Towards Robots-Assisted Ambient Intelligence

Marin Lujak, Noury Bouraqadi, Arnaud Doniec, Luc Fabresse, Anthony Fleury, Abir B Karami, Guillaume Lozenguez

Towards Robots-Assisted Ambient Intelligence

Marin Lujak, Noury Bouraqadi, Arnaud Doniec, Luc Fabresse, Anthony Fleury, Abir Karami, and Guillaume Lozenguez

IMT Lille Douai, France, firstname.lastname@imt-lille-douai.fr

Abstract. Integrated networks of mobile robots, personal smart devices, and smart spaces can provide for a more accurate data for user assistance than if the former are used individually. We call this network of personal and smart space devices and robots "Robots-Assisted Ambient Intelligence" (RAmI). Additionally, with the application of distributed network optimization, not only can we improve the assistance of an individual user, but we can also minimize conflict or congestion created when multiple users in large installations use the limited resources of RAmI that are spatially and temporally constrained. The emphasis of RAmI is on the efficiency and effectiveness of multiple and simultaneous user assistance and on the influence of individual robot actions on the desired system's performance. In this paper, we model RAmI as a multi-agent system with AmI and robot agents. Moreover, we propose a modular three-layer architecture for each robot agent and discuss its application and communication requirements with the emphasis on interaction between robots, humans, and AmI agents to facilitate efficient usage of limited RAmI resources. Our approach is showcased by means of a case study where we focus on meal and medicine delivery to patients in large hospitals.

Keywords: Service robotics; ambient intelligence; ambient assisted living; multi-robot systems; multi-agent systems; patient care

1 Introduction

Ambient Intelligence (AmI) uses multiple sensors fixed in a smart space to assist user's activities through recommendation, guidance, and appliance control. However, AmI is not capable of interacting with a user by physical contact since its user interfaces are usually tactile, auditive, and/or visual. On the other hand, mobile robots with installed robot arms are capable of physical user interaction, though with a world view that is limited to their local sensory and communication capabilities, see, e.g., [3].

The quality of service provided by mobile robot teams (MRT) to simultaneous multiple users (with the emphasis on patients and elderly with decreased mobility) depends on the efficiency of the robots' coordination with one another and with humans. To keep a good MRT performance in simultaneous multiple tasks, an updated task information is required. Even though the MRT quality of service depends on the quality of the available information that can be facilitated by maintaining the MRT connectivity [26], MRT task assignment can be performed both in perfect (e.g., [5,9]) and imperfect robot networks, e.g., [18]. Due to the loss in the information quality, the efficiency of a

MRT in the task execution can fall rapidly, e.g., [18,19]. The strategy to employ to mitigate this problem depends also on the environment that can be collaborative, neutral, or adversarial [18,19]. Providing redundant robots to keep the network's connectivity is a possible approach to this problem. However, it is costly and can create congestion in narrow spaces. This is why, in this paper, we propose to network mobile robots, users' smart devices, and AmI networks, such that we can use more accurate data for decision-making than when the former are used individually. Even more, with the application of distributed network optimization, not only can we improve the assistance of an individual user, but we can also ensure that robots' actions that are geographically and temporally constrained in the usage of limited resources do not result in conflict or congestion. We call this network of AmI, personal devices, and robots "Robots-Assisted Ambient Intelligence (RAmI)". The emphasis of RAmI is on the quality of service in simultaneous multiple users' assistance and the influence of individual robot decisions on the desired system's performance. One of the issues of RAmI in large installations is its computational efficiency. Mobile robot teams are intrinsically decentralized and should act quickly and efficiently in real-time in large smart spaces, e.g., [8,9]. One of the main tasks of multi-robot teams in patients' assistance in large hospitals is meal and medicine delivery and related task assignment depending on the minimization of the delivery times and other constraints like the cost of the RAmI resources.

There are various centralized and distributed approaches to the MRT task assignment e.g., [5,18]. To lower the computation time, we should balance between the robots' communication and computation load, but foremost, we should provide for a self-reconfigurable robot architecture that assures fast and efficient decision making. The objective of this paper is to consider requirements for such an architecture and to discuss interaction constraints that assure efficient and effective task performance in meal and medicine delivery to patients in large crowded hospitals.

This paper is organized as follows. In Section 2, we describe the State-of-the-Art related with RAmI. Section 3 describes the proposed three-layer decision-making architecture for each robot in RAmI. In Section 4, we formulate the RAmI coordination problem related with multiple simultaneous patients that require meal/medicine delivery by a multi-robot team. The principles of the proposed architecture are demonstrated by means of a case study in Section 5. We draw conclusions in Section 6.

