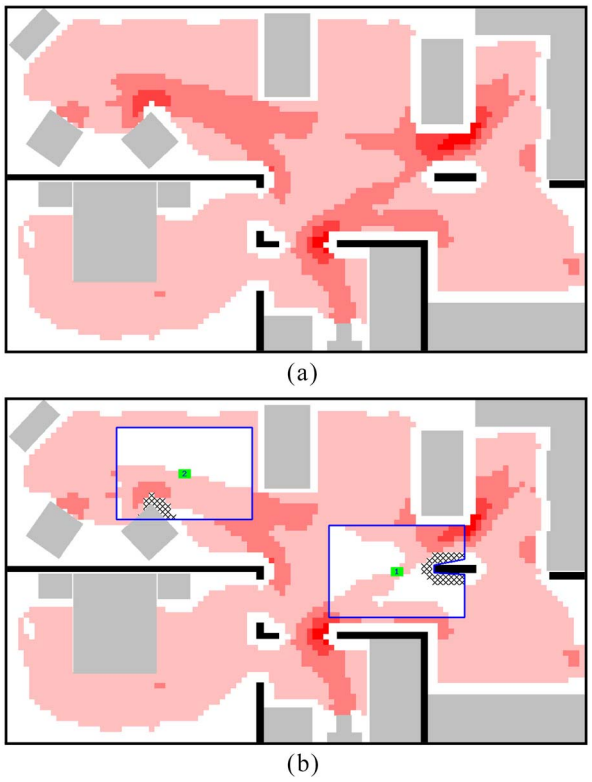

Fig. 8. Coverage utility heatmap before and after having placed sensors. (a) Heatmap of coverage utility values, $c_{\max} = 4$. (b) Updated coverage utility values with two sensors placed, crosshatched areas indicate negative values.

frequently traveled 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 a color quantization of the original heatmap. Consider Fig. 8(a): the heatmap of coverage utility values is colored in four different shades of a base color (due to $c_{\max} = 4$), to represent corresponding values $c_i^* > 0$, and white for all grid cells with $c_i^* = 0$.

Another issue addressed by the choice of $c_{\max}$ is the size of high-priority regions relative to the size of sensor footprints; clearly, the higher the $c_{\max}$, the smaller the eventual top-priority regions. As mentioned in Section IV-D, higher localization accuracy can be achieved by overlapping sensors if the overlapping region is only a fraction of the size of any of the contributing sensors' footprints. This strategy fails if the area of overlap approaches the area of a single sensor footprint, i.e., sensors are clustered too closely together. Therefore, the value of $c_{\max}$ is chosen so as to yield several small groups of high-priority points corresponding to the most traveled areas, and a large group of low-priority points, which simply outlines the overall walkable area. As long as clusters of top-priority points are significantly smaller than a sensor footprint, our placement algorithm tends to avoid placing sensors, overlapping over these clusters, too close to each other.

As with the vector $\mathbf{c}^{\{s_1,\ldots,s_k\}}$ representing a cumulative coverage model defined in Section IV-D, we define an N-dimensional vector $\mathbf{c}^* = (c_1^*, \ldots, c_N^*)$ to denote a coverage utility model of the space.

B. Objective Function

Our goal is to define an objective function in terms of coverage utility and sensor-coverage models. To that end, we express the utility gained by adding an individual sensor, as a function over the set of already placed sensors. Let us 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

$$\delta_{s_1}^{\varnothing} = \sum_{i=1}^{N} c_i^{\{s_1\}} c_i^*. \tag{3}$$

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. The expression (3) can also be rewritten in vector terms defined in Sections IV-D and V-A as $\delta_{s_1}^{\varnothing} = \mathbf{s_1} \cdot \mathbf{c}^*$.

Note that once sensor s_1 has been placed, another sensor placed in a nearby 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 as the dot product of the 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:

$$\begin{aligned} \delta_{s_2}^{\{s_1\}} &= \delta_{s_2}^{\varnothing} - \mathbf{s_2} \cdot \mathbf{s_1} \\ &= \mathbf{s_2} \cdot \mathbf{c}^* - \mathbf{s_2} \cdot \mathbf{s_1} \\ &= \mathbf{s_2} \cdot (\mathbf{c}^* - \mathbf{s_1}). \end{aligned} \tag{4}$$

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

$$\begin{aligned} \delta_{s_3}^{\{s_1,s_2\}} &= \delta_{s_3}^{\varnothing} - \mathbf{s_3} \cdot \mathbf{s_1} - \mathbf{s_3} \cdot \mathbf{s_2} \\ &= \mathbf{s_3} \cdot \mathbf{c}^* - \mathbf{s_3} \cdot \mathbf{s_1} - \mathbf{s_3} \cdot \mathbf{s_2} \\ &= \mathbf{s_3} \cdot (\mathbf{c}^* - \mathbf{s_1} - \mathbf{s_2}) \\ &= 7\mathbf{s_3} \cdot \left(\mathbf{c}^* - \sum_{1 \le j < 3} \mathbf{s_j} \right). \end{aligned} \tag{5}$$

We formalize this intuition into a closed-form expression for the utility score of the mth sensor after $m-1$ sensors have been placed

$$\delta_{s_m}^{\{s_1,\ldots,s_{m-1}\}} = \mathbf{s_m} \cdot \left(\mathbf{c}^* - \sum_{1 \le j < m} \mathbf{s_j} \right). \tag{6}$$

Note that the sum of sensor vectors $\sum_{1 \le j < m} \mathbf{s_j}$ is another vector whose ith component is a sum of coverage units allocated to the ith grid point by placing sensors s_1 through s_{m-1}, i.e., $\sum_{1 \le j < m} c_i^{\{s_j\}}$. Using (2) and replacing the scalar values with vector notation we rewrite expression (6) as follows:

$$\delta_{s_m}^{\{s_1,\ldots,s_{m-1}\}} = \mathbf{s_m} \cdot \left(\mathbf{c}^* - \mathbf{c}^{\{s_1,\ldots,s_{m-1}\}} \right) \tag{7}$$

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

We interpret (7) 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. Fig. 8(b) 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 (7). Zero-utility points are typically points that have not been traversed by the PFA: 1) due to a body-diameter constraint; 2) because they are unreachable; or 3) they lie off the typical paths (e.g., room corners). Thus, coverage units allocated to zero-utility points are, in effect, wasted. Negative utility in this case can be seen as a penalty that guides the algorithm to avoid further waste of coverage units.

According to (7) it is generally possible to obtain a negative utility score for sensor s_m. However, the notion of negative utility is inapplicable in a domain where additional sensors simply provide extra information. Therefore, we apply positive thresholding to the expression (7) so that the utility score of a sensor can never be negative, and redefine the utility score function as follows:

$$\delta_{s_m}^{\{s_1,\ldots,s_{m-1}\}} = \max\left\{0;\ \mathbf{s_m}\cdot\left(\mathbf{c}^* - \mathbf{c}^{\{s_1,\ldots,s_{m-1}\}}\right)\right\}. \tag{8}$$

Note that the algorithm may reach a state where many sensors have been placed and the total number of allocated coverage units $\sum_{i=1}^{N}\sum_{1\le j<m} c_i^{\{s_j\}}$ exceeds the total amount of coverage utility $\sum_{i=1}^{N} c_i^*$ but the budget of sensors has not been exhausted yet; this is an indication that the range of c_i^* values is inadequate for the given number of sensors. Although we resort to positive thresholding in (8), it is preferable to avoid this situation by adjusting the range of c_i^*, i.e., by increasing the $c_{\max}$ parameter (Section V-A).

