Context Aware Robot Architecture, Application to the RoboCup@Home Challenge

Fabrice Jumel, Jacques Saraydaryan, Raphael Leber, Laëtitia Matignon, Eric Lombardi, Christian Wolf, Olivier Simonin

Context Aware Robot Architecture, Application to the RoboCup@Home Challenge

Fabrice Jumel[1,4], Jacques Saraydaryan[1,4], Raphael Leber[1], Laetitia Matignon[3,4,5], Eric Lombardi[5], Christian Wolf[2,4,5], and Olivier Simonin[2,4]

[1]CPE Lyon, [2] INSA Lyon, [3] UCBL Lyon 1 Univ.
[4] CITI Lab., INRIA Chroma, [5] LIRIS Lab., CNRS,
Université de Lyon, Villeurbanne, France

Abstract. This paper presents an architecture dedicated to the orchestration of high level abilities of a humanoid robot, such as a Pepper, which must perform some tasks as the ones proposed in the RoboCup@Home competition. We present the main abilities that a humanoid service robot should provide. We choose to build them based on recent methodologies linked to social navigation and deep learning. We detail the architecture, on how high level abilities are connected with low level sub-functions. Finally we present first experimental results with a Pepper humanoid.

1 Introduction

Most of robotic architectures focus on a specific ability: autonomous navigation, grasping, human-robot interaction, etc. Today, more and more applications and professional or public contexts require a robot to be able to perform several tasks with a high level of abstraction (help a human, execute a human order, serve humans). While AI[1] methods and technologies progress, expectations about service robotics are growing. This is particularly true in complex human-populated environments [1]. In this context we aim to define software architectures that orchestrate and combine different high level abilities and their sub-functions.

Each high-level ability is generally built on sub-functions, that we call also low level tasks or controls. We can mention for instance path-planning computation, SLAM[2] function, object and human detection, or speech output. In this paper we present an architecture which is able to perform high level tasks that require several skills: navigation in dynamic - human-populated - environments, object detection, people detection and recognition, speech recognition, and task planning. Interacting with humans and answering to their orders is a very large and challenging problem. In this paper, as a first step, we limit our study to the high-level tasks proposed in the RoboCup@Home competition, by focusing on social navigation. These tasks are presented in the next section.

Our approach distinguishes two main levels. The first one is the "decision" level which selects the main high level task to do, and potentially in parallel.

[1] Artificial Intelligence
[2] Simultaneous Localization and Mapping

The second level is defined as a set of general tasks which are required in several high level tasks. We organize this level in a limited set of generic tasks, which are themselves built on sub-functions that can be shared.

The paper is organized as follows. Section 2 presents the context of the RoboCup@Home challenge and its relative tasks. We also discuss existing works on robot software architectures. Then in Section 3 we give an overview of the proposed architecture, before to detail in Section 4 how the General Behavior Manager orchestrates the execution and selection of tasks. Section 5 details the navigation strategy selection and Section 6 gives on overview of methodologies used for human detection and interaction. Section 7 illustrates in details the approach on a complex scenario, then Section 8 presents first results in the RoboCup@Home challenge. Finally Section 9 concludes the paper.

2 Service robots and software architectures

We aim at developping software architectures and AI modules allowing mobile/humanoid robots to navigate in human-populated environments and to carry out high-level tasks. In this context, we focus on tasks proposed in the RoboCup@Home challenge[3], which requires to develop and to connect different modules providing image analysis, decision making and learning, human-robot interaction, and autonomous navigation. These modules must be orchestrated in a whole architecture.

The scenarios driving our research concern mainly tour guide robots [2] and waiter robots [3] which are variants of tasks proposed in the Robucup@Home competition. For example, the cocktail party challenge of the RoboCup@Home [4] consists for a robot to take orders during a private party in a flat. In order to score, the robot has to find the room, ask guests about orders and give them to the bartender located in another room. In order to fulfill this scenario, navigation skill and human-robot interaction skill (principally speech interaction) are required. Other tasks are aimed, that are optional part of the challenge:

- Navigate inside the cocktail party room to find calling guests to take order
- Find a more discreet guest to take order
- Find a missing order (drink) on a table or equivalent (object recognition)
- Find back the guest concerned by the missing order (person recognition)
- Be able to describe the guests to the bartender, so that it can correctly distribute the orders in the cocktail party room (person description).