2 Background

With the improvements in service robotics, the same can be used to help people in some daily activities like vacuum cleaning. The recognition and analysis of the user's activities facilitates better user assistance in the performance of these activities [2,14]. For this aim, usually, smart homes are equipped with sensors, actuators and alarms while users dispose of smart devices for the interaction with the smart home [4].

However, actual scientific achievements in the robotic assistance of a user are still limited to a set of predefined activities and are, as such, still far away from the realization of fully intelligent robots that can substitute human care givers. One of the many challenges that still has to be resolved, as will be presented in the following, is the

problem of the coordination of a robot with the rest of the robot team when assisting multiple simultaneous patients in a common space.

By integrating stand-alone robots with web services and ambient intelligence technologies, we form ubiquitous robots [3,13]. In the focus of ubiquitous robotics is an individual human user and how this hybrid system can enhance the quality of living and working of a single person. The objective is the creation of a physical and virtual companion that can assist a user in his/her daily activities, and the creation of an autonomic guard capable to protect and rescue people. However, there are very few works concentrating on the coordination among robots and multiple humans in their activities that share the same resources, related congestion control and the influence of an individual (robot or human) action on the system's performance in such complex systems, e.g., [33].

Relatedly, a distributed ROS-based AmI architecture DAmIA integrating robotic and AmI sensors for human tracking has been proposed in [23]. A survey of cloud robotics that leverages the ad-hoc cloud formed by communicating robots, and an infrastructure cloud was presented in [13]. Moreover, in [17], we proposed ORCAS architecture for manufacturing MRTs that configures and schedules robots based on robots' and tasks' semantic descriptions.

In ORCAS [17], we consider a heterogeneous and reconfigurable multi-robot system made of multiple robot platforms, grippers, and robotic tools that can be combined to create new robotic configurations during operation, if necessary. The setting of the multi-robot system is a shop-floor with the assembly of multiple products. Here, every robot is considered a collaborative agent whose architecture is made of three layers: semantic, scheduling and the execution layer. The aim of the semantic layer is to find feasible robots' configurations that can satisfy customer demand based on given semantic descriptions about factory setting, available resources and product specifications. The semantic layer generates compatible subsets of resources for the given tasks. The scheduling layer determines robot-task assignments and sequencing of tasks assigned to each robot configuration considering task interrelations and the robot assembly capacities. The objective is to seamlessly optimize robots' performance by dynamic reconfiguration and rescheduling in case of contingencies thus minimizing overall assembly costs and off-line times. The solution is found through distributed minimization of total production time and cost considering resource combinations obtained from the semantic layer. We apply a modification of dynamic auction-based negotiation [18]. The execution layer monitors the correct execution of the schedule in real-time. In case of unpredicted contingencies, the objective here is to carry out local actions to minimize their effects. The schedule's quality and stability are controlled in real-time, e.g., [12].

The ORCAS architecture was designed for the use on reconfigurable robots working on a manufacturing shop floor. Therefore, in this paper, we modify the ORCAS architecture for the case of a team of heterogeneous mobile service robots with a fixed configuration for assistance of multiple human users in large installations. By the fixed configuration of a robot, we mean that the configuration cannot be rearranged during the operation times.

3 RAmI architecture

The architecture used for the distributed coordination of robots in task assignment and routing is implemented in each one of the robots and is presented in Fig. 1. It con-

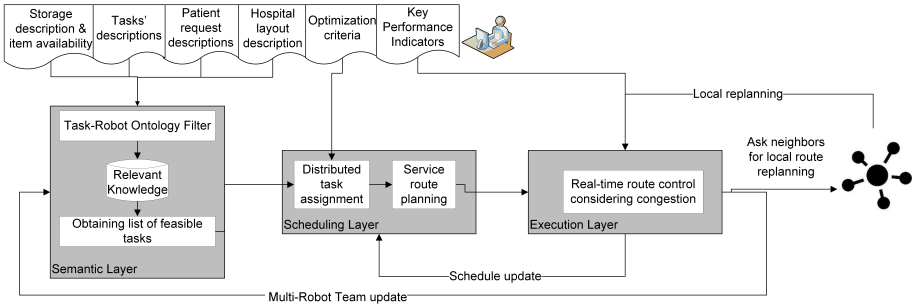

Fig. 1. Proposed RAmI architecture implemented in each robot

tains semantic, scheduling and the execution layer. Contrary to ORCAS, in RAmI, we assume that robot configuration is fixed and that each robot's delivery capacity is limited by maximum item's weight and dimensions. Moreover, all resources and each item delivery are semantically described by a human operator: e.g., meal/medicine, time of delivery, weight, dimensions, and type of a meal/medicine. The semantic storage description contains the information of available items and their hospital locations.