Finally, the optimal solution to the sensor-placement problem is the 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 using (6)

$$\Phi(\{s_1,\ldots,s_k\}) = \sum_{i=1}^{k}\mathbf{s_i}\cdot\left(\mathbf{c}^* - \sum_{1\le j<i}\mathbf{s_j}\right) \tag{9}$$

$$= \sum_{i=1}^{k}\left(\mathbf{s_i}\cdot\mathbf{c}^* - \mathbf{s_i}\cdot\sum_{1\le j<i}\mathbf{s_j}\right) \tag{10}$$

$$= \sum_{i=1}^{k}\mathbf{s_i}\cdot\mathbf{c}^* - \sum_{1\le j<i\le k}\mathbf{s_i}\cdot\mathbf{s_j}. \tag{11}$$

The second term on (11) is a summation of all possible distinct pairwise dot-products of sensor vectors, which can be interpreted as a total penalty for overlapping sensors. Eventually, we arrive at a closed-form expression for the objective function that we want to maximize. 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. Having established our objective, we comment on the computational difficulty of optimizing it in the next section.

C. NP-Hardness and Approximation Algorithm

The objective (11) is NP-hard to optimize for a fixed number of sensors k. The proof is too long for inclusion in the main text, and so we refer the reader to Theorem 1 in the Appendix. As a summary overview, we outline a polynomial time reduction of the exact cover problem, known to be NP-complete, to an instance of our optimization problem. Given a set of elements $\mathbb{U}$ and a collection $\mathbb{A}$ of subsets of $\mathbb{U}$, an exact cover is a subcollection $\mathbb{A}'$ of $\mathbb{A}$ comprised of mutually disjoint subsets that cover every element in $\mathbb{U}$. Our reduction works by mapping the semantics of "element coverage" by a particular set $\mathbb{A}_j$ from a collection $\mathbb{A}$ to "walkable point" coverage by an associated sensor s_j. Using a particular assignment of coverage utility values and sensor-coverage models, we obtain such an instance of our optimization problem for which exact coverings are strictly preferred due to a penalty associated with redundant coverage by any two overlapping sensors.

The immediate result is that finding a globally optimal solution is computationally prohibitive for nontrivial instances. We will therefore resort to an approximation algorithm. We propose a greedy algorithm that adds one sensor at a time to maximize utility gain at each iteration. The algorithm takes k iterations. At the mth iteration ($m = 1,\ldots,k$) we recalculate the utility scores of all sensor candidates from $\mathbb{S}$ using (8) and add a sensor with the maximum utility score. If multiple sensor candidates evaluate to the maximum score value, we choose the one with the largest footprint, measured as the number of covered grid points. A different tie-breaking mechanism can be employed depending on the purpose of the deployment, and as such, it allows for additional expert knowledge to be encoded into the algorithm. The algorithm terminates after the kth sensor has been placed.

D. Near-Optimality Guarantee

Nemhauser *et al.* [25] showed that, if the objective function of an optimization problem exhibits the properties of submodularity and monotonicity, a greedy algorithm produces a near-optimal solution. In this context, a near-optimal solution is defined as one that lies within a factor of two of the optimal solution for minimization problems, or such that is $\ge 50\%$ of optimal for maximization problems.

The concept of submodularity has been exploited in a number of sensor placement studies [15], [26]. 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. We prove that indeed our objective function (11) is both submodular and

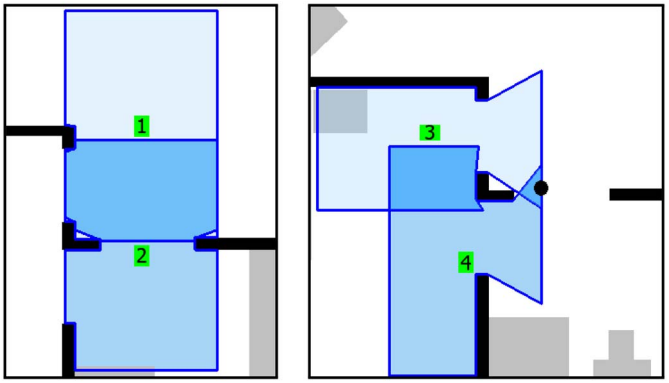

Fig. 9. Specifics of sensor footprints reaching through doorways. (a) Sensor 2's coverage is less reliable than sensor 1's coverage due to the possibility of the door being closed. (b) Two sensors' footprints overlap in regions corresponding to two different rooms.

monotonic. Detailed proofs are given as Theorems 2 and 3 in the Appendix.

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, as outlined in Section V-C, then $\Phi(\mathbb{S}_{\text{g}}) \geq (1 - 1/e) \cdot \Phi(\mathbb{S}_{\text{opt}}) \approx 0.63 \cdot \Phi(\mathbb{S}_{\text{opt}})$. 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.* [25].

VI. Parameter Selection

In this section, we discuss how the sensor placements produced 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).

A. Doors and Context Ambiguity

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 through doors [Fig. 9(a)]. Let us also assume that $\sum_{i \in \mathbb{C}_{s_1}} c_i^* = \sum_{j \in \mathbb{C}_{s_2}} c_j^*$, i.e., the total coverage utility within each sensor footprint is equal. Using (3), we can calculate the utility gain for each sensor placed independently as $\sum_{i=1}^{N} c_i^{\{s_j\}} c_i^*$, assuming no other sensors have been placed on the grid yet. Plugging sensor-coverage model values $c_i^{\{s_j\}}$ from (1) for each sensor, respectively, sensor s_2 will receive a remarkably lower score than s_1 due to a significant number of $c_i^{\{s_j\}}$ values equal to $p_{\text{od}} < 1$. In other words, by assigning lower coverage units to seen-through-doorway areas, the sensor-coverage model effectively penalizes potential placements that yield unreliable doorway coverage.

This penalizing side-effect can be exploited to avoid undesirable artifacts of the type shown in Fig. 9(b). Sensors 3 and 4 belong to the same room and overlap in two disjoint regions (the darkest shade in the image): 1) one lies within the same room and 2) 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 caregivers observing a patient. 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.

