Framework-supported mechanism of testing algorithms for assessing memory and detecting disorientation from IoT sensors

Iván García-Magariño
Dept. of Software Engineering and Artificial Intelligence
Complutense University of Madrid
Madrid, Spain
igarciam@ucm.es

Marlón Cárdenas
Dept. of Software Engineering and Artificial Intelligence
Complutense University of Madrid
Madrid, Spain
marlonca@ucm.es

Jorge Gómez-Sanz
Dept. of Software Engineering and Artificial Intelligence
Complutense University of Madrid
Madrid, Spain
jjgomez@ucm.es

Juan L. Pérez Díez
Dept. of Software Engineering and Artificial Intelligence
Complutense University of Madrid
Madrid, Spain
jlpdiez@ucm.es

***Abstract*—Assessing memory in daily life can be useful for both early detection and tracking the symptoms of neurodegenerative disease such as Alzheimer's disease. One of these symptoms is disorientation, which can cause domestic accidents. With the avenue of Internet of Things (IoT), houses can collect big data in real-time for being processed and acquiring knowledge between both edge computing (local processing for avoiding communication overload) and the cloud computing conforming the fog computing. What sensors are needed and how to collect the information is something that requires actually preparing prototypes of this setup in a lab. As an alternative, this work proposes a cheaper method which is based on the use of realistic 3D simulations of the scenarios to consider. Using these scenarios, different algorithms for assessing memory can be compared and evaluate how well they detect disorientation of a person living in a house by means of tracking the movement of a person. The contribution of this paper is a case study where researchers can design the scenarios, including the activity of the daily living of the individuals, the array of sensors to be deployed, and the disorientation detection algorithms.**

***Index Terms*—ambient-assisted living, IoT, memory assessment, multi-agent system, virtual living lab**

I. Introduction

Internet of Things (IoT) provides a wide range of possibilities for improving and automating some tasks in healthcare domain. IoT can both (a) improve the remote assistance of doctors and (b) automate the collection of data from patients in their daily lives. In the latter, IoT is conformed of common objects present in daily lives that can have sensors for collecting data and send these data through Internet. Examples of smart objects are toothbrushes, doors, cupboards, beds, hospital rooms, houses. These smart objects are usually equipped with sensors, a simple single-board computer and a connection to Internet. Normally, some of the processing is done locally to avoid transmitting excessive amounts of data, adhering to edge-computing field.

In the field of neurodegenerative disorders, regularly measuring memory can be crucial for early detection of diseases such as Alzheimer Disease. However, proper assessment of memory usually requires to perform some tests consuming time from people such as the common test about face-occupation pairs [1]. Although some frameworks such as FAMAP (a Framework for Developing m-Health Apps) [2] allows automating this kinds of task from a tablet device, these tasks still require the people to install this app and use the app for getting some preliminary results. In addition, the detection of disorientation can be useful for avoiding domestic accidents, as well as being some meaningful indicator related with some diseases such as epilepsy [3].

Navigation patterns have proven to be related with memory given the activity in hippocampal neurons while primates including humans observe a scene and navigate through it [4]. These memory activity can be associated with one-trial object or place recall task. Even some works like [4] associated the memory activity with navigation, those works omits how to measure memory from navigation.

The traditional design of this kind of aid is usually done with the involvement of end-users, in this case people with a neurodegenerative disorder or experts in health, using a co-creation approach in a living lab [5]. Nevertheless, it is an

We acknowledge the support of the projects "Collaborative Ambient Assisted living Design" (TIN2014-57028-R), "Diseño colaborativo para la promoción del bienestar en ciudades inteligentes inclusivas" (TIN2017-88327-R), and "Red Temática de Investigación en Ciudades Inteligentes" (TIN2016-81766-REDT) funded by the Spanish council of Science, Innovation and Universities from the Spanish Government. Also from project MOSI-AGIL-CM (S2013/ICE-3019) funded by the Comunidad de Madrid and European Regional Development Funds.

978-1-5386-4980-0/19/$31.00 ©2019 IEEE

expensive process and subject of strong ethical concerns.