In the semantic layer, a set of compatible robots for each patient demand is found by using a DL inference engine and SPARQL query language. Scheduling layer contains the task assignment and route planning module. Based on the semantically described delivery demand, each robot agent $o \in O$ coordinates with other robot agents for the task assignment through the bi-level task assignment algorithm in [16]. While MRT is responsible of the MRT task assignment, the AmI network is responsible of updating the travel times under congestion in the network and distributively optimizing robots' routes by using the route finding algorithm in [20]. Robots receive updated routes and travel times info from the belonging SA. In the execution layer, the individual performance is monitored in real time and in case of unpredicted events, a robot tries to coordinate locally with its neighbors to lower their impact. If the local coordination is not sufficient, the scheduling layer recomputes the robots' routes. In the case of larger contingencies that make the schedule infeasible or the addition of robots that can improve the MRT's performance, semantic layer updates matchings between the tasks and the MRT.

3.1 RAmI as a multi-agent system

We design RAmI as a multi-agent system made of mobile robot, personal smart device, and smart space agents. Smart device agents are installed on an app of a smart device of each user while smart space agents monitor a region within the range of its sensors.

The sensors can be cameras and iBeacons. The signals emitted by iBeacon sensors are read by smart device agents of the users. These signals, together with cameras serve to better locate the users and recognize their activities (aided by accelerometers placed on the smart devices). Smart space agents keep the track of the congestion and the updated shortest travel times between different origin-destination pairs.

In everyday activities, all these agents must coordinate throughout the day to facilitate patient assistance. Tasks include: encouraging physical activity, medical supervision, offering entertainment, maintaining social ties, item delivery (e.g., meal or a medicine) and assisting a patient in the case of urgency. These objectives for the agents can sometimes be antagonistic: for example, proposing physical activity while the cardiac sensor detects a problem. Sometimes, certain objectives are given priority because of urgency: the elderly person has fallen to the ground and cannot get up again. The activities should, therefore, be coordinated based on semantic rules on the priority of these activities, which is performed in the semantic layer of the proposed architecture.

Coordination must therefore take different user and task requirements into account to find the best possible sequence of tasks. In the patient assistance context, the best sequence is the one that brings the best well-being of the patient. To achieve this goal, we propose a coordination involving two steps. The first step consists in finding all the suitable and consistent sequences of tasks. Indeed, we can notice that some tasks are not compatible with each other and cannot be performed at the same time. For example, simultaneous eating and medical assistance. Following the same idea, some sequences of activities are not desirable for the patient, for instance: two consecutive heavy physical activities. All these requirements can be easily integrated by semantic matching and then a classical constraint programming approach to generate all the possible sequences.

The second step consists in finding the best solution among all the possible sequences. Designing a mathematical program related with the well-being of a patient is a challenging task since it depends on various factors. It is directly related to the physical condition, mental state, and habits of a patient. The context in which tasks are performed is also important: time of day, location, etc. These considerations lead to the inclusion of the patient in the decision-making process preferably by learning the sequences according to rewards. These rewards could be awarded directly by the patient depending on the sequence of activities offered. These rewards could also be obtained through physiological measures. For example, an evening activity sequence that improves the patient's sleep would have a high reward.

With this aim, we may use a reinforced learning approach. It consists of learning a policy (following actions to be carried out) based on an initial state and a set of expected rewards. In our case, all the actions to be explored through learning would be the result of first step, i. e. all the possible sequences of tasks.

The figure 2 gives an overview of this two-step coordination mechanism.

3.2 Influence of individual decisions on the system's performance

Users and (human) robot operators can be considered as sources of relevant knowledge (Fig. 1). Such knowledge helps in making adaptive and coherent decisions to improve the AmI performance.

