

COACHES: An assistance Multi-Robot System in public areas

L. Jeanpierre*, A.-I. Mouaddib*, L. Iocchi[†], M.T. Lazaro[†], A. Pennisi[‡], H. Sahli[‡], E. Erdem[§], E. Demirel[§] and V. Patoglu[§]

*GREYC Lab, University of Caen, France

[†]Dept. of Computer, Control and Management Engineering, Sapienza University of Rome, Italy

[‡]Vrije Universiteit Brussel, Belgium

[§]Sabanci University, Turkey

Abstract—In this paper, we present a robust system of self-directed autonomous robots evolving in a complex and public spaces and interacting with people. This system integrates high-level skills of environment modeling using knowledge-based modeling and reasoning and scene understanding with robust image and video analysis, distributed autonomous decision-making using Markov decision process and Petri-Net planning, short-term interacting with humans and robust and safe navigation in overcrowding spaces. This system has been deployed in a variety of public environments such as a shopping mall, a center of congress and in a lab to assist people and visitors. The results are very satisfying showing the effectiveness of the system and going beyond just a simple proof of concepts.

I. INTRODUCTION

Public spaces in large cities are increasingly becoming complex and unwelcoming environments. Public spaces progressively become more hostile and unpleasant to use because of the overcrowding and complex information in signboards. It is in the interest of cities to make their public spaces easier to use, friendlier to visitors and safer to increasing elderly population and to citizens with disabilities. Meanwhile, we observe, in the last decade a tremendous progress in the development of robots in dynamic, complex and uncertain environments. The new challenge for the near future is to deploy a network of robots in public spaces to accomplish services that can help humans. Inspired by the aforementioned challenges, COACHES project addresses fundamental issues related to the design of a robust system of self-directed autonomous robots with high-level skills of environment modeling and scene understanding, distributed autonomous decision-making, short-term interacting with humans and robust and safe navigation in overcrowding spaces.

COACHES project provides a modular architecture integrated in robots. We deployed COACHES at Caen city in the "Rives de l'Orne" shopping mall. It is a cooperative system based on fixed cameras and mobile robots. The fixed cameras can do object detection, tracking and events detection (objects or behavior). The robots combine these information with the ones perceived via their own sensors, to provide information through its multi-modal interface, guide people to their destination, show tramway stations and transport goods for elderly people, etc.... The COACHES robots use different modalities (speech and displayed information) to interact with the mall visitors, shopkeepers and mall managers. The project has enlisted an important

end-user (Caen la mer) providing the scenarios where the COACHES robots and systems will be deployed.

II. OVERALL SYSTEM DESCRIPTION

A. General Architecture and functionalities

The general architecture of the embedded software, in each robot, has the following components (Figure 1): (i) *Modelling and reasoning*: Modelling a variety of static/dynamic knowledge about the environment formally in three knowledge bases (KB): (SEM) static properties of entities in the environment and their relations, (COM) general static common sense knowledge about shopping malls (taxonomic knowledge and defaults), and (TEMP) short-term/long-term temporary knowledge about entities in the environment obtained from observations and interactions with humans. Utilizing SEM and TEMP, COM also involves common sense knowledge to derive simple goals of assistance, advertisement and security. These goals are sent to the planner. (ii) *Perception*: A system for detecting and tracking people using fixed cameras has been implemented. Such a system is able to detect particular events such as entering in a particular area of the shopping mall and sending an advertising message to the robot. (iii) *Interaction interface*: A GUI has been developed allowing a person to interact with the robot and expressing his requests in terms of assistance. The GUI has been realized in order to be fully customizable for different scenarios and different user profiles, allowing for personalized short-term interactions as described in [1]. (iv) *Markov Decision Process and Petri-Net Plans planning*: a task language based on Progressive Reasoning Units (PRUs) has been developed to express all assistance tasks of the robot, an appropriate MDP-based planning algorithm to compute the policy to accomplish the task and PNP-based planning to transform a policy into an execution Petri-Net plan to deal with execution error. Details on the implementation of this framework are provided in [2]. For the development of the software robotics components, we used the Robot Operating System (ROS) (www.ros.org), which is the standard middleware for robotics applications. In particular, we used the last stable version ROS Indigo and the last LTS (Long Term Support) version of the Linux/Ubuntu Operating System.