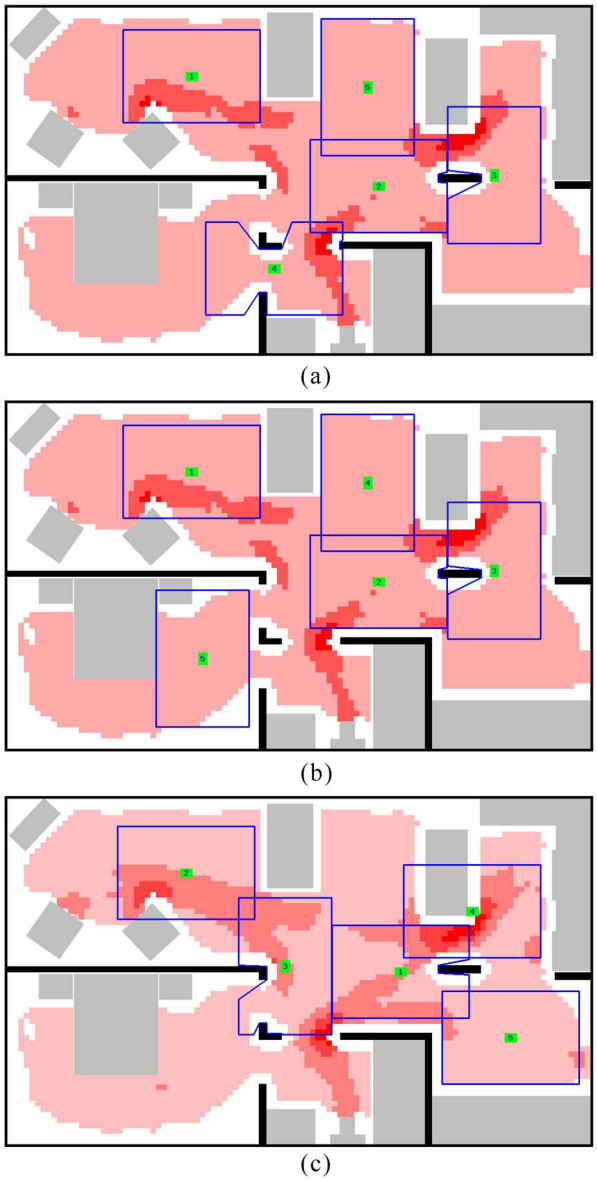

Fig. 10. Examples of placements with different values of $c_{\max}$ and p_{od}. (a) $c_{\max} = 3$, $p_{\text{od}} = 1$. (b) $c_{\max} = 3$, $p_{\text{od}} = 0.1$. (c) $c_{\max} = 4$, $p_{\text{od}} = 0.1$.

B. Illustrative Results

Fig. 10 illustrates three placements generated for a budget of five sensors and different $c_{\max}$ and p_{od} values. The

background of each image is colored according to the corresponding $c_{\max}$ value: Fig. 10(a) and (b) have three shades of color, and Fig. 10(c) has four.

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 rooms, about half of the footprint area of the sensor placed in the bathroom in Fig. 10(a) reaches into the adjacent rooms. Once the penalty is enabled [Fig. 10(b)], the same sensor is "pushed" out of the bathroom into the bedroom. The placement in Fig. 10(c) has been generated with the same p_{od} as in Fig. 10(b) 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 nontrivial. 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. In the next two sections, we describe our simulation methodology and proceed with an experimental evaluation.

VII. Simulation-Based Evaluation

Given a particular sensor-placement configuration, we evaluate the accuracy of the Smart-Condo localization component through simulation. 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 precision) and allows insights into alternative deployment strategies.

Our simulation methodology involves the following steps.

1) We build a 2-D model of the space from a floorplan.
2) Given a sensor placement, we integrate the 2-D sensor-coverage map into the model of the space.
3) An avatar–simulacrum of the occupant–is scripted (or manually controlled) to walk through the space. The avatar trace is recorded by the simulator as a sequence of <timestamp, location> tuples.
4) The trace is considered in the context of the sensor-coverage map, which results into a stream of artificial sensor events supplied to the localization component.
5) The original avatar trace is compared against the sequence of locations inferred by the localization component, to assess the accuracy of the sensor configuration.

The simulator allows us to generate traces that realistically represent the occupant's daily routine. Fig. 11(a) shows such a trace; each black dot is a <timestamp, location> tuple.

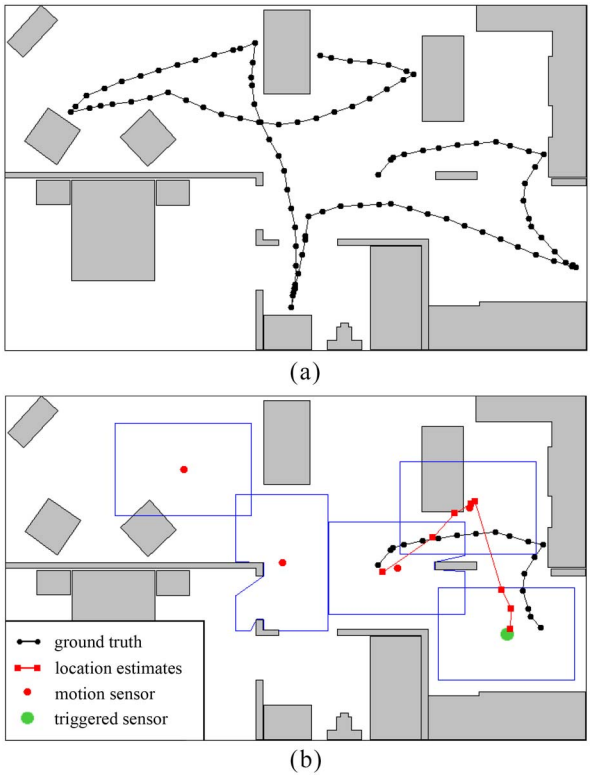

Fig. 11. Simulation sequence. (a) A fragment of an artificial "daily-routine" trace. (b) Comparison of the ground truth and estimated locations.

Fig. 11(b) depicts how the active sensor is determined by superimposing the trace tuples on the 2-D sensor-coverage map. It also illustrates how the localization algorithm used in our software generates its predictions. The algorithm's initial coarse estimate is the center of mass of the polygon corresponding to the overlap of the most recently triggered 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 history 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. At the moment, the algorithm does not exclude the area occupied by obstacles from possible locations of the moving target. We, however, 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. For more details, the interested reader should refer to [8].

VIII. Experiments

We systematically evaluate the localization accuracy of alternative sensor placements through simulation. Sensor placements are compared by two factors: 1) placement strategy and 2) cardinality. Based on the dimensions of the assumed deployment space and sensor footprints, we focus on cardinality in a range from 5 to 20 sensors.

We generated ten artificial traces of various length representing typical daily routines, as described in Section VII and

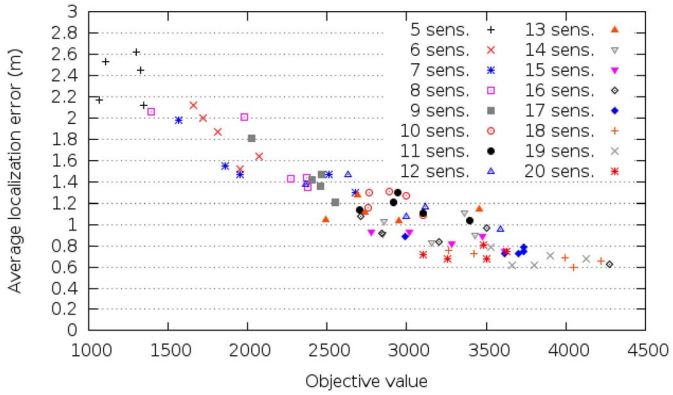

Fig. 12. Localization error with respect to objective values of randomized placements.

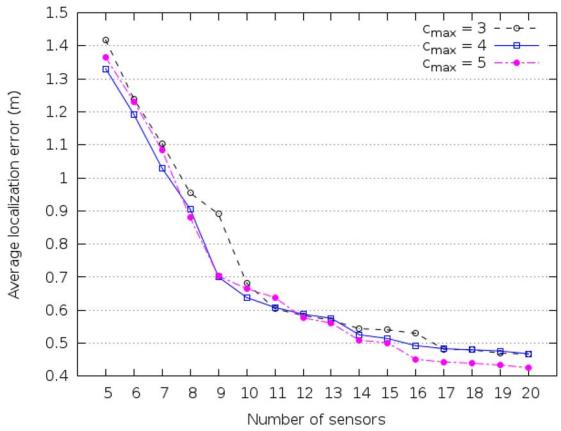

Fig. 13. Localization error with respect to $c_{\max}$.

illustrated in Fig. 11(a). For each <timestamp, location> tuple from a trace, 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 standard error as well as three quartiles of error values.

A. Objective Value and Localization Accuracy

To demonstrate that higher objective values as computed using (11) translate into lower localization errors, we perform the following test. We generate five randomized placements for $k = 5, \ldots, 20$ sensors 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. In this manner, we avoid occurrences of all k sensors being clustered too close to each other.