To reduce this cost, the work addresses this problem using a virtual living lab approach [6], [7] where the situations the IoT system will find can be reproduced in 3D simulations. These simulations can be connected to algorithms and their performance can be evaluated in advance. Though such systems cannot replace the real world testing, yet, they can be used to polish the ideas and prepare a more convincing solution.

The contribution of this work is an evaluation of the effectiveness of this kind of virtual living lab approach to the design of a smart house that is capable of measuring episodic memory or detecting disorientation. In particular, the illustration of this approach mainly explores the presence sensors, and focuses on navigation patterns. IoT devices of smart homes can provide information of the user in order to measure their memory based on their behavior. In fact, many smart objects could contribute to this measurement, including storage smart objects, presence sensors, usage of some kitchen or bathroom equipment. The underlying idea of all these possibilities is based on checking whether the user is repeatedly doing something because they forgot something like the location of any item or having done some daily task and repeatedly doing it again. Similar patterns could also reflect disorientation.

II. Related work

The most related work could be classified in the categories of assisting living, disorientation detection, and measurement of health indicators.

One of the most common uses of smart homes is to assist cognitively-impaired peopled in their daily lives. For example, [8] proposed an approach of using sensors near to appliances to assist users in using them. Their system detected erratic behaviors normally related with cognitive deficits for giving them cues for helping users to complete their daily activity. Their system avoided cameras, and used movement sensors and radio-frequency identification sensors, for avoiding the use of sensors. In this line of research, [9] also proposed to combine collecting information from sensors of smart homes and apply artificial intelligence for detecting some problems and assist in elder people. In particular, they exemplified their approach with the emergency situation of a fall in a bedroom, and how their approach applied fall detection in this case. Nevertheless, these works did not measure the memory of users for tracking their development of symptoms of some common diseases such as Alzheimer Disease (AD).

Some works have analyzed navigation paths for some purposes. For instance, [10] focused on elder with cognitive impairments that suffered disorientation in unfamiliar or even familiar environments. Their proposed a detection method that analyzed the trajectories of individuals revealed by GPS for warning its users when suffering disorientation in real-time. Basically, it detected outlying trajectories achieving appropriate results in most cases. Their work is similar to the current one in goal, since both works pursue to notify users when suffering from memory losses. However, the environments are different, since the current work focus in indoor scenarios, and the nature of memory loss is different, i.e. losing some item instead of getting lost.

Other works focus on using information collected in daily lives of users as evidence of some impairments. For example, [11] used the information of wearable sensors for collecting information that could be used as medical proof of some symptoms. They analyzed some cases in which this information could be used for tracking symptoms of diseases such as dementia, focusing more in the regulatory aspects rather than in the technological challenges. In addition, [12] used the collection of activity-aware smart home information for detecting functional health decline, which is one of the most common consequences of aging. Their approach applied regression models to estimate some standardized functional health scores, based on the training with data from older adults considering information of more than two years. Their prediction models could detect the fluctuations in everyday tasks. Nonetheless, these works did not consider the navigation paths as a valuable information to assess memory of users.

Therefore, the current work belongs to the category of ambient assisted-living (AAL) systems, as the current approach is intended to finally be integrated in one of these systems as another layer of data analysis of the information collected by smart home sensors. It is related with the analysis of navigation paths and disorientation, as one could say that the system analysis the trajectories of going through places with storage. Finally, this work can also be considered as another mechanism for tracking symptoms of diseases (in this case the ones with memory-impairment like AD). However, this work is novel in comparison to all these, as it proposes a new mechanism of measuring memory different to all the aforementioned approaches.

To analyze the performance of the algorithms, a testing ground is needed. This testing ground can be a real facility or a simulated environment. The literature contains several examples of simulation platforms where one can reproduce a physical location and allocate sensors that generate similar output as real ones.

Ubiwise [13] is a proof of concept development where ubiquitous computing can be simulated. Ubiwise uses simulated devices, virtual representations of the physical objects. There are no simulated characters, but it is the user who walks through the scene through an avatar. The platform used in this paper has built-in sensors and devices too. They are hard-coded, but new ones can be programmed and associated to the simulation. Also, the platform used in this work allows to model the expected behavior of characters, even when interacting with devices.