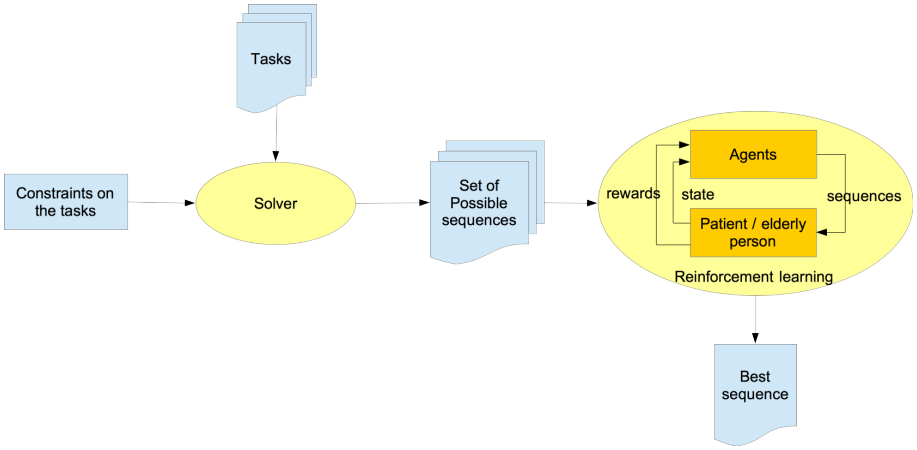

Fig. 2. Overview of the two-step coordination mechanism

To better assist patients and caregivers, robots should reply to patients' requests that are declared explicitly and others that might be expressed implicitly. In fact, users through their reactions and feedback are an important source of information on how to achieve tasks in an adaptive and personalized way (time to give medicine, entertainment recommendation, etc.). This implies that feedback should be collected using existing sensors. This feedback can either be implicit (normal operation of standard devices like turning off the lights) or explicit through friendly interfaces or by interacting with the robots.

Relatedly, the CASAS (Center for Advanced Studied in Adaptive Systems) project including the CASA-U (the CASAS User interface) [24] introduces an adaptive smart home system that applies machine learning techniques to discover patterns in resident's daily activities and to dynamically adapt to the user's explicit or implicit wishes in daily routine activities. CASAS can also adapt to the changes in discovered patterns based on the resident implicit and explicit feedback and can automatically update its model to reflect the changes.

Moreover, the COBOT chat system proposed in [7] concentrates on adaptive behavior in multi-user environments using reinforcement learning. However, as the authors argue in their paper, the problem of learning from multiple users' (not experts) feedback implies that the users have different characters and depending on their characters and the application itself, they tend to react by rewarding with positive actions or only penalizing with negative actions. Also, a lot of ambiguities in the interactions can result from human mistakes or missing information in the model. The learning process should deal with the confusion caused by the lack of feedback from users. A lack of feedback in some scenarios might represent a satisfaction and in others it might represent an unsatisfying situation. Also, it is important to deal with the problem of contradictions in feedback. Analyzing contradictions might help in detecting the need of a more representative model of the current environment. Furthermore, in the RAmI architecture,

observations are gathered from several sensors and both smart space agents and robots can help in handling conflict and ambiguities in gathered data (feedback).

In [10,11], Karami et al. focus on smart homes and companion robots that are able to adapt to different users by learning the preference of each of them using their individual implicit and explicit feedback and those of others. They propose global architecture that integrates users' profiles to the learning and decision making processes. In this work, an attribute is defined as an information representing a part of the state of the environment (for instance level of brightness, temperature, etc.) or the user profile (age, gender, habits of living, etc.). Some of the user profile attributes are learned automatically using the activity recognition system. For example, an attribute representing whether the person has a habit of practicing sport frequently can be learned/updated from the observations. The goal is to be able to generalize the learned adaptive behavior to new situations and unknown users and for decreasing the complexity of convergence to an optimal adaptive policy (decreases the size of needed dataset to reach an acceptable adaptive behavior).

3.3 Interaction of a robot with its environment

One of the goals of RAmI is to network mobile robots, users' smart devices, and AmI networks to empower multiple and simultaneous end-users of RAmI by enabling them to use and benefit from their surrounding technologies (sensors, actuators, etc.) through useful services. This ambitious objective is rather general and abstract but it already pinpoints a drawback in current systems: the lack of discoverability, flexibility and dynamic interactions to really support Robots-Assisted Ambient Intelligence (RAmI). Therefore, in this section, we focus on the capabilities of a mobile robot in the context of RAmI.

Mobile and autonomous robots are constituted of a network of sensors and actuators as well as of one or more computing unit. The autonomous property means that a mobile robot can achieve some missions without any human intervention. Therefore, the control software of such a robot must be able to "take decisions" based on information coming from its sensors and send commands to its actuators.

In the context of RAmI, the robot's decisions can be improved using non-local information coming from ambient sensors (camera, iBeacons, etc.) [27,25] and the robot capabilities can also be improved by ambient actuators, e.g., automatic doors [28]. Nevertheless, such interactions between the environment and a mobile robot requires to rethink current control architectures of mobile robots.

We are specifically interested in the *dynamic update* of the control software of a robot to deal with it. Indeed, interacting with a new equipment discovered in the environment requires to change the control software of the robot while it is running. Obviously, such a change should be applied in safe and atomic way. Previously, we worked on an architectural solution for robotics applications and propose the MaDcAr model [6]. More recently, we are working on a general-purpose for dynamic software update (DSU) [31] in the context of dynamic object-oriented languages. Dynamic Software Update (DSU) solutions allow updating applications while they are executing. Some modifications do not allow one to continue running the application without a proper migration of its state, as the modified instances and method are used by running threads.