A major issue in the design of a robot architecture is to allow the robot to fulfill high level objectives while dealing with a complex and dynamic environment. More precisely, the robot has to simultaneously sense the environment, navigate and detect people to interact, and so on. This requires to define dedicated architectures able to orchestrate several tasks, mostly in parallel. Such an issue requires to build behaviors over a middleware embedded in the robot. In

[3] http://www.robocupathome.org/

Calisi et al. [5] authors showed that a lot of frameworks have been proposed for robotics software development. More recently, Chitic et al. [6] have underlined that ROS has become a classic middleware to support such a development, with a rich library of functions. Other middlewares, as [7], focus on tasks description language and life cycles. To build a generic architecture, we have chosen ROS as middleware: we inherit the publish-subscribe way of communication, as well as services and actions, and the manager of concurrent processes (or nodes). Our objective is to propose a generic architecture, able to deal with high level tasks, ie. not focusing on a specific task, scenario, or robot. Concerning software architectures dedicated to humanoids, there are, up to now, few generic solutions to manage service tasks. We can mention the work of Natale et al. [8] dedicated to the iCub, where the architecture is turned towards code generation, motor control, and interfaces.

3 Overview of the LyonTech architecture

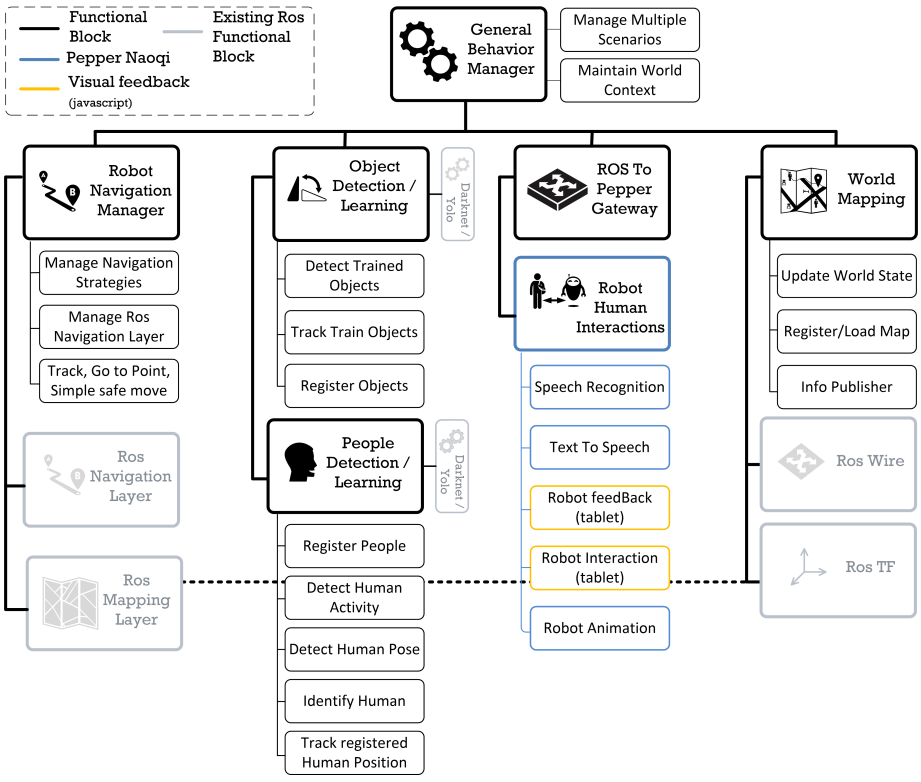

Fig. 1. LyonTech Software Architecture Overview

(a) (b)

Fig. 2. Our recent visual perception work: (a) gesture recognition [11]; (b) activity recognition [12]

In this section we present the embedded AI software architecture of LyonTech's team, aiming at managing both high level decisions and low level tasks of Pepper. Figure 1 gives a general view of the proposal. It contains modules which have been developed in different research work of the consortium, completed by off-the-shelf modules which tackle standard tasks, as well as engineering bricks interconnecting these modules.

On top, the **General Behavior Manager** works like an orchestrator and gives orders to other blocks. Their combination allows to achieve high level tasks, as detailed later. The architecture is organized in four principal functionality blocks (cf. Figure 1), that we detail below.