We then extend this architecture to multi-robot settings by developing the following general principles : let robots $A = \{A_1, A_2, \dots, A_n\}$ receive from different KB modules

a set of goals $G = \{g_1, g_2, \dots, g_k\}$ where goals concern advertisement, patrolling, assisting and escorting. We consider that goals are sorted according to their type assuming that the type conveys some goal's priority. In our case, we consider that security goals have a higher priority than assistance goals and the advertisement goals have the lowest priority.

Let assume that KB modules **communicate** their local generated goals to each other leading to the same set of goals handled by different decision-making modules of different robots (Figure 1). The general principle consists in receiving information from external sensor (external cameras in our case) by communicating these information through the network (wireless). The KB modules receive the same information and thus generate the same list of goals. Each KB sends this list of goals to its decision making module to generate a policy of accomplishing one or many goals.

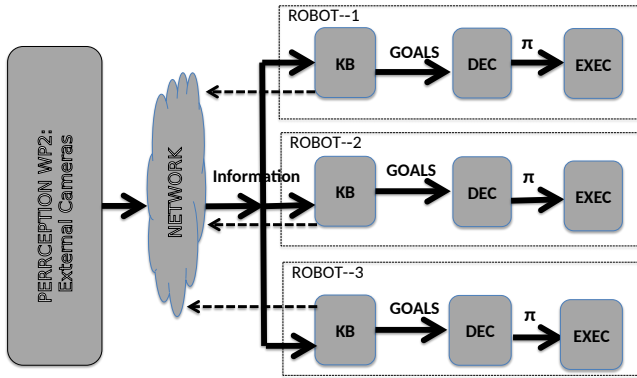


Fig. 1. General architecture of multi-robot decision-making module

B. Communication

KB modules communicate to exchange the information concerning the status of execution and also the level of interruptibility allowing, at the receipt of a list of goals, to consider only robot that could accomplish the goals according to their current status. The list of messages exchanged between different modules (KB, DEC and EXEC) and between robots are : $msg_new_facts_#\#id$ coming from the local perception to the local KB; $msg_new_external_facts_#\#id$ coming from the local KB to the other robots; $msg_end_local_goals_#\#id$ coming from the local EXEC module to the local KB; $msg_end_external_goals_#\#id$ coming from the local KB to the other robots; $msg_selected_goals_#\#id$ coming from the local DEC module to KB local; $msg_selected_external_goals_#\#id$ coming from the local KB to the other robots; $msg_goals_values_#\#idrobot$ coming from the other robots.

The general principal is depicted in Figure 2 where each robot has a local Knowledge base, a local decision maker and a local executor interacting each other and exchanging information with the other robots through a communication infrastructure. In our current case, we consider a direct communication between robots assuming that they evolve in an environment where their ranges of communication

cover all the space as in the mall. Limited communication ranges issues are let to the future work. However, we use a procedure allowing robots to move towards a central point to establish the communication and update their KB.

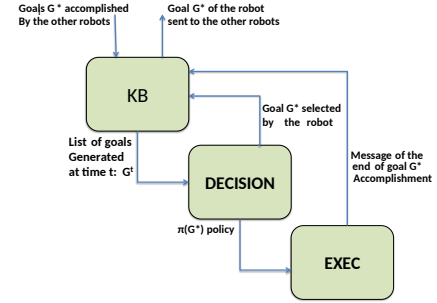


Fig. 2. Communication between KB, decision and execution modules

III. IMAGE AND VIDEO ANALYSIS FOR PERCEPTION

In this section, we describe the adopted techniques for detecting and tracking people from fixed and mobile cameras. Fixed cameras are mounted in the shopping mall to analyze the behavior of the people in order for detecting particular events (e.g. interactions, standing in front of a shop for a while, etc.). Mobile cameras, mounted on the robot, are used for detecting and tracking people in the environment during the navigation of the robot and are useful for approaching people. A detailed description of both the systems is provided in the following sections.

