RoboBreizh
Video Description Paper
2020 RoboCup@Home Education Online Challenge, Standard Platform

A. Legeleux, M. Neau, A. Dizet, C. Le Bono, and C. Buche

Lab-STICC CNRS UMR 6285, France

Team website : `www.enib.fr/~robobreizh`
Qualification video : `https://youtu.be/dWZ7mA_Ev4E`

Abstract. RoboBreizh is a French RoboCup@Home team of the Lab-STICC laboratory. This year, we aim to participate in our first competition. We developed an architecture to solve the tasks of 2020 RoboCup@Home Online Education Challenge. The Qibullet simulator was used to test and enhance our features before using them on a robot. Based on a ROS architecture, we produced several modules linked to each other. Those modules allow Pepper to understand, act and adapt itself in a local environment. Our research focuses on the environment's perception and motion learning.

1 Introduction

The RoboBreizh team belongs to the French Lab-STICC laboratory. The Lab-STICC's areas of research include Human-Robot Interaction and Artificial Intelligence. Our team focuses on environment's perception, robot's motion and robot's navigation. This is our first participation in a robotics competition. We use ROS to communicate with the robot and Qibullet to simulate a virtual situation including robot's behaviour. Every ROS module tackles a different challenge. The motion module allows Pepper to learn and generalize movements from human demonstrations by machine learning algorithms. The perception module uses YOLOV3 and OpenPose to respectively detect objects in the environment and perform pose estimation. The navigation module builds a world map using Pepper's cameras. The localization module computes a person's position by using the results from the perception and Pepper's depth camera. By combining these modules together we propose a solution to the competition's tasks of 2020 RoboCup@Home Online Education Challenge.

2 Architecture and environment

As highlighted in Figure 1, our architecture uses ROS for its flexibility and the ability to easily segment our code into modules as following :

- Movement : GMM / GMR
- Object detection : YOLOV3
- Person detection : YOLOV3 / OpenCV
- Pose estimation : OpenPose
- Mapping : Gmapping ROS node
- Navigation : ROS Navigation Stack
- Dialog & interactions : Naoqi Python SDK

We decided to use Qibullet to test all this modules in a virtual environment [1]. This way, we could achieve more robust features while limiting the risks on our robots. QiBullet is a software based on PyBullet [2] and provided by SoftbanksRobotics to emulate Pepper. A common alternative to Qibullet, while working with ROS and Pepper, is Gazebo. However, we preferred the former as the physics engine is more realistic. Moreover, the QiBullet official release comes with a ROS wrapper. This wrapper makes it possible to use the same code in simulation and on a real robot. Object detection and pose estimation are performed on a separate computer as they are too costly for Pepper. Pepper sends images to the server via HTTP request when required.

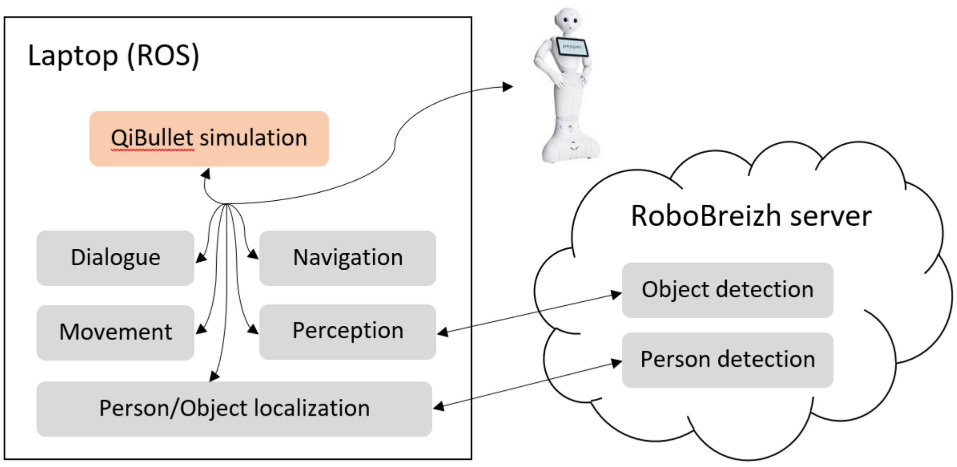

Fig. 1. Architecture of our contribution

3 Contribution

3.1 Perception

The images obtained from Pepper's cameras are sent to the server and fed to YOLOV3 [3] and OpenPose [4]. The YOLOV3 neural network was trained for the detection of bags, chairs and persons. By using the positions of the chairs and persons in an image and how their bounding boxes overlap, we determine whether the chairs are available or taken. OpenPose is used to extract the positions of all the hands in an image and thus whether someone is waving. Our perception modules were built to work in simulation as well as in real-life as shown in Figure 2 a and 2 b. The server then returns the positions of all the objects, persons, and waving hands detected in an image. Those positions are used to measure the objects' distances to the robot.