The **Robot Navigation Manager** is in charge of localizing the robot and allowing dynamic navigation (obstacles avoidance). It uses two sub-functions which are the ROS Navigation layer and the ROS Mapping layer (ie. provided by ROS).

Perception of the environment is performed by the **Object Detection and learning module**, based on deep neural networks [9] and off-the-shelf modules like YOLO-2 [10], which have been retrained and fine-tuned on additional data which we collected specifically for the targeted tasks. Labeled object positions are provided to other blocks. In addition, detailed human information such as human activity, posture, and identity is provided by the complementary People Detection Learning module.

All human-robot interactions are managed by the **Robot Human Interaction** block embedded in the robot. The robot also maintains a knowledge database about its environment (humans, objects, and points of interest positions).

The last module, called the **World Mapping**, is in charge of mapping the environment, including also semantic information, and relies on the ROS Mapping layer shared with the Robot Navigation Manager.

These modules are built on previous team's work, which can be summarized in four topics:

Perception: Our computer vision module brings knowledge in gesture recognition [11] and activity recognition [13], see illustrations in Fig. 2. These

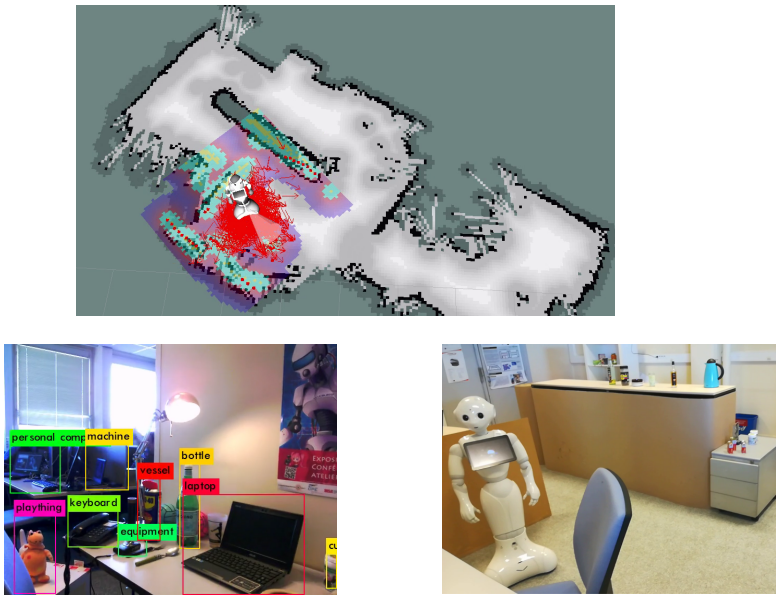

Fig. 3. Example of integration scenario including navigation and object recognition.

methods are capable of running on real time and have been integrated in our platforms of mobile robots. Combined with object detection [10], this allows us to be aware of the objects present in a room, their locations, as well as the ongoing activities in this room (see illustration in Figure 3).

Motion planning and Decision making: This module aims to deal with motion planning in dynamic and uncertain environments, 2D mapping, and 2D localization. We combine our work on autonomous navigation in crowded environments (human-aware navigation) [14] with service delivery strategies [3].

Human-Robot Interactions: We have been working for years on different interactions with robots (from teleoperation to multi-robot orchestration [15, 16]). The Naoqi SDK, provided by Softbank Robotics with the Pepper robot, gives a set of API that are mainly used for Robot and Human interactions (speech recognition, text to speech, robot behavior feedbacks). In order to highlight the robot activity, the Pepper tablet gives visual feedbacks (javascript framework). In the context of the RoboCup@Home we use these different functionalities to give a user friendly interaction with robots including Animations, Tablet, and Dialogues.

Integration: All components are integrated using the ROS middleware, which is the base system of our robot. ROS offers the ability to connect a set of programs through synchronous and asynchronous communication. Moreover, by allowing a set of heterogeneous components (probes, actuators, services) to communicate in a normalized way, processing program can be reused. Figure 3 illustrates this work and its implementation with a Pepper robot.

4 General Behavior Manager

The General Behavior Manager acts as an orchestrator of high level tasks. Four main tasks are executed in parallel : Perception Task, Navigation Task, HRI Task, and General Manager Task.