A. Fixed Camera System

The fixed cameras are used for detecting specific events, like entering a specific area of the shopping mall, and sending the coordinates of the detected event to the robot. The system is based on the following steps: (i) Background Subtraction, (ii) Person Detection, (iii) Re-projection on the map, (iv) Tracking, and (vi) Event Detection (see the general scheme in Fig. 3).

In order to reduce the person search space in the image, a background subtraction method has been adopted. In particular the Fastest Adaptive Foreground EXtraction (FAFEX) method of Pennisi *et al.* [3] has been used. The output of FAFEX is a foreground mask containing the blobs, which are the possible candidates to be people. Each blob is classified as *person* or *not a person* by using a Support Vector Machine (SVM) classifier, based on Histograms of Oriented Gradients (HOG) [4], trained on the INRIA Person dataset ¹. Then, the detections are reprojected onto a map of the shopping mall by using a Homography technique. To track the people inside the environment, a tracker, called PTracking [5], has been used. Then, in order to detect events such as entering a particular area of the shopping mall, an event detection module has been developed. By selecting specific areas of the shopping mall, such a module is able to recognize if a person entered one of those areas, which could be used as inputs to the navigation system of the robot.

¹<http://pascal.inrialpes.fr/data/human/>

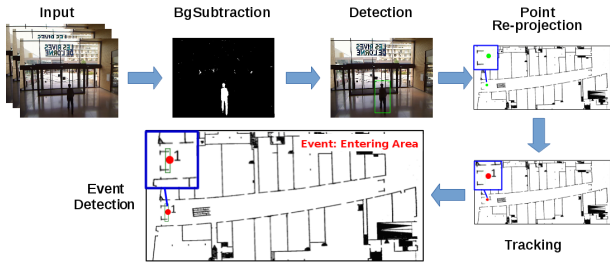


Fig. 3. Fixed Camera System scheme.

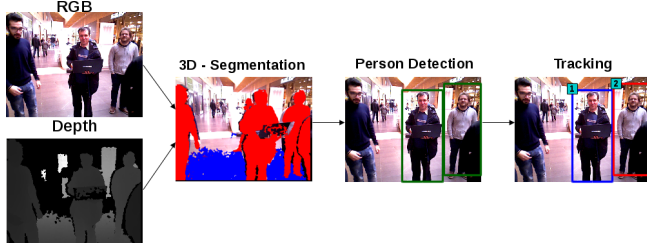


Fig. 4. The 3D Person Detection and Tracking module is based on 3 main steps: 1) 3D Segmentation, 2) Person Detection, and 3) Tracking.

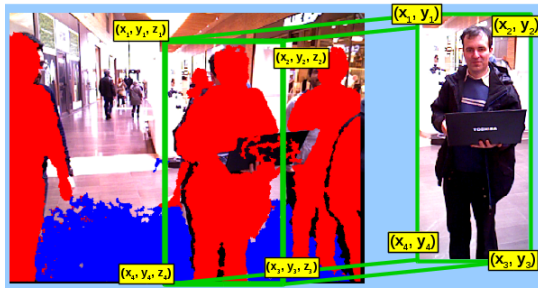


Fig. 5. The bounding box of the 3D cluster is converted in 2D image coordinates for extracting the correspondent RGB patch.