A more modern variant of Ubiwise is Ubiksim [14]. Ubiksim is more advanced than Ubiwise in realism, it has better scene rendering engines, and integration with external devices. Also, Ubiksim includes simulation of characters whose behavior is programmed as hierarchical state machines. Ubiksim is not used in this paper because, despite advances, Ubiksim works in 2D. It bases on the sweet home 3D rendering engine and scenes appear to be 3D, but character movement and object manipulation is really in 2D. The platform used in this work works in 3D and allows to perform typical operations in such environments, such as ray picking to identify objects in the scene.

GLS [15] is a system for the visualization and testing of location-aware event-driven middleware and applications. The simulation reproduces in a 2D world the feedback given by sensors, so that the application can be connected to them and then tested within the simulation. Compared with GLS, the selected platform gives the advantage of more realism through the use of 3D while retaining the capability of simulating sensor output. Besides, characters can be added with their own behavior and they can interact with the applications.

3dSim [16] is a system for rapid prototyping ambient intelligence applications. It uses a 3D representation for creating entities in the environment and connecting some of them with the real world. Initially conceived to define meeting rooms and give orders to connected elements. Compared with 3DSim, the platform chosen for this experiment has the advantage of allowing to define the behavior of the participants.

The paper uses the AIDE platform [7] that, besides reproducing situations in 3D, it allows the developer to graphically describe the behavior of simulated characters and allocate certain kind of sensors typical in this kind of problems, such as presence sensors or accelerometers. We call the result a **virtual living lab**, because it aims to give the developer feedback on how the interaction with simulate users works.

III. Mechanism for assessing algorithms that measure memory or detect disorientation in navigation from IoT devices

This work proposes to assess algorithms for measuring memory or detecting disorientation through a virtual lab. The data collected from sensors could be transferred to visualize the movements of a person in a virtual lab. Then, the virtual lab can simulate different kits of sensors. These sensors output data through Internet sockets simulating IoT real devices. Then, researchers can test algorithms that interpret these IoT data. This article mainly focus on the definition of these algorithms, rather than the transfer of real data to virtual lab, which is an open challenge that we have already started addressing in other works such [6].

In this proposed mechanism, the researcher can either start from real data transferred to a virtual lab or some theoretical behavior designed with a model in the Ambient Intelligent Development Environment (AIDE) environment and simulated with its underlying model-driven development (MDD) approach. Another step is the definition of the houses and the type and location of IoT devices, which can be partially done with a MDD approach.

Since the definition of IoT systems usually rely on the quality of data, the selection of sensor types and their installation locations should be done with a reasonable strategies with some clear goals. Then, different algorithms can be defined to assess memory or detect disorientation. In this way, researcher can test different algorithms.

In order to illustrate this approach, we focus on the assessment of memory based on the navigation patterns detected with presence sensors. The proposed solution for measuring memory is intended to be generic enough, so it can adapt to most house distributions. In particular, since we have decided to check the navigation patterns, we also propose a mechanism to set the presence sensors, so that the collected data have quality, following the principles of the data-driven approach.

This case proposes to firstly place the sensors in the locations where the user usually stands for accessing to places where the user can leave or store some items. Among others, examples of these places are (a) all kinds of tables like dining-room, kitchen, bed-side and sofa-side tables, (b) all types of pieces of furniture that can store or hold items such as kitchen cupboards, bedroom closets and dining-room or entrance wardrobes.

All the sensor locations are represented relative to certain referred common point, which is normally the smart home central processing unit. If the user can afford, the room doors could also have presence sensors. In the Phat virtual living labs, different configuration of sensors could be tested. Notice that an user looking for something may just reach the door and take a look inside. In this way, the system can notice every time the user checks a piece of furniture and every time the user moves from one room to another.

In order to analyze the output of presence sensors, the proposed algorithm pre-processes the data by filtering only the information of the intended sensor type (in this case, presence sensors) and conforming the following sequence of data:
$s_1, \Delta t_1, \Delta t_{s,1}$
$s_2, \Delta t_2, \Delta t_{s,2}$
...
$s_N, \Delta t_N, \Delta t_{s,N}$
where s_i is the sensor that sensed the user in the position i of the sequence, where s_i belongs to the **S** set of the available presence sensors of the smart home, Δt_i is time elapsed from the previous sensor notification in the global sequence of sensors, and $\Delta t_{s,i}$ denotes the time elapsed from the last time that the same sensor notified a entering notification. Notice that following the edge computing approach, each sensor only reports through Internet when the user was outside the presence area and entered in this area. Thus, when the user stays in a position, no new notification is reported. The $\Delta t_{s,i}$ values could be calculated by either by the sensor, or the smart home system. In our simulations, this value is calculated by the smart home system.