Fig. 12 illustrates the relationship between the objective values calculated with $c_{\max} = 5$ using (11) for 80 randomized placements and their respective localization performance on ten daily routine traces. The plot reveals a strong trend: the localization errors drop with the increase in the objective values, lending empirical weight to the validity of our objective function.

B. Experimental Setup

We compare three placement strategies: 1) placements generated by our algorithm, i.e., optimized; 2) placements crafted manually; and 3) randomized placements. The following subsections describe the placement generation in more detail.

1) Optimized Placements: According to our reasoning in Section V-A, the localization accuracy 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 was based on the following observations: with $c_{\max} < 3$ the impact of coverage utility assignment on the results of the placement algorithm is hardly noticeable; with $c_{\max} > 5$ 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 generated three placements, i.e., with $c_{\max} = 3, 4, 5$, for each $k = 5, \ldots, 20$ and tested them against the artificial traces.[2] Fig. 13 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., $c_{\max} = 3$ and $c_{\max} = 4$ for 11 and 17–20 sensors), we observe a general trend: lower error rates are achieved with $c_{\max} = 4$ for $k \leq 11$ sensors and with $c_{\max} = 5$ 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 nonwalkable 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 $c_{\max} = 4$ for $k \leq 11$ and $c_{\max} = 5$ for $11 < k \leq 20$. Examples of optimized placements for $k = 9$ and $k = 20$ are shown in Fig. 14.

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, one would likely attempt to design placements with a fairly regular "grid" of overlapping regions. Fig. 15 shows two examples of manually crafted placements: nine sensors, uniformly covering the space with gaps between sensors, and 20 sensors, regularly overlapping.

We manually designed 15 placements for each $k = 5, \ldots, 20$ following the aforementioned considerations. Our final goal was to compare the performance of the optimized

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

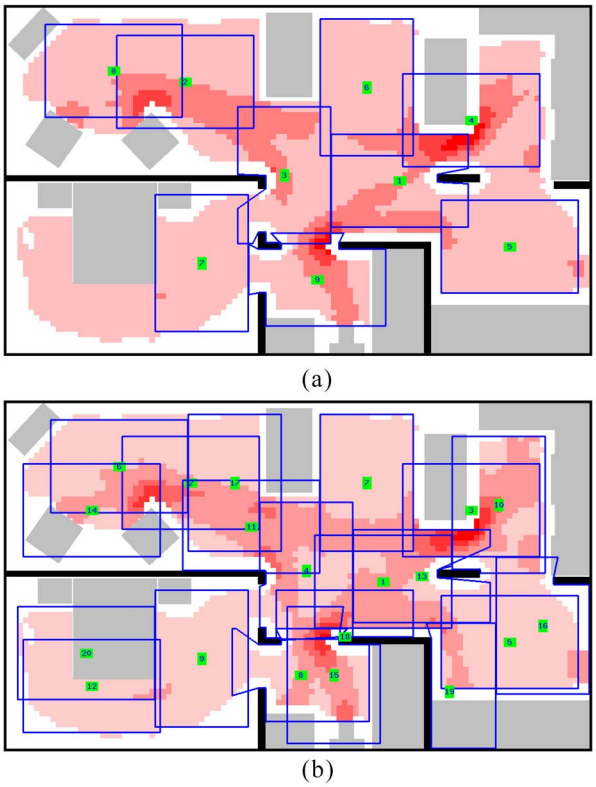

Fig. 14. Examples of optimized placements. (a) 9 sensors, $c_{\max} = 4$. (b) 20 sensors, $c_{\max} = 5$.

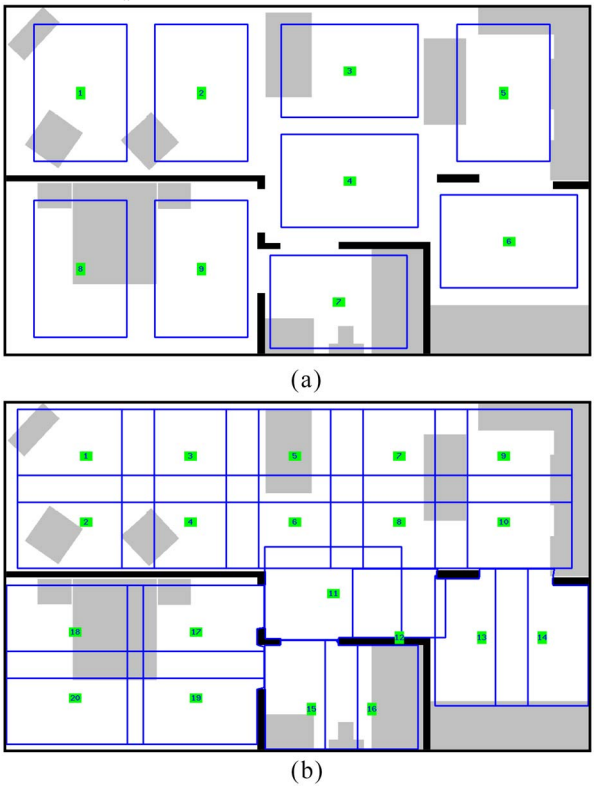

Fig. 15. Examples of manual placements achieving uniform coverage. (a) 9 sensors. (b) 20 sensors.

placements 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 toward

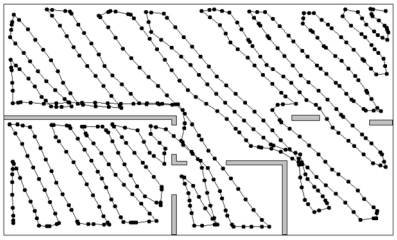

Fig. 16. "Zigzag" trace that systematically covers the space and ignores furniture.

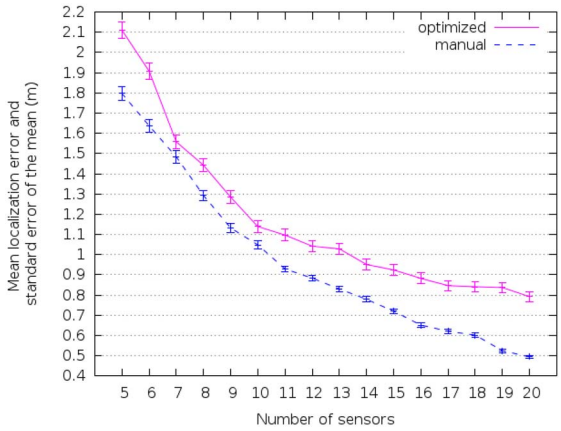

Fig. 17. Localization error of manual versus optimized placements tested on zigzag traces.

a particular data set. We generated a number of traces that systematically cover the entire space in a zigzag-like fashion (see Fig. 16). 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.

Fig. 17 compares the 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 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.

3) Randomized Placements: As a performance baseline, we generated five randomized placements for each $k = 5, \ldots, 20$ as described in Section VIII-A. In the subsequent sections, we report localization error of the kth randomized placement as the average of 5 placements generated for every k, each tested on ten daily-routine traces.