The General Manager task is in charge of loading a scenario description and managing high level call of Perception, Navigation, and HRI tasks. Each task call could interact synchronously or asynchronously with the General Manager Task. This interaction is based on action as defined in ROS Architecture and leads to a task state (pending, success, failure). The scenario description defines the sequence and conditions of task calls including task timeout, multi-conditional success, and target of the task (navigation point, dialogues, object to find, ...).

With the help of given ontologies, when a task failure occurs, the general manager attempts to change the task target before retrying. As mentioned in [17], ontologies can be used to select objects sharing same properties/usages. As example, if the robot was asked to find a coke bottle, and if no one was found, the general manager could provide the alternative target "fizzy drink" (considered as parent class of coke) to the perception task.

5 Navigation Strategy Selection

an

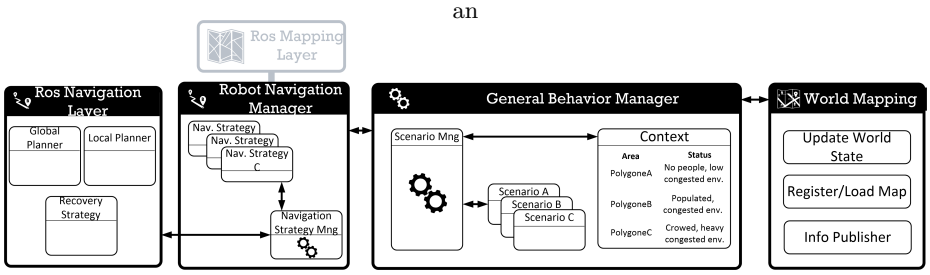

Fig. 4. Navigation Strategy selection

The main objectives of a social aware robot navigation is first to be safe for people around him, to be safe for itself, and then to navigate as quickly as possible while respecting social conventions. The targeted environment can be composed of dynamic obstacles, populated and congested areas. To deal with such constraints, we provide our robot with the ability to select a navigation strategy compliant with the current environment. To optimize robot navigation, we propose to maintain a **context of the robot environment**. To do so we distinguish two main dynamic parameters, the human density in the environment and the environment complexity (level of congestion). On one hand, with the help of the built map, we are able to compute the obstacle density of an area.

Once the area delimitation is done, a percentage of obstacle pixel in this area is computed. With the combination of the color distribution entropy [18] we can determine if an area contains a lot of obstacles and how they are distributed in the targeted area. The higher the obstacle density and dispersion is, the higher the probability to freeze the robot during navigation is. On the other hand, the world mapping function helps to maintain the observed human locations into a map. The area delimitation is then provided by segmentation following the approach of [19].

As shown in Figure 4, maintaining different contexts into a map allows the general manager to ask the navigation manager to select a navigation strategy according to the goal to achieve and the context of the environment. The Navigation manager holds a set of navigation strategies. Each navigation strategy is focused on different objectives:

- optimize the robot navigation to go as quickly as possible to the objective, while being adapted to the environment.
- ensure a safe navigation including recovery mode in case of freezing (eg. replay last command before robot freeze).
- optimize the robot navigation into populated environment (eg. by following the human flow [14]).

When the General Behavior Manager needs a Navigation Task, a first path is computed from the robot actual position A to the goal G. This path, $A \rightarrow G$ is computed by the ROS navigation task service (make_plan service). Once the path is computed, the General Behavior Management get Context information and associated areas. Regarding to the path $A \rightarrow G$, crossed areas are considered and become sub navigation goals S_i. Then the General Behavior Management ask for a navigation task according to the appropriated navigation strategy π_i for each sub goal until the goal is reached: $A \rightarrow S_0, \ldots, S_{i-1} \rightarrow S_i, \ldots, S_{n-1} \rightarrow G$.

6 Human Detection and Interaction

The interaction of the robot with close people requires that the robot can collect information about humans. To do so we build interaction abilities on different functionalities which are finally integrated (see Figure 5 Human Detection/Learning block):

- Human Detection: with the huge increase of the efficiency of Deep learning based tools, detection of objects and people is quickly carried out using standard RGB camera. For this purpose, we integrate the well known **Yolo-2 model** [10] (in its original C++ "*darknet*" implementation) in order to get estimates of bounding boxes centered on humans in the visual field.
- Human Identification (face detection): to allow interaction with human, it is also required to recognize already met people. To tackle this problem, we adapt a module of face recognition to our system. We use the method provided by Adam Geitgey's library[4].