B. Robot Perception

The cameras mounted on the robot are used for detecting and approaching the people inside the shopping mall. To this end, we developed a framework for 3D person detection and tracking framework as shown in Fig. 4. The system is based on three steps: (i) scene segmentation, (ii) person detection, and (iii) person tracking. The robot is equipped with an RGB-D camera (i.e. Asus Xtion), which is mounted on the top of the robot. Thanks to the depth and camera location information, the system is able to straighten the 3D scene and to compute the point cloud. Then, the points under $5cm$ of height are excluded from the cloud assuming that they belong to the floor. The same assumption is made for all the points above $2m$ of height (we assume that a person is tall less than 2 meters). The remaining points are grouped by using a clustering method based on the Euclidean distance [6]. Each cluster is considered as a candidate to be a person. In order to verify this, RGB patches are extracted by converting the 3D bounding box of each cluster in a 2D image bounding box (see Fig. 5).

To recognize if a patch is a person, an approach based on the Aggregate Feature Channels (ACF) [7] has been adopted. The ACF method computes the features as the aggregation of multiple features. In our current implementation we used



Fig. 6. a) Detection: the green bounding boxes represent the detections and b) the numbered box are the tracks.

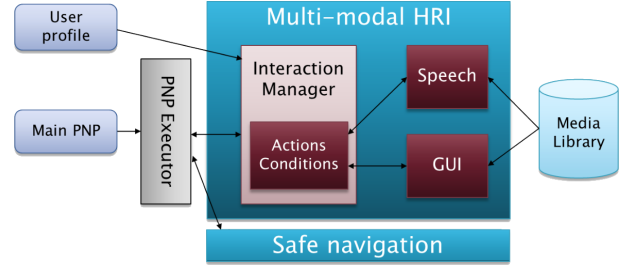


Fig. 7. Multi-modal HRI architecture.

the L, U, and V color components, the HOG features and, the gradient magnitude. All these features are integrated into the Fastest Pedestrian Detector in the West (FPDW) [8]. Then, to classify a single patch, a boosting tree classifier has been trained by using the INRIA person dataset. Such a strategy increases the detector speed maintaining robust detection performance (see Fig. 6a). Finally, a multiple target tracker is used for tracking the detected people. The tracker uses the appearance model of each detection together with a euclidean distance approach for carrying out the data association step. The target appearance is represented by using the combination of the RGB color information and the Speed Up Robust Features (SURF) [9]. Such features are reduced by using a sparse dictionary. Then, an on-line Adaboost classifier, one for each detection, is trained using such a dictionary. A Kalman filter is used for filtering out the noise of each detection (see Fig. 6b). A track is assigned to the current detection if, the confidence of one classifier is the largest possible and is greater than a minimum threshold c (experimentally evaluated), and if the euclidean distance between the related Kalman prediction and the current detection position is less than a predefined threshold e (experimentally evaluated).

Thanks to such a system, the robot is able to track the people inside the environment and to understand if a person wants to approach it by measuring the relative proximity distance.

IV. MULTI-MODAL HUMAN-ROBOT INTERACTION

The interactions realized in the COACHES scenario are characterized by being short in time with many users who are not expert and who have not been trained about the capabilities of the robot. In this context, the use of an HRI system offering multiple modalities of interaction can greatly increase its usability and improve user experience, since the users can choose the modality more comfortable to them.

Figure 7 illustrates the overall architecture of our HRI system. Inputs to the system are a Petri Net Plan (PNP) describing the robot behavior, a user profile and a multi-media library.

The PNP is generated by the reasoning and planning module of the system (as it will be described in next section V) and contains both *robotic* actions (e.g., move, goto) and *interaction* actions. The execution of this plan is managed by the PNP Executor module which derives the execution of each action to the safe navigation component in the case of robotic actions or to the multi-modal HRI component in the case of interaction actions.

The execution of interaction routines is managed by the Interaction Manager (IM). The IM acts as a server that executes the interaction actions when requested by the PNP Executor and returns the input of the user in the form of PNP conditions which are evaluated by the PNP Executor to enable the corresponding transitions to make evolve the course of the PNP.