C. Performance Evaluation

Having selected 15 optimized placements, 15 manual placements and averaging the results of five randomized placements for each $k = 5, \ldots, 20$, we compare the performance of the three placement strategies relative to each other. Fig. 18 depicts the average localization error achieved by each placement in

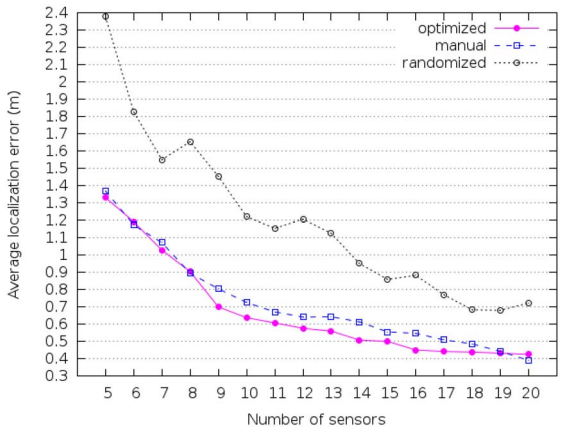

Fig. 18. Localization error of optimized, manual, and randomized placements tested on daily-routine traces.

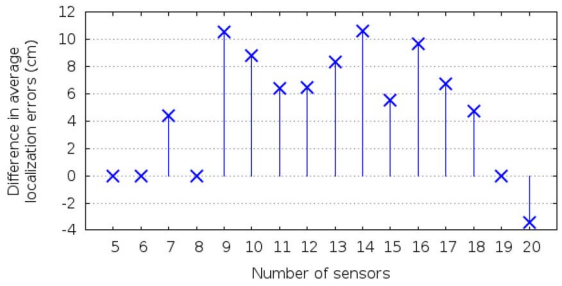

Fig. 19. Difference in average localization errors of the optimal and manual placements.

simulation experiments with ten 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.[3]

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, the manual and optimized placements simply emphasize this trend by monotonically improving localization accuracy with each additional sensor. We also observe that the performance of the optimized placements 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 the optimized placements are statistically better than the manual ones for all other k except $k = 20$, the only case when the manual placement statistically outperforms the optimized placement. The difference in average localization error values for all statistically significant cases is shown in Fig. 19.

Note the drastic change in results of the optimized placements between $k = 8$ and $k = 9$. One notable detail distinguishing the placement with nine sensors is that it is the first placement (in the order of ascending k) with a sensor in

[3]For full descriptive statistics please refer to Table 1 in the supplementary file on the journal website.

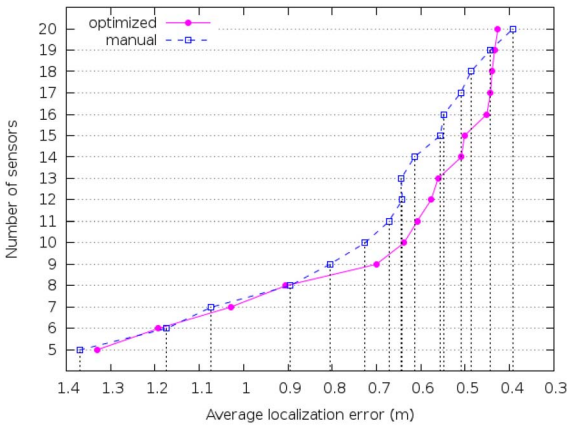

Fig. 20. Correspondence between the localization error and the number of sensors.

the washroom [sensor #9 in Fig. 14(a)]. Our sensor-placement algorithm deems that sensors placed in the washroom have a rather low utility gain, for two reasons: 1) the walkable area inside the washroom is very limited and 2) 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 9th 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. In principle, this shortcoming of our algorithm could be mitigated by incorporating expert knowledge about especially important areas into the coverage utility assignment by artificially inflating the coverage utility values of grid points in those areas.

Aside from optimizing localization accuracy under a cardinality constraint, we are also interested in the reverse problem, 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. Fig. 20 is plotted from the same data as Fig. 18 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 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 [6]), then the percentiles of error distribution become more consistent quality indicators than the

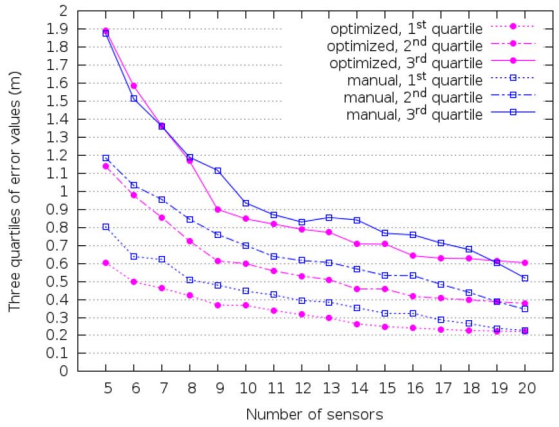

Fig. 21. Three quartiles of errors produced by optimized and manual placements.

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 run-time 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 Fig. 21. 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 for at least 50% of the system runtime.

Altogether, 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 deployments of the Smart-Condo system.

IX. Conclusion

The Smart-Condo project aims to support people with chronic conditions to live independently at home, longer. Our approach has emphasized a privacy-respecting and nonintrusive 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, which can be used for early diagnosis of chronic conditions or simply to better deliver intelligent services spatially distributed in the apartment. An accurate localization method is crucial to this project.

Localization accuracy greatly depends on the sensor placement, which is typically designed manually for each new deployment and whose quality for localization purposes depends on the designer's expertise. The work we reported in this paper investigated the problem of optimizing sensor placement for localization in 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 daily routine. We then proposed a greedy placement algorithm that generates near-optimal solutions for the formulated optimization problem, and evaluated the localization performance of the generated placements.

To empirically validate 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.

Our framework has the advantage of simplicity and yet readily accommodates several practical extensions. For instance, it is simple for a practitioner to constrain the optimization to exclude specific sensor placement locations for practical and/or aesthetic purposes (e.g., to account for existing ceiling infrastructure such as HVac, fire, and lighting equipment). It can also be used to produce "incremental placements," i.e., placing new sensors to complement an existing sensor array, all while retaining the same approximation bounds.

Avenues for future work include investigating the fault-tolerance of the optimized placements as well as testing them 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 the mobility patterns modeled within our framework prior to deployment with the occupant's true mobility data. In particular, we would like to study the robustness of our placements with respect to different sources of uncertainty in the 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 required 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.

Appendix

Theorem 1: Given N walkable grid points, a set of sensor candidates $\mathbb{S}$, where each element $s_j \in \mathbb{S}$ is defined through an

N-dimensional vector $\mathbf{s_j}$, an N-dimensional vector of coverage utility values $\mathbf{c}^*$, and a number k, to maximize the objective function $\Phi(\{s_1, \ldots, s_k\})$ is NP-hard.

Proof: We show that our problem is NP-hard through a polynomial-time reduction from the exact cover problem, known to be NP-complete: given a set $\mathbb{U}$ and a collection $\mathbb{A}$ of subsets of $\mathbb{U}$, determine whether such a subcollection $\mathbb{A}'$ of $\mathbb{A}$ exists that every element in $\mathbb{U}$ is contained in exactly one subset in $\mathbb{A}'$. That is, an exact cover is a collection of mutually disjoint subsets of $\mathbb{U}$ whose union is $\mathbb{U}$.

Throughout the reduction procedure, we will assume that the number of subsets in $\mathbb{A}'$ is known and equals k. If this is false, we can always try all possible choices, i.e., $k = 1, \ldots, |\mathbb{A}|$, increasing the running time by no more than a factor $|\mathbb{A}|$. Therefore, if we can answer this decision problem under the constraint that the subcollection is of size k in polynomial time, then we have a polynomial time solution to the exact cover problem as well.