[4] https://github.com/ageitgey/face_recognition

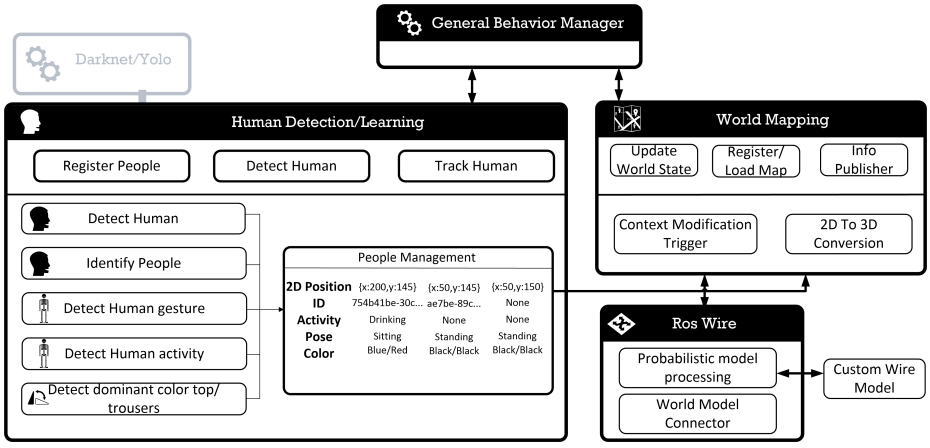

Fig. 5. People Perception processing organization

- Human pose: articulated pose provides important information for interaction with humans. The openpose tools [5] are used to extract information about the skeleton of each person, i.e. a set of body key point positions. In addition we developed a function to determine main people poses like standing, sitting, lying and hand call, by calculating the different key points relative positions. Thanks to camera settings and pose, combined with human 2D normalized data, we can estimate partial 3D pose of people. Therefore we can improve the posture detection, estimate the depth, and discern the different people groups.
- Human dominant color top/trousers: all different clues are important to discriminate people. Using openpose and a dominant color detection we can detect main colors of specific body position if available (we developed a ROS node including a HSV color transformation and a K-mean color clustering).
- Human activities: we recognize complex human activities with our own method, recently introduced, and which obtains state of the art performance on standard datasets [13]. This method was specifically designed for robot applications: it is capable of running in near time and requires only standard sensors, i.e. a single RGB camera, while at the same time outperforming other methods, which use multiple modalities.

An additional process is required to track all these data/encountered humans. To do so we customize the world state estimator of detected object wire[6]. Wire provides a toolset to track various object attributes during the time. Object location, color and label can be tracked through various models including Kalman Filter (e.g for position attribute), uniform distribution. All these tools include a model of uncertainty to be robust in real situation. For this purpose,

[5] https://github.com/CMU-Perceptual-Computing-Lab/openpose
[6] http://wiki.ros.org/wire

we customize world object model of wire to track human position according to a defined Kalman filter and all additional labels (such as id, pose, activity, and dominant color) through uniform distribution models.

With the help of this information, the General Manager updates the context map. The people density of a given area could be computed and the navigation strategies change according this update.

7 Step by Step Scenario Execution

In order to illustrate the behavior of our architecture, let's see the execution of an example of a general purpose robot scenario. The following scenario is aimed: after the operator is found by the robot, the operator asks the robot to go to the kitchen, to find Tom, and to say him a joke. First of all, with the existing map, the World Mapping block provides to the General Behavior Manager a Context including different room polygons with associated congestion level and people density (0.5 by default).
The General Behavior Manager starts the 4 main tasks: Perception (using the Object Detection/learning and People Detection/Learning blocks), Navigation, HRI, and **General Manager Task** (**GMT**).
The **GMT** loads the targeted scenario, including sequences of executions and task controls. The execution of this scenario is presented in table 1. In this table, gray rows refer to actions executed by the **GMT** itself and other rows to task calls. Concerning the columns, the id refers to the sequence number of the operation, the operation column to the type of general manager execution, the task type column gives the name of the task called, the mode column can take two values synchronous (Sync.) or asynchronous (ASync.). In the target column, the inputs of the task call is set. Finally, The success conditions are defined by the column of the same name.
After waiting the door is opened (1), the **GMT** computes sub goals and get associated strategy to go to the living room (2). For each sub goal (3), the Navigation Task is called to go to the current sub-goal S_i applying the associated navigation strategy π_i. When all Navigation Task succeeded, a Perception Task is called to find a people (4). The **GMT** waits until a people is detected. After that, it is asked to the operator to explain the mission to do (5). When achieved, the **GMT** transforms the given information into a set of actions (6). After recomputing sub goals (7), the robot is asked to detect human around (8) and to go to the kitchen (9). Note that during the robot navigation, human information is collected and transmitted to the World Management Block. When the destination is reached, the **GMT** calls the World Management database to see if there is a human in the kitchen (10). The robot asks to the human confirmation about it's identity (11). However, it is not Tom. The **GMT** tries to find a new valid target by changing the level of abstract of Tom resulting into "Human" (12). After checking again if a human is still there (13), the robot tells a joke (14).