Currently, the IM manages two components: a C# speech server using the multi-language Microsoft Speech Recognition and Synthesis engine and, a Python GUI on a touch-screen. This allows for the use of the following modalities of interaction: output of information is provided visually in the form of texts or images displayed on the Python GUI or by voice using a Text-to-Speech (TTS) component while the input from the user can be given by using the touch-screen or via spoken commands interpreted by the Automatic Speech Recognition (ASR) system.

V. DISTRIBUTED REASONING, DECISION-MAKING AND EXECUTION

A. Semantic reasoning

Since the relevant knowledge about the environment is heterogeneous (e.g., static/dynamic, spatial/nonspatial), we classify the knowledge bases into three parts: Semantic Map (SEM), Commonsense Knowledge Base (COM), and Temporary Knowledge Base (TEMP). SEM and COM represent the static knowledge about the environment while TEMP represents the temporary knowledge and can be updated due to observations or human-robot interactions.

1) *Knowledge Base for Semantic Map:* We define a semantic map (SEM) for an environment which consists of a set of entities, a set of spatial relations and a set of non-spatial relations. The entities include access points (e.g., doors) of places like stores, restaurants, elevators, escalators, restrooms, etc. The spatial relations include qualitative spatial relations, like “next-to” and “up-down” directions. The nonspatial relations include the names of stores, what kind of objects they sell and wheel-chair accessibility condition. Overall, these two sorts of relations describe static knowledge about the shopping mall.

Entities of a shopping mall can be represented by a set of atoms of the form `entity(entityID)`. **Qualitative spatial relations** of the entities in the environment can be represented by a set of atoms, like `acc(store1, store2)` (to describe accessibility of `store1` from `store2`)

or `dir(store1, store2, next-to)` (to describe relative direction of `store1` from `store2`). **Nonspatial relations** of entities can be represented by atoms, like `name(store1ID, abc)`, `sells(abc, shirts)`, and `hasRamp(store1ID)`. We represent the names of the stores, which product they sell and if they have a ramp for stroller or wheelchair access as nonspatial relations. These nonspatial relations are necessary for computing a personalized path. For example, if a customer with stroller asks “Where can I buy shoes?”, the robot should consider stores which sell shoes and has ramp for stroller access. After finding the possible goal locations, the path finder module computes the path. This computation also requires nonspatial relations, because the path should not include routes without stroller access.

We can represent the entities and qualitative spatial relations as a directed graph. The vertices of the graph denote the entities, whereas the edges denote the accessibility relations of the entities. We can also label the edges with the directionality relations.

2) *Knowledge Base for Commonsense Knowledge:* We define commonsense knowledge base (COM) in a shopping mall with two parts. The first part is about taxonomic relations between entities (e.g., “French restaurant is a restaurant”). We represent these relations as an ontology. The second part is about default knowledge related to shopping malls (e.g., “Children usually desire toys”). Since these relations necessitate nonmonotonicity, we represent them as ASP rules.

Taxonomic knowledge of a shopping mall consists of hierarchies of classes and their relationships. We model the nonspatial relations of entities, and taxonomic commonsense knowledge about these entities as a formal Shopping Mall Ontology. We represent this ontology in OWL (Web Ontology Language) [10], [11], and use DL reasoners, such as PELLET [12], to extract relevant knowledge from the ontology using the query language Sparql [13]. **Default knowledge** about entities in a shopping mall can be represented in Answer Set Programming (ASP) [14]. For instance, we can represent the default commonsense knowledge “Normally, a package belongs to the adult next to it.” by the following rule:

```
belongs(X,P) :- object(X,package), nextTo(X,P),
instanceOf(P,adult), not -belongs(X,P).
```

Similarly, we can represent “Normally, a package is suspicious if it does not belong to anyone.” by a rule in ASP. Then, if the robot sees a package which does not belong to anyone, it can infer (using the ASP solver CLINGO [15]) that the package is suspicious with the commonsense knowledge.