We map the input of the exact cover problem to the input of our problem as follows. First, we associate set $\mathbb{U}$ with the collection of N walkable grid points, i.e., $u_i \in \mathbb{U}$ corresponds to the ith grid point. Second, the collection $\mathbb{A}$ of subsets of $\mathbb{U}$ becomes a collection of sensor candidates, where each subset $\mathbb{A}_j$ from $\mathbb{A}$ corresponds to the following sensor-coverage model of sensor s_j: if u_i is present in $\mathbb{A}_j$, then the ith grid point is covered by sensor s_j, i.e., $c_i^{\{s_j\}} = 1$ for all $u_i \in \mathbb{A}_j$, and $c_i^{\{s_j\}} = 0$ otherwise (where $c_i^{\{s_j\}}$ values are components of sensor vector $\mathbf{s_j}$). In other words, we want to determine whether such a collection of k sensors exists that covers all the grid points without overlapping. The last input parameter required for our problem is the coverage utility vector $\mathbf{c}^*$, and we assign every component of this vector c_i^* to be any fixed constant C: $0 < C < 1$, for all $u_i \in \mathbb{U}$, i.e., any positive value less than the lowest positive $c_i^{\{s_j\}}$ value (1 in our case).

Assume we have a polynomial time solver for our problem that returns collection $\mathbb{A}'$ of k subsets of $\mathbb{U}$. We can examine the solution given by this solver and verify whether these subsets are mutually disjoint, i.e., $\mathbb{A}_i \cap \mathbb{A}_j = \varnothing, \forall \mathbb{A}_i \neq \mathbb{A}_j$ in $\mathbb{A}'$, and count the total number of covered elements, i.e., $|\cup_{j=1}^{k} \mathbb{A}_j|$; both operations are performed in polynomial time. If the subsets are disjoint, and $|\cup_{j=1}^{k} \mathbb{A}_j| = |\mathbb{U}|$, then we return "yes" for the original exact cover problem, and "no" otherwise. To prove the correctness of this procedure, assume for purposes of contradiction that an exact cover does exist, but that our solver returned either (case 1) a disjoint collection that does not cover the entire space, or (case 2) a collection of nondisjoint subsets. Note these two cases are mutually exclusive and exhaustive.

Let us consider the first case, in which the union of returned disjoint subsets $\mathbb{A}_j$ does not entirely cover set $\mathbb{U}$, i.e., $|\cup_{j=1}^{k} \mathbb{A}_j| < |\mathbb{U}|$. Recall our objective function (11): we want to maximize the sum of utility scores of k sensors placed on the grid while imposing a penalty for overlapping sensors. According to (7), the utility score of arbitrary sensor s_j that does not overlap with any other sensors, i.e., $\mathbf{s_j} \cdot \mathbf{s_m} = 0$ for all $j \neq m$, equals $\mathbf{s_j} \cdot \mathbf{c}^* = \sum_{i=1}^{N} c_i^{\{s_j\}} c_i^*$. The objective value for $s_1, \ldots, s_k$ nonoverlapping sensors represented by $\mathbb{A}_1, \ldots, \mathbb{A}_k$ disjoint subsets is therefore

$$\Phi(\{s_1, \ldots, s_k\}) = \sum_{j=1}^{k} \sum_{i=1}^{N} c_i^{\{s_j\}} c_i^* \tag{12}$$

$$= C \sum_{j=1}^{k} \sum_{i=1}^{N} c_i^{\{s_j\}} \tag{13}$$

$$= C|\cup_{j=1}^{k} \mathbb{A}_j| < C|\mathbb{U}|. \tag{14}$$

Equation (12) sums up the utility scores of k nonoverlapping sensors; c_i^* values are replaced with the constant C and taken out of the summation on (13); since $c_i^{\{s_j\}} = 1$ for every element $u_i \in \mathbb{A}_j$, and no u_i belongs to two distinct subsets simultaneously, the summation of $c_i^{\{s_j\}}$ values across all k sensors is equivalent to the number of elements in the union of subsets $\mathbb{A}_j$ on (14). Note, however, that since the exact cover exists, and it also consists of mutually disjoint subsets, our objective function for such a cover would be equal $C|\mathbb{U}|$, which is greater than the objective value for the subsets $\mathbb{A}_j$. This contradicts the assumed optimality of our hypothetical solver, altogether proving that this procedure will not return a partial cover if a full one exists.

Let us address the second case, i.e., the subsets returned by the solver are not disjoint. We consider two pairs of arbitrary sets $\mathbb{A}_1$, $\mathbb{A}_2$, and $\mathbb{B}_1$, $\mathbb{B}_2$ (with corresponding sensors a_1, a_2, and b_1, b_2) such that $\mathbb{A}_1 \cap \mathbb{A}_2 \neq \varnothing$, $\mathbb{B}_1 \cap \mathbb{B}_2 = \varnothing$, and $\mathbb{A}_1 \cup \mathbb{A}_2 \subseteq \mathbb{B}_1 \cup \mathbb{B}_2$. Let us calculate values of our objective function for these two pairs of sets separately. According to the derivation for disjoint sets (14), $\Phi(\{b_1, b_2\}) = C|\mathbb{B}_1 \cup \mathbb{B}_2|$. The objective value for overlapping sensors, however, includes a penalty, which leads to the following inequality:

$$\Phi(\{a_1, a_2\}) = \mathbf{a_1} \cdot \mathbf{c}^* + \mathbf{a_2} \cdot \mathbf{c}^* - \mathbf{a_1} \cdot \mathbf{a_2} \tag{15}$$

$$= \sum_{i=1}^{N} c_i^{\{a_1\}} c_i^* + \sum_{i=1}^{N} c_i^{\{a_2\}} c_i^* - \sum_{i=1}^{N} c_i^{\{a_2\}} c_i^{\{a_2\}} \tag{16}$$

$$= C\left(\sum_{i=1}^{N} c_i^{\{a_1\}} + \sum_{i=1}^{N} c_i^{\{a_2\}} \right) - \sum_{i=1}^{N} c_i^{\{a_2\}} c_i^{\{a_2\}} \tag{17}$$

$$= C(|\mathbb{A}_1 \cup \mathbb{A}_2| + |\mathbb{A}_1 \cap \mathbb{A}_2|) - |\mathbb{A}_1 \cap \mathbb{A}_2| \tag{18}$$

$$= C|\mathbb{A}_1 \cup \mathbb{A}_2| + (C-1)|\mathbb{A}_1 \cap \mathbb{A}_2| \tag{19}$$

$$< C|\mathbb{B}_1 \cup \mathbb{B}_2| = \Phi(\{b_1, b_2\}). \tag{20}$$

Equation (15) is composed by plugging the utility score expression (7) into the objective function (11). We replace vector terms with corresponding scalar values on (16). The sums in the first term on (17) are essentially a sum of the number of elements in sets $\mathbb{A}_1$ and $\mathbb{A}_2$, respectively. To express this number through the union of two sets, we should also account for the elements that belong to both sets and get double-counted in the original summation. We do so by adding up the number of elements in the union and the number of elements in the intersection of the two sets in the first term on (18). The strict inequality on (20) holds due to the following.

1) The assumption that $\mathbb{A}_1 \cup \mathbb{A}_2 \subseteq \mathbb{B}_1 \cup \mathbb{B}_2$, which is equivalent to $|\mathbb{A}_1 \cup \mathbb{A}_2| \leq |\mathbb{B}_1 \cup \mathbb{B}_2|$.
2) The assumption that $\mathbb{A}_1 \cap \mathbb{A}_2 \neq \varnothing$.

3) The assigned value $C < 1$, so that $(C-1)|\mathbb{A}_1 \cap \mathbb{A}_2|$ is strictly negative.