Table 1. Operation list of the General Manager (gray rows are GMT actions, others are task calls)

id.	Operation	Task Type	Mode	Target	Success Condition
1	Loop until Success	Perception	Sync.	Opened Door	State=success
2	Make Path to living Room, Compute sub. goals S_i, get navigation strategy π_i				
3	For all S_i	Navigation	Sync.	S_i, π_i	State=success
4	Loop until Success	Perception	Sync.	Human	State=success
5	Wait Success	HRI	Sync.	Ask for Goal	State=success
6	Compute task list from results of task.id=5				
7	Make Path to Kitchen, Compute sub. goal S_i, get Navigation strategy π_i				
8	Loop until Success of task.id=9	Perception	ASync.	Human	State=success
9	For all S_i	Navigation	Sync.	S_i, π_i	State=success
10	Call world management and check if human is in the kitchen				
11	Wait Task End	HRI	Sync.	Ask for Name	Result=Tom
12	In case of Failure get new HRI Task Target = Human				
13	Call world management and check if human is in the kitchen				
14	Wait Task End	HRI	Sync.	Tell a joke	State=Success

8 Experimental Evaluation

In order to experiment our architecture with a Pepper humanoid, we prepared Pepper to be controlled by a PC with ROS. The gateway between ROS and the Pepper robot has been customized and allow to gather Pepper sensor information (RGBD data, laser information, odometry,..). The ROS Pepper navigation task has been done with the ability to map the environment and avoid obstacles. After comparing two object/people detection techniques (tensorflow Single Shot Multibox Detector SSD and MobileNet, Darknet Yolo 2), the Darknet framework with the Yolo 2 model gives currently better results concerning object detection.

Interaction between ROS and Naoqi Dialogue System has been made allowing to test a simple general purpose robot service scenario: detect a people, ask for a task, understand task "Find Maria", navigate to search people, and identify the targeted people.

Preliminary results with these tasks can be shown in the video[7]. In this video, after introducing our research work on perception and navigation, a first

[7] `https://www.youtube.com/watch?v=TPTh_KjVUJQ`

experimental result with this scenario is shown. On the left side, the Robot navigation is presented including map representation and simple object detection from the robot side view. On the right side, results of object and people detection are shown thanks to the Darknet framework and Yolo 2 model. Finally, in the last part of the video, a simple general purpose robot service is presented.

9 Conclusion

In this paper, we offered an overview of the architecture developped by the LyonTech team to target the SSPL RoboCup@Home competition with a Pepper humanoid. After introducing the different AI modules we developed, we presented the architecture which allows to organize them. Then we detailed the navigation strategy selection and the human detection blocks. Finally, we introduced first experimental results in a robot service scenario, including detection of people, navigation and search for a person. The video presented in Section 8 illustrates the ability of the Pepper to manage such high level tasks. The next step of this work is the evaluation of the architecture during the Social Standard Platform League (SSPL) competitions in Montreal, June 2018. This will be helpful to analyze and improve the general orchestration and each sub-function.