Moreover, these modifications must be applied all at once (atomicity) and correctly migrate the application state and the running threads. In our robotics scenario, the use of such a DSU mechanism eliminates the stop, install and restart cycle. Updating a running application should preserve its running state and the service provided to the users. Our solution provides the means to migrate the application's state in an atomic manner, guarantee the correct continuation of all the application's threads and validate the application constraints after the execution of the update.

We believe that such an infrastructure is crucial for building ambient-aware mobile robots.

Efficient distributed planning under uncertainty Flexible robot control architecture, with continuous service and multi-agent system (Robot, Smart Environment and Human) induces the design of specific Artificial Intelligence for control supervision. All those aspects imply to deal with uncertain events when planning the multi-robot movements. Markov decision process permits to model such a problem. Unfortunately optimally-solving decentralized Markov decision processes is impossible for problems involving more than tree robots in a environment described by few variables [1].

The assumption in *RAmI* architecture (Fig. 1) is to take advantage of distributed planning where each robot is responsible for planing its own movements (module *service route planning*). The coordination is devoted to the module *distributed task assignment*.

Previous work that was done based on this approach gave some interesting results [15] that were limited to a Multi-Robot fleet working alone in a human free scenario. The challenge is triple: i) to allow individual decision making process to communicate with the coordination model the data regarding the travel time estimation ($t_{i,j}$) congestion of the environment, the demands and the available resources in the smart environment; ii) efficiently supervising robot control in dynamic configuration (environment, control software), and iii) generating new plans in few second to allow on-line and on-board planning and re-planning in coordination context.

3.4 Robot Navigation

To achieve any useful task (e.g. delivering medicine or food, transporting blood samples, medicine or a meal), a robot needs to navigate inside a building. Thus, it needs a map of the building to figure out how to go to target places. To navigate, a robot also needs to localize itself, i.e. a robot needs to match observations of its surroundings at any given moment and match it with a point on the map. Relying on a GPS is not an option, since robots operate inside buildings that block GPS signals emitted by satelites.

Map construction can be performed by robots themselves thanks to **Simultaneous Localization and Mapping (SLAM)** algorithms [30]. These algorithms allow robots to incrementally build a map, and localize within. There exist different families of SLAM algorithms with different approaches. Some rely on vision, and extract features from pictures taken by cameras. Other measure distances to objects around a robot using some range sensor.

Another dimension of SLAM algorithms is the nature and the dimension of the produced maps. Maps can be 2D or 3D. Map can also be either metric or topological.

Metric maps are quantitative representation of the environment (e.g. lengths, widths). While topological maps are a qualitative representation of the environment. They allow distinguishing different locations of the mapped environment, and how they are connected. Locations can be identified as high-level entities (e.g. corridor, room), or low-level sets of extracted features.

The most widely spread SLAM algorithms rely on 2D metric SLAM based, that often rely on laser range sensors. We use it to illustrate the functionning of SLAM. Starting from an initial position, distances to surrounding objects measured by a laser scanner allow building a first parial map. This map is then extended with new sensory data from regions where the robot moves to. The robot position relative to the initial point is estimated from different data sources. That can be distances to objects that are sensed at different times, motion derived from an Inertial Measurement Unit (IMU), or an odometer. Eventually, this process result into a complete map of all places reachable by a robot.

Navigation for Map Construction The SLAM algorithms allows building a map based on data collected from robot sensors. However, a robot performing SLAM has to "somehow" move around the building. This motion can be conducted manually, by relying on a human operator to guide the robot. Conversely, the robot can autonomously roam in the environment to construct the map.

Autonomous map construction is also referred to as **exploration**. An efficient explorer robot has to plan its motion in a way to reach quickly unmapped areas. The seminal work of Yamauchi addresses this issue by targetting **frontiers** between explored areas, and ones that are still unexplored [32]. A frontier separtes free space, i.e. space where a robot can go, with areas that have not been sensed yet.

Exploration algorithms define strategies to select frontiers that are (1) reachable to the robot, and that (2) maximize the acquired knowledge, in a minimum time. A frontier is reachable to a robot if the passage along the path to go the frontier is always wider than the robot. The amount of acquired knoweldge refers to the size of the area mapped once reaching the frontier. A straight forward solution is to select the nearest reachable frontier.

Machine to Machine Communication for Navigation According to the state of the art, only wheeled robots can safely operate in an environment along people [29]. However, hospitals are multi-floor buildings, equipped with elevators. To allow robots reach any floor, robots need to use the elevators.

Obviously, robots can be adapted to push buttons and call the elevators, as human do. A more straight-forwad alternative is to enable wireless communication between robots and elevator. Thus a robot can send a message to call an elevator and the elevator can notify the robot once available.

This machine to machine communication paves the way for optimal multi-floor path planning. At the decision level, a robot can query multiple elevators for their state and use frequency. An idle elevator might be a better choice than one in use.