The current approach is aimed at detecting the patterns in which the user recursively returns to the same storage place. We have considered two features as relevant for detecting these patterns:

- The number of sensors that reported the user presence in a give time window.
- The duration elapsed since the user was accessing the same storage access.

Regarding the first pattern, if a user is checking many storage points very shortly, then they probably is looking for something. For example, if a user checks all the storage points of one bedroom in less than 20 seconds, they are probably looking for something missing, rather than retrieving clothes and putting them on, as for the latter, they would probably need more time and would probably need to check less storage points.

In order to detect this pattern, we could set some threshold time $t_{p,1}$ of this first pattern and a number of sensors $n_{p,1}$, so that for each new item, it measures all the times of the last $n_{p,1}$ sensors and check whether is above the limit. In order to do this check efficiently, we perform the algorithm 2, implemented in a listener of the system.

Algorithm 1 Algorithm for measuring memory with presence sensors with pattern P_1

procedure INITIALIZEMEMORYTRACKINGSYSTEM( )
    sum $\leftarrow$ 0
    queue $\leftarrow$ new CircularQueue()
procedure HANDLESENSORINFORMATION($s_i, \Delta t_i, \Delta t_{s,i}$)
    queue.add (Δt_i)
    sum $\leftarrow$ sum + Δt_i
    if queue.length $> n_{p,1}$ **then**
        oldT $\leftarrow$ queue.begin
        queue.removeBegin()
        sum $\leftarrow$ sum - oldT
        if (sum $< t_{p,1}$) **then**
            notifyMemoryPattern (p_1)

Concerning the second pattern, our approach is based on the assumption that if the user comes back to the same storage point very shortly is probably because they are rechecking the location of some item.

This pattern is based on the time for returning to some specific storage place. In particular, it measures the specific times for returning to each specific sensor. It notifies the user about memory losses if the median returning time for some specific sensor is below certain threshold, using the thresholds $t_{p,2}$ for the median time of this second pattern (p_2) and the threshold $n_{p,2}$ for the minimum number of processed sensors for performing the analysis. Algorithm 2 represents this pattern as another listener of the same tracking system and consequently implementing the same methods. Notice that in this algorithm, s_i represents the sensor identifier. In this way, the lists of returning times of each sensor can be indexed by it. It is worth noting that we used the median instead of the average to avoid counting very large intervals of returning times to the same storage place, such as when the user is sleeping.

Algorithm 2 Algorithm for measuring memory with presence sensors with pattern P_1

procedure INITIALIZEMEMORYTRACKINGSYSTEM( )
    returningTimes $gets$ new List[$|S|$]
    for i $\in$ [0, $|S|$-1] **do**
        returningTimes[i] $\leftarrow$ new List();
procedure HANDLESENSORINFORMATION($s_i, \Delta t_i, \Delta t_{s,i}$)
    returningTimes[s_i].add ($\Delta t_{s,i}$)
    if (returningTimes[s_i].length $> n_{p,2}$) and
(returningTimes[s_i].median $< t_{p,2}$) **then**
        notifyMemoryPattern (p_2)

IV. SIMULATION IN THE PHAT 3D ENVIRONMENT

Since the experimentation with real smart homes would require high expenses, the proposed approach uses a living lab, which simulates common activities based on either interviews with real users, real data from sensors or theoretical behavior models for initial prototyping. This living-lab can simulate the output of these devices over the network ports. In this way, we can simulate programs of smart homes for measuring memory over simulated outputs. More concretely, we have decided to use the Phat framework for conforming the visual virtual lab. Figure 1 shows an example of the virtual lab in 3D with built with this framework. In particular, the navigation is shown with arrows on the floor.