3) *Knowledge Base for Temporary Knowledge:* We define temporary knowledge base (TEMP) in a shopping mall as ASP facts, like `disabled(c1)`, `promotionAt(store1)`, `noAccess(elevator)`, and `interestedToBuy(c1, cosmetic)`. These knowledge can be obtained from observations via perception or from human-robot interactions. The mall manager may tell the robot that the elevator is broken. The robot can recognize

that the customer asking a question is at a wheelchair. The shopkeepers may tell the robot that they have some promotions over the weekend. These temporary knowledge can be represented as follows:

The robot will use this temporary knowledge, in addition to the knowledge bases SEM and COM, to infer a list of possible goals. Whenever a customer asks for a place, we add that place with $\text{goalLoc}(X)$ predicate. Whenever a customer asks for a product, we add that product with $\text{interestedToBuy}(C, X)$ predicate.

B. Distributed decision-making

Once local KB synchronized, each robot computes the value of its optimal policy to accomplish a goal and thus communicates a vector of values to the other robots. Each robot computes the vector values of the goals and receives from the others their vector values. From these exchanged information, each robot maintains a matrix of values of the couple (robot, goal). This global information gathered from local information allows each robot to select the best goal using a distributed market-based auctioning. Indeed, each robot i maintains a matrix M_i^h per goal priority h . Robots concern goals of lower priority when goals of higher priorities are allocated. The matrix is constructed as follows:

- 1) each robot i computes the optimal value V_i^{*,g_l} to accomplish goal g_l . Value vector $(V_i^{*,g_1}, V_i^{*,g_2}, \dots, V_i^{*,g_k})$ represents the values of optimal policies accomplishing goals in the list. This vector represents the line i of the matrix.
- 2) Each robot j sends its value vector to each others, allowing them to complete their matrix
- 3) Each robot i has thus a matrix :

$$M_i^h = \begin{pmatrix} V_1^{*,g_1} & V_1^{*,g_2} & \dots & V_1^{*,g_k} \\ V_2^{*,g_1} & V_2^{*,g_2} & \dots & V_2^{*,g_k} \\ \dots & \dots & \dots & \dots \\ V_n^{*,g_1} & V_n^{*,g_2} & \dots & V_n^{*,g_k} \end{pmatrix}$$

The allocation of goals to the robots is performed by a distributed decision-theoretic market auction, where each robot i computes for each goal g a value $V_i(g)$ of following a policy accomplishing it. The agent α proposing the best value is elected to accomplish this task.

$$(\alpha, g^*) = \underset{A_i, g_k}{\text{argmax}} V_{A_i}(g_k) \quad (1)$$

It's possible there exist many robots α able to accomplish the goal g^* with the same value and thus the equation has a lot of solutions. When many robots optimize the accomplishment of the goal g^* , we allocate the goal to the robot with the minimum regret. The *Regret* of not accomplishing a goal g^* is a loss in value when accomplishing the best goal other than g^* . More formally,

$$\text{regret}_j(g) = V_j^{\pi^*}(g) - \max_{g' \neq g} V_j^{\pi^*}(g')$$

Let S_g be the set of robots α optimizing the value of accomplishing the goal g (solutions of Equation 1), the

best robot to which we allocate the goal g is given by the following equation:

$$\alpha^* = \min_{\alpha \in S_g} \text{regret}_\alpha(g)$$

If this equation has many solutions, we can proceed in the same way with the other goals and so on.

C. Execution monitoring

A crucial feature for deploying robots in public spaces is their ability of reliably executing their plans in presence of uncertainty about the world and the user interactions and of perception noise.

During the COACHES project, we have studied a method for improving robustness of the plan generated by the decision making modules. This processing step (called *Robustification*) during the plan generation phase aims at improving the robustness of the plan to situations that are not modelled in the planning domain.

To this end, we use the Petri Net Plan (PNP) formalism [16] and the plan execution framework is formed by the following elements: 1) Plan translation into PNP [17], 2) Execution Rules (ER) [2], 3) ROS action execution.