At the execution level, once a robot can call the elevator before reaching it. The call would be timed in a way to ensure the elevator is there with the door open when the robot reaches it.

Multi-Robot Map Construction and Update Map construction can take only few minutes in a small building. However, since we target hospitals that are often large if not very large buildings, mapping is likely to require more time. Regardless the size of the building, having many floors also contributes to slowing down the mapping process.

The problem is that building a first map is not enough. To optimally compute paths and assign tasks to robots, we need accurate maps that take into account the day to day changes that may occur. For example, building cleaning, maintenance and repair might require locking at least partially a corridor, or an elevator for few hours, days or even longer. Other examples are plants, decorations, or trash cans that might be moved say upon cleaning.

To both speedup map construction and update, an attractive option is to rely on a called a **multi-robot system** [22]. A such system is built by making a group of robots collaborate to achieve the tasks at hand. In our case, we need to distinguish here between map construction and map updating. In both cases, robots need to actively coordinate their actions to draw actual benefits.

During exploration, robots have to communicate in order to assign frontiers or floors to each of them, and hence explore a different part of the building. The assignment algorithm should be smart enough to reduce if not minimize overlaps, collisions, and redundant explorations. The global map is built by merging maps produced by various robots. This operation is complex because partial maps built by individual robots are not perfect because of incertitude on the robot position and sensors noises. Merging should take this into account to ensure a good alignment of partial maps, and correction of disparities of maps areas that were constructed by different robots exploring the same areas.

Map updating can be done either by rebuilding the map from scratch or by correcting an existing map. Actually, both options can be combined. On a regular basis (e.g. once a day), the robotic fleet can rebuild the whole map. Then, this map can be updated once a robot detects significant changes in a given area.

4 Case study problem formulation

In this section, we showcase the functioning of the RAmI architecture in the problem of a meal or medicine delivery to multiple patients in a smart hospital.

We represent a smart hospital building layout by an undirected graph $G = (N, A)$, where N is a set of smart space agent (SA) nodes representing rooms, offices, halls, and, in general, a relatively small portion of space within a building. Each arc $(i, j) \in A$ has an associated travel time t_{ij}, which depends on its length and the relative congestion. Each SA is responsible of monitoring its surrounding area (by, e.g., iBeacons and cameras) to locate users' momentary positions and compute space congestion. Moreover, we assume that each mobile robot agent assumed with a limited communication range is positioned in one of nodes $n \in N$ and it can communicate with the rest of robots

within its communication range and with the belonging SA. Alike, each user is represented by a user agent u installed on an app of a user's personal smart device (e.g., tablet or a smartphone) containing user-relevant info and able to communicate with the closest SA and the robots if located within their communication range.

We consider multiple simultaneous item delivery by a MRT to patients in a building. The most frequent items for delivery to patients are a meal or a medicine. We assume that there is a set $I \subseteq N$ of item storage locations in the building. Furthermore, let $O \subseteq N$ and $D \subseteq N$ be the set of all robots and patients at their momentary positions, respectively. Moreover, we assume that the items are packaged such that only grasping is required to handle them. Then, the objective is to assign item delivery tasks to robots in O such that the overall delivery time is minimal considering travel time under congestion from the robots' momentary locations through item storages in I, and delivering the items to patients in D.

5 Case study setting

We demonstrate the functionality of the proposed approach by means of a simple case study example in Figure 3. Given is a simple scenario of a building network with 5

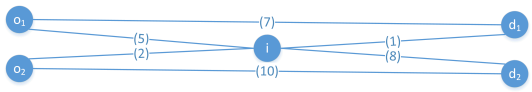

Fig. 3. A simple 5 node smart space network. Arcs' travel times in parentheses

nodes and 6 arcs. There are two mobile robots positioned at o_1 and o_2, two patients (at d_1 and d_2), and inventory node i. Moreover, given are arcs' travel times t_{ij} in minutes for each arc (i, j). Patients' delivery items are ontologically described through RDF. The objective is to find routes from the robots' positions o_1 and o_2 through inventory i to patients d_1 and d_2 that minimize the overall patient delivery time.

Let us assume that both robots can deliver the demands of both patients d_1 and d_2. Then, in the scheduling layer, the robots get assigned to patients' demands (tasks) following steps in the MRTA algorithm [5] based on the updated paths with shortest travel times given by SAs. The travel time computation is done by the AmI network where SA nodes compute distributively the routes through [20].