Fig. 1. An example of execution of navigation with Phat environment

This lab has presence sensors that output the data over a network among other sensors. Figure 2 shows an example of information listened from the simulation environment by listening to the localhost through the 60000 port by means of the Putty program. This simulation also contained many other sensor information such as sensors with accelerometers in the hands of the avatar simulating a user of the smart home.

In Phat framework, the avatar can represent the opening of doors, as one can observe an example of an avatar opening a fridge door in Figure 3. In addition, Figure **??** shows an example of how modeling the opening/closing door action with a model in AIDE.

In this example, we have used the AIDE visual modeling language to express simple recurrent behaviors for the initial testing in figure 4. The whole specification is longer and one can either program the actions with actual Java code or use the visual modeling language. The character of the example from time to time gets disoriented. This can be expressed with

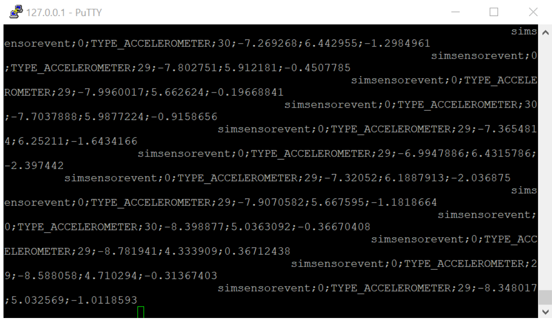

Fig. 2. Example of information received through network

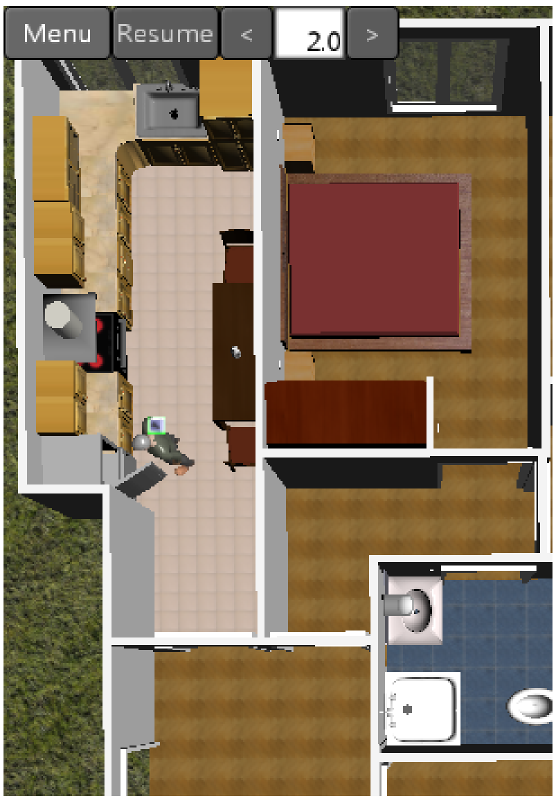

Fig. 3. An example of execution in which the user opens the door of a fridge

the action sequence from figure 4. Each activity is chosen depending of a stochastic decision. The system generates a random number between 0 and 1. Firstly, if it is higher than 0.32, then it chooses the *task living room* to execute. Otherwise, it chooses the next task. Again, another random number is generated. If it is greater than 0.32, the *task kitchen* is executed. It goes on until the last task *get lost*. This activity triggers the irregular behavior, which has to do with going from room to room without spending much time.

Since there is access to the simulated user data, it is possible to tell when this behavior is being triggered and use this information to tag the results. This way, we can generate rows of data that represents the activation of presence sensors and accompany the row with a label indicating when the behavior being executed is *get lost*.

The character generates sequences of actions similar to the one from figure 4. This way, the character is always moving and it is generating the data we need to test the algorithms. The difference between generating the data this way and just having a dataset of the numbers is that the simulation gives a visual reference that helps to understand the data you are generating.