Briefly, the policy or the conditional plan generated by the planner is first transformed into a PNP. This is a straightforward procedure since Petri Nets can easily represent trees and DAG. Finally, the actual execution of the robot actions and interactions is performed by the PNP engine. Each action name in PNP is mapped into an action that can be either a robotic action or an interaction action (interpreted by MODIM, as described in Section IV).

VI. EVALUATION METHODOLOGY AND RESULTS ON REAL ROBOTS

A. Evaluation methodology

Evaluation of a complex system like the one developed in the COACHES project requires a specific methodology and setup. While the evaluation of the components presented in the previous sections of this paper is described in the relative papers, here we want to focus on the overall evaluation of the entire system by the actual users (i.e., customers of the shopping mall).

For such a user evaluation, we have designed the experimental protocol that is described in this section. The actual execution of the experiment is planned for July 6-9, 2017 in a public event in the shopping mall Rives de l'Orne in Caen, where we aim at involving around 100 users². The experimental protocol we propose here has been successfully applied in the different scenario of a teaching assistant robot [18], where several COACHES components were used to implement the system.

More specifically, the experimental protocol will be based on the Godspeed Questionnaire Series (GQS) [19], a common evaluation method for HRI. GQS will be used to assess the success of the robot, evaluating the emotional states

²Results of these experiments will thus be ready by mid-July 2017.

of people during the interaction. The questionnaire will be submitted to users (customers and vendors of the shopping mall) characterized by different features (independent variables). The measure of the GQS features (dependent variables) and the consequent statistical analysis will allow for an important user evaluation of the COACHES robots. A sketch of the experimental methodology is the following: (1) Definition of the independent variables (e.g., type of user: customer/vendor, gender, age, task to be executed, type and level of interaction with the robot, etc.); (2) Selection of the users, following a *between-user* approach, each user will participate to a single test; (3) Filling the paper questionnaires by the users before and after the experience with the robot; (4) Statistical analysis of all the collected data.

The output of the statistical analysis will be useful to assess the effectiveness of the COACHES approach in achieving the considered tasks as well as the user feelings about the system.

B. Public demonstration

We ran a demonstration of the two robots collaborating to guide visitors to researchers offices and to various services in our lab with the same tasks as in the mall. The mission was made of 3 layers: first, the robots come to predefined wait-points near entrances, where they offer assistance to visitors. Second, visitors use the touchscreen (fig.8) to ask for services like a specific office. Finally, the robot escorts the visitor (fig. 9.a) to the destination and then returns to a free wait-point. Each step is planned jointly by the two robots to avoid conflicts (fig. 9.b).



Fig. 8. Interaction with visitors



(a)



(b)

Fig. 9. a) Escorting to office; b) Robots coordination

VII. CONCLUSION AND FUTURE WORKS

We presented a practical and novel models for cooperative robots for assistance in public areas by sharing tasks and interacting with people. We developed a framework allowing a fleet of robots to reason and synchronize their local static and dynamic KBs by exchanging appropriate information and to develop an augmented MDP using a value matrix for a marked-based auctioning to better coordinate the robot activities. We also presented a module of perception dedicated

to the people detection and tracking, face detection and 2D escorting with different algorithms and a multi-modal human robot interaction. We developed a software allowing a fleet of robots to assist in a cooperative way a group of peoples by sharing autonomously tasks and maintain interaction with people they assist. A video is available on the web site presenting the overall behavior of the system with two robots at greyc.coaches.fr.