Essentially, the objective value for a number of disjoint subsets is strictly greater than its value for a number of intersecting subsets that cover the same or fewer elements, given a particular assignment of c_i^* and $c_i^{\{s_j\}}$ values. Although we showed this only for pairs of subsets, this property holds for any number of nondisjoint subsets since the "overlap" penalty is accumulative. Since we know that the exact cover $\mathbb{A}'$ of set $\mathbb{U}$ exists, the objective value for $\mathbb{A}'$ is strictly greater than that of any other collection of nondisjoint subsets, which contradicts the assumed optimality of our hypothetical solver.

We proved that if the exact cover of set $\mathbb{U}$ exists, then our problem solver will always find it. Since there exists a polynomial-time reduction of the exact cover problem to an instance of our problem, our problem is also NP-hard. ■

Theorem 2: The objective function (11) is submodular.

Proof: Submodularity is formalized for a set function F, defined on subsets of $\mathbb{V}$, as follows:

$$F(\mathbb{A} \cup \{s\}) - F(\mathbb{A}) \geq F(\mathbb{B} \cup \{s\}) - F(\mathbb{B}) \tag{21}$$

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

$$\begin{aligned}&\Phi(\{s_1, \ldots, s_{m-1}, s'\}) - \Phi(\{s_1, \ldots, s_{m-1}\}) \\ &\quad \geq \Phi(\{s_1, \ldots, s_{m-1}, s_m, s'\}) - \Phi(\{s_1, \ldots, s_{m-1}, s_m\})\end{aligned} \tag{22}$$

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 (11) into the left and right sides of (22), respectively, we obtain

$$\Phi(\{s_1, \ldots, s_{m-1}, s'\}) - \Phi(\{s_1, \ldots, s_{m-1}\}) \tag{23}$$

$$= \left(\sum_{i=1}^{m-1} \delta_{s_i}^{\mathbb{P}_{i-1}} + \delta_{s'}^{\{s_1, \ldots, s_{m-1}\}} \right) - \sum_{i=1}^{m-1} \delta_{s_i}^{\mathbb{P}_{i-1}} \tag{24}$$

$$= \delta_{s'}^{\{s_1, \ldots, s_{m-1}\}} \tag{25}$$

$$\Phi(\{s_1, \ldots, s_m, s'\}) - \Phi(\{s_1, \ldots, s_m\}) \tag{26}$$

$$= \left(\sum_{i=1}^{m} \delta_{s_i}^{\mathbb{P}_{i-1}} + \delta_{s'}^{\{s_1, \ldots, s_m\}} \right) - \sum_{i=1}^{m} \delta_{s_i}^{\mathbb{P}_{i-1}} \tag{27}$$

$$= \delta_{s'}^{\{s_1, \ldots, s_m\}}. \tag{28}$$

Plugging results (25) and (28) into inequality (22) is equivalent to the following:

$$\delta_{s'}^{\{s_1, \ldots, s_m\}} \leq \delta_{s'}^{\{s_1, \ldots, s_{m-1}\}}. \tag{29}$$

To prove this inequality, we transform the left term using the utility score function definition (8) (omitting positive thresholding for clarity)

$$\delta_{s'}^{\{s_1, \ldots, s_m\}} = \mathbf{s}' \cdot \left(\mathbf{c}^* - \mathbf{c}^{\{s_1, \ldots, s_m\}} \right) \tag{30}$$

$$= \mathbf{s}' \cdot \left(\mathbf{c}^* - \sum_{1 \leq j \leq m} \mathbf{s_j} \right) \tag{31}$$

$$= \mathbf{s}' \cdot \left(\mathbf{c}^* - \sum_{1 \leq j \leq m-1} \mathbf{s_j} - \mathbf{s_m} \right) \tag{32}$$

$$= \mathbf{s}' \cdot \left(\mathbf{c}^* - \mathbf{c}^{\{s_1, \ldots, s_{m-1}\}} \right) - \mathbf{s}' \cdot \mathbf{s_m} \tag{33}$$

$$= \delta_{s'}^{\{s_1, \ldots, s_{m-1}\}} - \mathbf{s}' \cdot \mathbf{s_m}. \tag{34}$$

Hence, the inequality (29) holds as long as $\mathbf{s}' \cdot \mathbf{s_m} \geq 0$. Since an arbitrary sensor vector (individual sensor-coverage model) $\mathbf{s_j}$ is specified using $c_i^{\{s_j\}} \geq 0$, a dot product of any two vectors is also always nonnegative. 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. ■

Theorem 3: The objective function (11) is monotonic.

Proof: 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 (25) 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 nonnegative in (8), thus proving the monotonicity of our objective (11). ■

ACKNOWLEDGMENT

The authors would like to thank M. Vosoughpour Yazdchi for developing simulation software used for experimental evaluation of this paper and they also thank C. Rayner whose expertise and valuable advice were crucial for this paper. They would also like to thank L. Liu, S. King, and R. Lederer for their role in the broader Smart-Condo project.

REFERENCES

[1] 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.

[2] D. Cook, "Health monitoring and assistance to support aging in place," *J. Univ. Comput. Sci.*, vol. 12, no. 1, pp. 15–29, 2006.

[3] E. Stroulia *et al.*, "Software engineering for health education and care delivery systems: The smart condo project," in *Proc. ICSE Workshop Softw. Eng. Health Care (SEHC)*, Vancouver, BC, Canada, 2009, pp. 20–28.

[4] C. Scanaill *et al.*, "A review of approaches to mobility telemonitoring of the elderly in their living environment," *Ann. Biomed. Eng.*, vol. 34, no. 4, pp. 547–563, 2006.

[5] V. Ganev, D. Chodos, I. Nikolaidis, and E. Stroulia, "The smart condo: Integrating sensor networks and virtual worlds," in *Proc. 2nd Workshop Softw. Eng. Sens. Netw. Appl.*, 2011, pp. 49–54.

[6] N. Boers *et al.*, "The smart condo: Visualizing independent living environments in a virtual world," in *Proc. 3rd Int. Conf. Pervasive Comput. Technol. Healthcare (PervasiveHealth)*, Apr. 2009, pp. 1–8.

[7] K. Woo *et al.*, "Sensors as an evaluative tool for independent living," in *Advances in Human Aspects of Healthcare*, V. G. Duffy, Ed. Boca Raton, FL, USA: CRC Press, 2012, pp. 612–621.

[8] I. Vlasenko, M. V. Yazdchi, V. Ganev, I. Nikolaidis, and E. Stroulia, "The Smart-Condo infrastructure and experience," in *Evaluating AAL Systems Through Competitive Benchmarking* (Communications in Computer and Information Science), vol. 362, S. Chessa and S. Knauth, Eds. Berlin, Germany: Springer, 2013, pp. 63–82.

[9] 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.