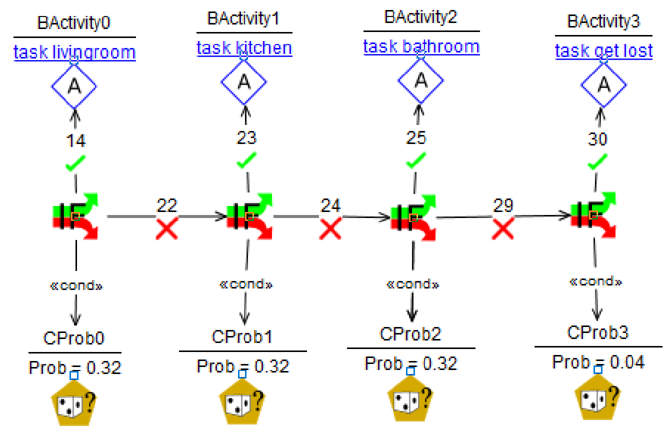

Fig. 4. The flow from which the agent chooses the next action to perform

V. Experimentation

In this experimentation, one developer designed the behavior of a person with an AIDE based on the common behaviors of disorientation, while other two researchers designed an implemented the two proposed algorithms for detecting disorientation. For the sake of reproducibility, the repository of this work [1] contains the project for executing the simulated behavior, the script with the proposed algorithms and the corresponding datasets.

In this behavior, the simulated avatar was disoriented about 4% of the time, while the remaining time presented normal behavior patterns, just as it was presented in figure 4. We labeled the different time intervals to distinguishing between simulated disorientation and normal behavior. Then, we applied both algorithms to determine which ones properly detected the disorientation according to these simulated behaviors. This experimentation simulated 310 min of real time.

In the experimentation, we collected all the measurements of these algorithms every time a presence sensor was activated, to compare the known behaviors with the prediction. Table I shows some examples of these collected data. This data include the timetamps, the output of each sensor (1: detecting presence; 0: otherwise), the action of the person, the label of the known behavior, and the prediction of each algorithm. Some of the existing actions were going to some rooms and staying there, while other actions simulated being lost and navigating disoriented through the house, randomly visiting different rooms.

[1] https://github.com/Melkoroth/AIDEdisorientExperiment.git (last accessed 12-14-2018)

timestamp	s_1	s_2	s_3	s_4	s_5	s_6	known behaviour	predicted Algorithm 1	predicted Algorithm 2
1514819514501	0	0	0	1	0	0	GoToKitchen10	Normal	Disoriented
....									
1514822418501	0	0	0	1	0	0	GoGetLost15	Normal	Disoriented
1514822434501	0	1	0	0	0	0	GoGetLost15	Normal	Normal
....									
1514822447501	0	0	0	0	0	1	GoGetLost15	Normal	Normal
1514822478501	0	0	0	1	0	0	GoGetLost15	Normal	Disoriented
1514822493501	0	1	0	0	0	0	GoGetLost15	Disoriented	Normal
....									
1514822507501	0	0	0	0	0	1	GoGetLost15	Disoriented	Normal
1514822538501	0	0	0	1	0	0	GoGetLost15	Disoriented	Disoriented
....									

TABLE I

COLLECTED DATA FROM THE SIMULATION

Finally, we measured the accuracy, precision and specificity of each algorithm, regarding the percentages of matches between prediction and known value respectively in all the cases, in only the positive cases (while disoriented) and in the negatives (i.e. while non-disoriented). Table II shows the results of all these metrics.

	Algorithm 1	Algorithm 2
Accuracy	90.72%	87.63%
Precision	83.33%	60.00%

TABLE II

MEASUREMENTS OF THE CLASSIFICATION PERFORMANCE OF THE TWO ALGORITHMS

In the initial testing, the algorithms provided worse results, because the sensing areas of two presence sensors were overlapped. We avoided this overlapping by removing an unnecessary sensor. This reflects that this approach needs to properly place the sensors avoiding situations like this.

VI. CONCLUSIONS AND FUTURE WORK

The proposed approach allows researchers to design algorithms for either measuring memory or detecting disorientation based on the outputs of IoT devices in a simulated 3D virtual living-lab environment. As a proof of concept, this work has proposed and tested two algorithms that can notify the user when certain patterns have been detected, usually related with memory losses or disorientation. This illustrative cases are designed to be deployed with IoT smart home devices, more concretely presence sensors. This approach can let researchers advance the state of the art in algorithms for tracking symptoms of diseases such as AD, like memory losses.

In the future, we plan to apply the algorithms tested with this approach in real environments in the houses of some volunteers, considering both healthy and impaired people, to assess whether the current approach can actually distinguish between these two conditions.