References

1. L. Iocchi, E. Erdem, L. Jeanpierre, A. Mouaddib, and H. Sahli. COACHES Cooperative Autonomous Robots in Complex and Human Populated Environments. In *14th Conf. Italian Association for AI. IAxAI*, Ferrara, Italy, 2015.
2. W. Burgard, A. B. Cremers, D. Fox, D. Hähnel, G. Lakemeyer, D. Schulz, W. Steiner, and S. Thrun. The interactive museum tour-guide robot. In *Proc. Fifteenth National/Tenth Conference on Artificial Intelligence/Innovative Applications of Artificial Intelligence*, AAAI '98/IAAI '98, pages 11–18, Menlo Park, CA, USA, 1998. American Association for Artificial Intelligence.
3. J. Saraydaryan, F. Jumel, and O. Simonin. Robots delivering services to moving people : Individual vs. group patrolling strategies. In *IEEE ARSO, IEEE International Workshop on Advanced Robotics and its Social Impacts*, 2015.
4. L. van Beek, D. Holz, M. Matamoros, C. Rascon, and S. Wachsmuth. Robocup@home 2018: Rules and regulations. `http://www.robocupathome.org/rules/2018_rulebook.pdf`, 2018.
5. D. Calisi, A. Censi, L. Iocchi, and D. Nardi. Design choices for modular and flexible robotic software development: the openrdk viewpoint, 2012.
6. S. Chitic, J. Ponge, and O. Simonin. Are middlewares ready for multi-robots systems? In *International Conference on Simulation, Modeling, and Programming for Autonomous Robots*, Bergamo, Italy, October 2014.
7. I. Lutkebohle, R. Philippsen, V. Pradeep, E. Marder-Eppstein, and S. Wachsmuth. Generic middleware support for coordinating robot software components: The task-state-pattern. *Journal of Software Engineering for Robotics*, 2, 2011.
8. L. Natale, A. Paikan, M. Randazzo, and D. Domenichelli. The icub software architecture: Evolution and lessons learned. *Frontiers in Robotics and AI*, 3:24, 2016.

9. B. Moysset, J. Louradour, and C. Wolf. Learning to detect and localize many objects from few examples. *Pre-print: arxiv:1611.05664*, 2016.
10. J. Redmon and A. Farhadi. Yolo9000: Better, faster, stronger. In *CVPR, IEEE Conference on Computer Vision and Pattern Recognition*, 2017.
11. N. Neverova, C. Wolf, G.W. Taylor, and F. Nebout. Moddrop: adaptive multi-modal gesture recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, 38(8):1692–1706, 2016.
12. F. Baradel, C. Wolf, and J. Mille. Pose-conditioned spatio-temporal attention for human action recognition. *Pre-print: arxiv:1703.10106*, 2017.
13. F. Baradel, C. Wolf, J. Mille, and G.W. Taylor. Glimpse clouds: Human activity recognition from unstructured feature points. In *CVPR, IEEE Conference on Computer Vision and Pattern Recognition*, 2018.
14. F. Jumel, J. Saraydaryan, and O. Simonin. Mapping likelihood of encountering humans: application to path planning in crowded environment. In *The European Conference on Mobile Robotics* , ECMR 2017, Paris, France, 2017.
15. L. Sevrin, N. Noury, N. Abouchi, F. Jumel, B. Massot, and J. Saraydaryan. Preliminary results on algorithms for multi-kinect trajectory fusion in a living lab. *IRBM, Innovation and Research in BioMedical engineering*, 36, Issue: 6:361–366 Special Issue: SI, 2015.
16. E. Nauer A. Cordier, E. Gaillard. Man-machine collaboration to acquire adaptation knowledge for a case-based reasoning system. In ACM DL, editor, *WWW 2012, 21st Int. Conference on World Wide Web - SWCS'12 Workshop, Semantic Web Collaborative Spaces*, pages 1113–1120, Lyon, France, April 2012. ACM.
17. L. Sommaruga, A. Perri, and F. Furfari. Domoml-env: an ontology for human home interaction. In *Proc. SWAP 2005, the 2nd Italian Semantic Web Workshop, Trento, Italy, December 14-16, 2005, CEUR Workshop Proceedings*, 2005.
18. J. Sun, X. Zhang, J. Cui, and L. Zhou. Image retrieval based on color distribution entropy. *Pattern Recogn. Lett.*, 27(10):1122–1126, July 2006.
19. M. Mielle, M. Magnusson, and A.J. Lilienthal. A method to segment maps from different modalities using free space layout - MAORIS : Map of ripples segmentation. *CoRR*, abs/1709.09899, 2017.