Let us analyze this simple example. Robots start the task assignment through [16]. From o_1 to d_1 and from o_2 to d_2, there is only one simple path available passing through i. The overall cost of this assignment is 16. From o_1 to d_2 and from o_2 to d_1, there are four simple paths available for each one of the patient nodes d_1 and d_2. The overall cost of optimal paths $(o_1, i), (i, d_2)$ and $(o_2, i), (i, d_1)$ is also 16. Since both assignments have the same cost, the solution is found lexicographically. In the case of contingencies during the moving from one node to another, the robots try to coordinate among themselves by locally recomputing their routes by following the algorithm in [16]. If

the solution is unsatisfactory, they recompute routes in the scheduling layer. If one of them breaks, then the other recomputes its route starting from the semantic layer.

In case of high travel time variations, robots should be able to reroute. This is where SA agents play a crucial role in observing congestion and updating travel times. The SA agents compute the routes and inform the robots of the available routes' arrival times. MRT performance depends on the navigational maps (i.e. areas where the robots can safely go) by tracking human trajectories and integrating them within the probabilistic map which is built directly through the conventional sensory readings (see, e.g., [21]).

6 Conclusions

In this work, we proposed "Robots-Assisted Ambient Intelligence (RAmI)" architecture for robots that should work in multi-robot teams integrated with the networks of smart spaces (Ambient Intelligence networks) and users' personal smart devices. We discussed some open challenges to reach fully intelligent multi-robot teams that can assist people in various daily activities. Moreover, we showcased the functioning of the RAmI architecture on a meal or medicine delivery to simultaneous multiple patients needing assistance.

The focus of RAmI is on the coordination of robot teams and multiple humans that share the same space resources in their daily activities, related congestion control and the influence of an individual (robot or human) action on the system's performance in such complex systems.

In future work, we intend to analyze in depth the efficiency of our RAmI approach related with unpredictable scenarios through simulations on building networks of varying complexity. Related is the issue of scalability. We plan to evaluate real-time responsiveness of our approach to varying number of users and a varying size of the evacuation network.

Acknowledgements. This work has been partially supported by the COMRADES project within the framework "Fonds d'amorçage Santé" by Institut Mines Telecom in France.

References

1. Bernstein, D.S., Zilberstein, S., Immerman, N.: The Complexity of Decentralized Control of Markov Decision Processes. In: 16th Conference on Uncertainty in Artificial Intelligence (2000)
2. Boger, J., Hoey, J., Poupart, P., Boutilier, C., Fernie, G., Mihailidis, A.: A planning system based on markov decision processes to guide people with dementia through activities of daily living. IEEE Transactions on Information Technology in Biomedicine 10(2), 323–333 (April 2006)
3. Chibani, A., Amirat, Y., Mohammed, S., et al.: Ubiquitous robotics: Recent challenges and future trends. Robotics and Autonomous Systems 61(11), 1162–1172 (2013)
4. Cook, D., Krishnan, N.: Activity learning from sensor data (2014)
5. Giordani, S., Lujak, M., Martinelli, F.: A distributed algorithm for the multi-robot task allocation problem. In: García-Pedrajas, N.e.a. (ed.) Trends in Applied Intelligent Systems, Proceedings of the 23rd Int. Conf. on Industrial Engineering and Other Applications of Applied Intelligent Systems, IEA/AIE 2010, LNCS, vol. 6096, pp. 721–730 (2010)