REFERENCES

[1] R. E. Amariglio, K. Frishe, L. E. Olson, L. P. Wadsworth, N. Lorius, R. A. Sperling, and D. M. Rentz, "Validation of the face name associative memory exam in cognitively normal older individuals," *Journal of clinical and experimental neuropsychology*, vol. 34, no. 6, pp. 580–587, 2012.

[2] I. García-Magariño, M. G. Bedia, and G. Palacios-Navarro, "FAMAP: A Framework for Developing m-Health Apps," in *World Conference on Information Systems and Technologies*, ser. Advances in Intelligent Systems and Computing, vol. 745. Springer, 2018, pp. 850–859.

[3] M. Peer, R. Lyon, and S. Arzy, "Orientation and disorientation: lessons from patients with epilepsy," *Epilepsy & Behavior*, vol. 41, pp. 149–157, 2014.

[4] E. T. Rolls and S. Wirth, "Spatial representations in the primate hippocampus, and their functions in memory and navigation," *Progress in neurobiology*, 2018.

[5] B. Bergvall-Kåreborn and A. Ståhlbröst, "Living lab: an open and citizen-centric approach for innovation," *International Journal of Innovation and Regional Development*, vol. 1, no. 4, pp. 356–370, 2009.

[6] M. Cárdenas, J. Gómez Sanz, and J. Pavón, "Testing ambient assisted living solutions with simulations," in *IFIP International Conference on Testing Software and Systems*, ser. Lecture Notes in Computer Science, vol. 11146. Springer, 2018, pp. 56–61.

[7] J. J. Gómez-Sanz, M. Cardenas, R. Pax, and P. Campillo, "Building prototypes through 3d simulations," *PAAMS*, pp. 299–301, 2016.

[8] C. Belley, S. Gaboury, B. Bouchard, and A. Bouzouane, "Nonintrusive system for assistance and guidance in smart homes based on electrical devices identification," *Expert Systems with Applications*, vol. 42, no. 19, pp. 6552–6577, 2015.

[9] Â. Costa, J. C. Castillo, P. Novais, A. Fernández-Caballero, and R. Simoes, "Sensor-driven agenda for intelligent home care of the elderly," *Expert Systems with Applications*, vol. 39, no. 15, pp. 12 192–12 204, 2012.

[10] Q. Lin, D. Zhang, K. Connelly, H. Ni, Z. Yu, and X. Zhou, "Disorientation detection by mining gps trajectories for cognitively-impaired elders," *Pervasive and mobile computing*, vol. 19, pp. 71–85, 2015.

[11] S. Teipel, A. König, J. Hoey, J. Kaye, F. Krüger, J. M. Robillard, T. Kirste, and C. Babiloni, "Use of nonintrusive sensor-based information and communication technology for real-world evidence for clinical trials in dementia," *Alzheimer's & Dementia*, vol. 14, no. 9, pp. 1216–1231, 2018.

[12] A. A. Aramendi, A. Weakley, A. A. Goenaga, M. Schmitter-Edgecombe, and D. J. Cook, "Automatic assessment of functional health decline in older adults based on smart home data," *Journal of biomedical informatics*, vol. 81, pp. 119–130, 2018.

[13] J. J. Barton and V. Vijayaraghavan, "Ubiwise, a simulator for ubiquitous computing systems design," *Hewlett-Packard Laboratories Palo Alto, â AI HPL-2003-93*, 2003.

[14] T. Garcia-Valverde, F. Campuzano, E. Serrano, A. Villa, and J. A. Botia, "Simulation of human behaviours for the validation of ambient intelligence services: A methodological approach," *Journal of Ambient Intelligence and Smart Environments*, vol. 4, no. 3, pp. 163–181, 2012.

[15] K. Sanmugalingam and G. Coulouris, "A generic location event simulator," in *International Conference on Ubiquitous Computing*. Springer, 2002, pp. 308–315.

[16] A. A. Nazari Shirehjini and F. Klar, "3dsim: rapid prototyping ambient intelligence," in *Proceedings of the 2005 joint conference on Smart objects and ambient intelligence: innovative context-aware services: usages and technologies*. ACM, 2005, pp. 303–307.