REFERENCES

- [1] L. Iocchi, M. T. Lázaro, L. Jeanpierre, and A.-I. Mouaddib, "Personalized short-term multi-modal interaction for social robots assisting users in shopping malls," in *Int. Conf. on Social Robotics*, Paris, France, Oct 26-30 2015.
- [2] L. Iocchi, L. Jeanpierre, M. T. Lázaro, and A.-I. Mouaddib, "A practical framework for robust decision-theoretic planning and execution for service robots," in *Proc. of Int. Conf. on Automated Planning and Scheduling (ICAPS)*, London, UK, June 12-17 2016, pp. 486-494.
- [3] A. Pennisi, F. Previtali, D. D. Bloisi, and L. Iocchi, "Real-time adaptive background modeling in fast changing conditions," in *12th IEEE Int. Conf. on Advanced Video and Signal Based Surveillance (AVSS)*, 2015, pp. 1-6.
- [4] N. Dalal and B. Triggs, "Histograms of oriented gradients for human detection," in *IEEE Computer Society Conference on Computer Vision and Pattern Recognition - Volume 1 - Volume 01*, 2005, pp. 886-893.
- [5] F. Previtali and L. Iocchi, "Ptracking: Distributed multi-agent multi-object tracking through multi-clustered particle filtering," in *IEEE Int. Conf. on Multisensor Fusion and Integration for Intelligent Systems*, 2015, pp. 110-115.
- [6] R. B. Rusu, "Semantic 3d object maps for everyday manipulation in human living environments," Ph.D. dissertation, Computer Science department, Technische Universitaet Muenchen, Germany, Oct. 2009.
- [7] P. Dollar, R. Appel, S. Belongie, and P. Perona, "Fast feature pyramids for object detection," *IEEE Trans. Pattern Anal. Mach. Intell.*, pp. 1532-1545, 2014.
- [8] P. Dollar, S. Belongie, and P. Perona, "The fastest pedestrian detector in the west," in *Proceedings of the British Machine Vision Conference*, 2010, pp. 68.1-68.11.
- [9] H. Bay, A. Ess, T. Tuytelaars, and L. Van Gool, "Speeded-up robust features (surf)," *Comput. Vis. Image Underst.*, vol. 110, no. 3, pp. 346-359, 2008.
- [10] I. Horrocks, P. F. Patel-Schneider, and F. van Harmelen, "From shiq and rdf to owl: the making of a web ontology language," *J. Web Sem.*, vol. 1, no. 1, pp. 7-26, 2003.
- [11] G. Antoniou and F. van Harmelen, "Web ontology language: Owl," in *Handbook on Ontologies*, 2004, pp. 67-92.
- [12] E. Sirin, B. Parsia, B. C. Grau, A. Kalyanpur, and Y. Katz, "Pellet: A practical owl-dl reasoner," *Web Semantics: Science, Services and Agents on the World Wide Web*, vol. 5, no. 2, pp. 51-53, 2007, software Engineering and the Semantic Web.
- [13] E. Prud'hommeaux, A. Seaborne, et al., "Sparql query language for rdf," *W3C recommendation*, vol. 15, 2008.
- [14] G. Brewka, T. Eiter, and M. Truszczynski, "Answer set programming at a glance," *Commun. ACM*, vol. 54, no. 12, pp. 92-103, 2011.
- [15] M. Gebser, B. Kaufmann, R. Kaminski, M. Ostrowski, T. Schaub, and M. T. Schneider, "Potasso: The potsdam answer set solving collection," *AI Commun.*, vol. 24, no. 2, pp. 107-124, 2011.
- [16] V. A. Ziparo, L. Iocchi, P. U. Lima, D. Nardi, and P. F. Palamara, "Petri net plans - A framework for collaboration and coordination in multi-robot systems," *Autonomous Agents and Multi-Agent Systems*, vol. 23, no. 3, pp. 344-383, 2011.
- [17] V. Sanelli, M. Cashmore, D. Magazzeni, and L. Iocchi, "Short-term human robot interaction through conditional planning and execution," in *Proc. of International Conference on Automated Planning and Scheduling (ICAPS)*, 2017.
- [18] P. Ferrarelli, M. T. Lázaro, and L. Iocchi, "Design of robot teaching assistants through multi-modal human-robot interactions," in *Proc. of International Conference on Robotics in Education (RIE)*, 2017.
- [19] A. Weiss and C. Bartneck, "Meta analysis of the usage of the godspeed questionnaire series," in *2015 24th IEEE International Symposium on Robot and Human Interactive Communication (RO-MAN)*, Aug 2015, pp. 381-388.