6. Grondin, G., Bouraqadi, N., Vercouter, L.: Madcar: an abstract model for dynamic and automatic (re-)assembling of component-based applications. In: Proceedings of the 9th International Symposium on CBSE (Component-Based Software Engineering). pp. 360–367. LNCS, Springer, Sweden (Jun 2006)
7. Isbell Jr, C.L., Kearns, M., Singh, S., Shelton, C.R., Stone, P., Kormann, D.: Cobot in lambdamoo: An adaptive social statistics agent. Autonomous Agents and Multi-Agent Systems 13(3), 327–354 (2006)
8. Julian, B.J., Angermann, M., Schwager, M., Rus, D.: Distributed robotic sensor networks: An information-theoretic approach. The Int. J. of Robotics Res. 31(10), 1134–1154 (2012)
9. Kantaros, Y., Zavlanos, M.M.: Global planning for multi-robot communication networks in complex environments. IEEE Transactions on Robotics 32(5), 1045–1061 (2016)
10. Karami, A.B., Fleury, A., Boonaert, J., Lecoeuche, S.: User in the loop: Adaptive smart homes exploiting user feedback—state of the art and future directions. Information 7(2), 35 (2016)
11. Karami, A.B., Sehaba, K., Encelle, B.: Adaptive artificial companions learning from users' feedback. Adaptive Behavior 24(2), 69–86 (2016)
12. Katragjini, K., Vallada, E., Ruiz, R.: Flow shop rescheduling under different types of disruption. International Journal of Production Research 51(3), 780–797 (2013)
13. Kehoe, B., Patil, S., Abbeel, P., Goldberg, K.: A survey of research on cloud robotics and automation. IEEE Transactions on automation science and engineering 12(2), 398–409 (2015)
14. Kon, B., Lam, A., Chan, J.: Evolution of smart homes for the elderly. In: Proceedings of the 26th International Conference on World Wide Web Companion. pp. 1095–1101. WWW '17 Companion, International World Wide Web Conferences Steering Committee, Republic and Canton of Geneva, Switzerland (2017), https://doi.org/10.1145/3041021.3054928
15. Lozenguez, G., Adouane, L., Beynier, A., Mouaddib, A.I., Martinet, P.: Punctual versus continuous auction coordination for multi-robot and multi-task topological navigation. Autonomous Robots 40(4), 599–613 (2016), http://dx.doi.org/10.1007/s10514-015-9483-7
16. Lujak, M., Billhardt, H., Ossowski, S.: Distributed coordination of emergency medical service for angioplasty patients. Annals of Mathematics and Artif. Intell. 78(1), 73–100 (2016)
17. Lujak, M., Fernandez, A.: ORCAS: Optimized robots configuration and scheduling system. In: Proc. of the 30th ACM Symp. On Applied Computing. vol. 1, pp. 327–330 (2015)
18. Lujak, M., Giordani, S.: On the communication range in the auction-based multi-agent target assignment. In: Bettstetter, C., Gershenson, C. (eds.) Self-Organizing Systems. Proc. of the 5th Int. WS, IWSOS 2011, LNCS, vol. 6557, pp. 32–43 (2011)
19. Lujak, M., Giordani, S., Ossowski, S.: Value of incomplete information in mobile target allocation. In: Klügl, F., Ossowski, S. (eds.) Multiagent System Technologies. Proceedings of the 9th German Conference, MATES 2011, LNCS, vol. 6973, pp. 89–100 (2011)
20. Lujak, M., Giordani, S., Ossowski, S.: Route guidance: Bridging system and user optimization in traffic assignment. Neurocomputing 151(1), 449–460 (2015)
21. Papadakis, P., Rives, P.: Binding human spatial interactions with mapping for enhanced mobility in dynamic environments. Autonomous Robots 41(5), 1047–1059 (2017)
22. Parker, L.E.: Handbook of Robotics, chap. 40: Multiple Mobile Robot Systems, pp. 921–941. Springer (2008)
23. Petitti, A., Di Paola, D., Milella, A., Lorusso, A., Colella, R., Attolico, G., Caccia, M.: A network of stationary sensors and mobile robots for distributed ambient intelligence. IEEE Intelligent Systems 31(6), 28–34 (2016)
24. Rashidi, P., Cook, D.J.: Keeping the resident in the loop: Adapting the smart home to the user. Systems, Man and Cybernetics, Part A: Systems and Humans, IEEE Transactions on 39(5), 949–959 (2009)

25. Rodic, A., Katie, D., Mester, G.: Ambient intelligent robot-sensor networks for environmental surveillance and remote sensing. In: 2009 7th International Symposium on Intelligent Systems and Informatics. pp. 39–44 (Sept 2009)
26. Sabattini, L., Chopra, N., Secchi, C.: Decentralized connectivity maintenance for cooperative control of mobile robotic systems. The International Journal of Robotics Research 32(12), 1411–1423 (2013)
27. Saffiotti, A., Broxvall, M.: Peis ecologies: Ambient intelligence meets autonomous robotics. In: Proc of the Int Conf on Smart Objects and Ambient Intelligence (sOc-EUSAI. pp. 275–280 (2005)
28. Sakaguchi, T., Ujiie, T., Tsunoo, S., Yokoi, K., Wada, K.: Intelligent ambience-robot cooperation - door-closing tasks with various robots -. In: 2009 IEEE Workshop on Robotic Intelligence in Informationally Structured Space. pp. 60–65 (March 2009)
29. Siciliano, B., Khatib, O. (eds.): Handbook of Robotics. Springer (2008)
30. Stachniss, C.: Robotic Mapping and Exploration, vol. 55. Springer (2009)
31. Tesone, P., Polito, G., Fabresse, L., Bouraqadi, N., Ducasse, S.: Instance Migration in Dynamic Software Update. In: Proceedings of International Workshop on Meta-Programming Techniques and Reflection (META 2016) (2016), http://car.mines-douai.fr/luc/files/pdfs/2016-Tesone-META.pdf
32. Yamauchi, B.: A frontier-based approach for autonomous exploration. In: Proceedings of CIRA'97 (1997)
33. Yan, Z., Jouandeau, N., Cherif, A.A.: A survey and analysis of multi-robot coordination. International Journal of Advanced Robotic Systems 10(12), 399 (2013)