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

[11] T. Yi and H. Li, "Methodology developments in sensor placement for health monitoring of civil infrastructures," *Int. J. Distrib. Sensor Netw.*, vol. 2012, 2012, Art ID. 612726.
[12] S. Toumpis and G. A. Gupta, "Optimal placement of nodes in large sensor networks under a general physical layer model," in *Proc. IEEE Commun. Soc. Conf. Sensor Ad Hoc Commun. (SECON)*, vol. 4. 2005, pp. 275–283.
[13] K. Chakrabarty, S. Iyengar, H. Qi, and E. Cho, "Grid coverage for surveillance and target location in distributed sensor networks," *IEEE Trans. Comput.*, vol. 51, no. 12, pp. 1448–1453, Dec. 2002.
[14] Q. Wu, N. S. Rao, X. Du, S. S. Iyengar, and V. K. Vaishnavi, "On efficient deployment of sensors on planar grid," *Comput. Commun.*, vol. 30, nos. 14–15, pp. 2721–2734, 2007.
[15] A. Krause, C. Guestrin, A. Gupta, and J. Kleinberg, "Near-optimal sensor placements: Maximizing information while minimizing communication cost," in *Proc. 5th Int. Conf. Inf. Process. Sensor Netw.*, 2006, pp. 2–10.
[16] S. Dhillon and K. Chakrabarty, "Sensor placement for effective coverage and surveillance in distributed sensor networks," in *Proc. Wirel. Commun. Netw.*, vol. 3. New Orleans, LA, USA, Mar. 2003, pp. 1609–1614.
[17] P. David, V. Idasiak, and F. Kratz, "A sensor placement approach for the monitoring of indoor scenes," in *Proc. 2nd Eur. Conf. Smart Sens. Context*, 2007, pp. 110–125.
[18] Q. Wang, K. Xu, G. Takahara, and H. Hassanein, "WSN04-1: Deployment for information oriented sensing coverage in wireless sensor networks," in *Proc. IEEE Glob. Telecommun. Conf. (GLOBECOM)*, 2006, pp. 1–5.
[19] A. Roy, S. Das, and K. Basu, "A predictive framework for location-aware resource management in smart homes," *IEEE Trans. Mobile Comput.*, vol. 6, no. 11, pp. 1270–1283, Nov. 2007.
[20] P. Hart, N. Nilsson, and B. Raphael, "A formal basis for the heuristic determination of minimum cost paths," *IEEE Trans. Syst. Sci. Cybern.*, vol. 4, no. 2, pp. 100–107, Jul. 1968.
[21] (2014, Sep.). *MP Motion Sensor NaPIOn, Datasheet. Panasonic Electric Works* [Online]. Available: http://www3.panasonic.biz/ac/e/control/sensor/human/napion/index.jsp
[22] S. Lee, K. Ha, and K. Lee, "A pyroelectric infrared sensor-based indoor location-aware system for the smart home," *IEEE Trans. Consum. Electron.*, vol. 52, no. 4, pp. 1311–1317, Nov. 2006.
[23] X. Luo, B. Shen, X. Guo, G. Luo, and G. Wang, "Human tracking using ceiling pyroelectric infrared sensors," in *Proc. IEEE Int. Conf. Control Autom. (ICCA)*, Christchurch, New Zealand, 2009, pp. 1716–1721.
[24] B. Song, H. Choi, and H. Lee, "Surveillance tracking system using passive infrared motion sensors in wireless sensor network," in *Proc. Int. Conf. Inf. Netw. (ICOIN)*, 2008, pp. 1–5.
[25] G. L. Nemhauser, L. A. Wolsey, and M. L. Fisher, "An analysis of approximations for maximizing submodular set functions," *Math. Program.*, vol. 14, no. 1, pp. 265–294, 1978.
[26] X. Xu, Y. Lu, S. Huang, Y. Xiao, and W. Wang, "Incremental sensor placement optimization on water network," in *Machine Learning and Knowledge Discovery in Databases* (Lecture Notes in Computer Science), vol. 8190, H. Blockeel, K. Kersting, S. Nijssen, and F. Železný, Eds. Berlin, Germany: Springer, 2013, pp. 467–482.

Iuliia Vlasenko received the bachelor's and Specialist degrees in radio engineering and communication from the Dnipropetrovsk National University, Dnipropetrovsk, Ukraine, and the M.Sc. degree in computing science from the University of Alberta, Edmonton, AB, Canada, in 2007, 2009 and 2013, respectively.

She was a Research and Teaching Assistant, while pursuing the M.Sc. degree. She is currently a Software Developer with Pleasant Solutions, Edmonton, AB, Canada. Her current research interests include intersection between wireless sensor networks and ambient assisted living, which led to her involvement with the Smart-Condo Project. As part of this research project, she was able to participate in two real-world deployments, publish her work, and complete a master's thesis.

Ms. Vlasenko was the recipient of the President of Ukraine Scholarship awarded for Outstanding Academic Achievement during her undergraduate studies in 2006 and the Graduate Student Teaching Award in Recognition of Excellence in the performance of teaching assistant duties with the University of Alberta, Edmonton, AB, Canada, in 2012.

Ioanis Nikolaidis (M'93) received the B.Sc. degree from the University of Patras, Patras, Greece, and the M.Sc. and the Ph.D. degrees from Georgia Tech, Atlanta, GA, USA, in 1989, 1991 and 1994, respectively.

He joined European Computer-Industry Research Center GmbH, Munich, Germany, as a Research Scientist from 1994 to 1996, and joined the University of Alberta, Edmonton, AB, Canada, as an Assistant Professor, in 1997, where is currently a Full Professor and holds an Adjunct Professor appointment with the Department of Occupational Therapy. He has supervised, or is currently supervising, a total of 13 Ph.D. and 11 M.Sc. students. His current research interests include computer network protocol modeling and simulation, network protocol performance, and wireless sensor network architectures and applications. He has published over 100 papers in refereed journals and conferences, four book chapters, and recently co-edited with Dr. K. Iniewski a book entitled *Building Sensor Networks: From Design to Applications*.

Prof. Nikolaidis was the co-recipient of the Best Paper Award of Conference on Communication Networks and Services Research (CNSR) 2011, SMARTGREENS 2014, and the University of Alberta Teaching Unit Award as part of the Smart-Condo teaching team, and the recipient of the Best Paper Presentation of ADHOC-NOW 2012 and the Best Paper Award of SMARTGREENS 2014. He was an Area Editor for *Computer Networks*, (Elsevier, 2000–2010), and one of the most long-standing Editors from 1999 to 2013, and an Editor-in-Chief from 2007 to 2009 for the IEEE NETWORK MAGAZINE. He has co-chaired CNSR 2011, International Conference On Ad Hoc Networks and Wireless (ADHOC-NOW) 2004 and 2010. He serves as a Steering Committee Member of IEEE International Workshop on Wireless Local Networks (annual workshop co-located with IEEE Conference on Local Computer Networks) and of ADHOC-NOW. He served as an National Science Foundation Panel Member in 2010 and, since 2007, he belongs to the MITACS College of Reviewers. He has served as Technical Program Committee Member and reviewer for numerous conferences and journals, as well as for the following funding agencies: Natural Sciences and Engineering Research Council, NSF, Ontario Centres of Excellence, FWF/START in Austria, NTU/IntelliSys in Singapore, and NWO in Holland. He is a Lifetime Member of ACM.

Eleni Stroulia received the B.Sc. degree from the University of Patras, Patras, Greece, and the M.Sc. and Ph.D. degrees from Georgia Tech, Atlanta, GA, USA, in 1989, 1991 and 1994, respectively.

She is a Professor and NSERC/AITF Industrial Research Chair on Service Systems Management (with support from IBM) with the Department of Computing Science, University of Alberta, Edmonton, AB, Canada. Her current research interests include industrial and societal needs, with software systems that support people activities and improve the quality of their outcomes. Her serving has made substantial contributions to the areas of software analysis, migration, and reengineering. More recently, she has been focusing on software services through web platforms (web 2.0 collaborative tools, mobile devices and virtual worlds) as a way to offer innovative healthcare and education services. She has supervised over 40 graduate students and many undergraduate and high-school students. She is the Co-Leader of the Smart-Condo Project, whose objective is to use technology as a means to support people with chronic conditions to live independently longer and to educate health-science students (with serious games and simulation-based training) in using technology to provide better care.

Prof. Stroulia has served as a reviewer of many top-tier journals, as Co-Chair and PC Member on many international conference committees, and as member of several grant-adjudication panels.