3.2 People Localization

The distance between humans and Pepper are obtained by combining the human detection (Section 3.1) and Pepper's depth camera values. This distance is the measured depth corresponding to the center of the person's bounding box (Figure 2 c). The position of each detected human in our mapped environment is then computed using trivial mathematical equations. The current application is designed for the localization of humans, but can be set up for any object known through supervised learning.

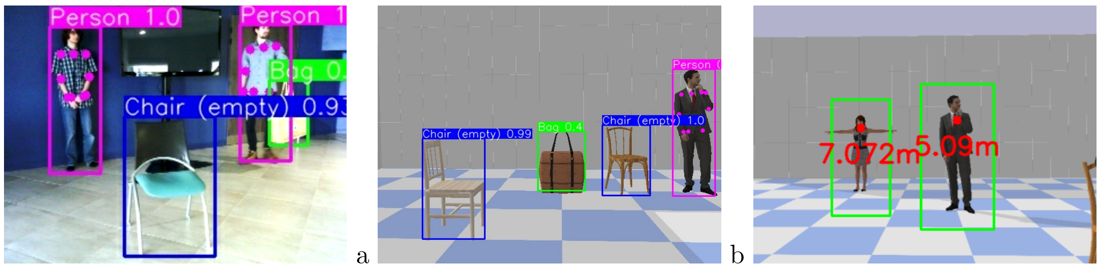

Fig. 2. Object detection in real (a) and simulation (b). Human detection and distance computation (c)

3.3 Navigation

Our approach for navigation focuses on a known environment, according to the tasks description of the competition. To perform this, we mapped multiple environments using Pepper's RGB-D camera. Data from this depth sensor are first converted in point cloud and then in laser data to feed the ROS mapping node. The reason we do not use Pepper's lasers for mapping is that they are not accurate enough to build a detailed map as shown in Figure 3. Then, we used AMCL (Adaptive Monte Carlo Localization) algorithm for localization in the map. Finally, pathfinding and navigation are performed through the ROS Navigation Stack [5] which builds a local and global costmap to avoid obstacles. A diagrammatic representation of this architecture is shown in Figure 4.

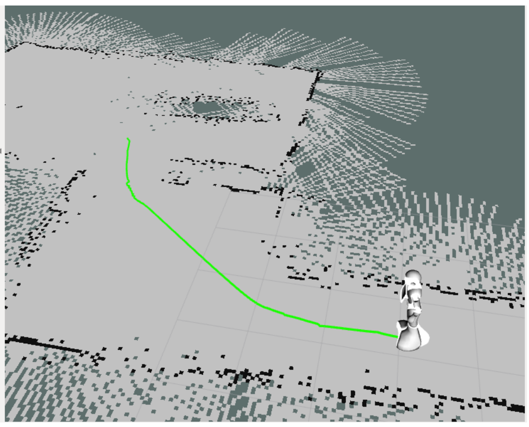

Fig. 3. RVIZ display of navigation node

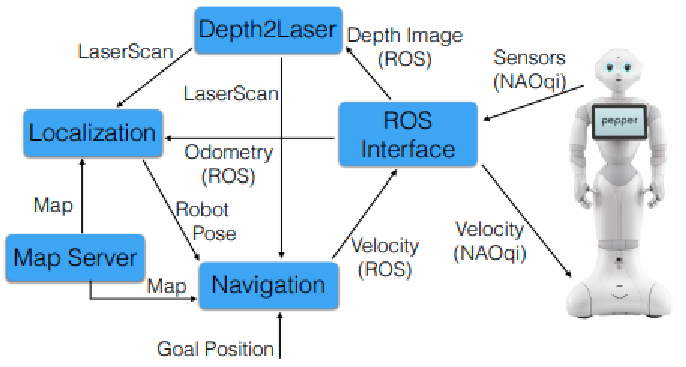

Fig. 4. ROS nodes and their communication through the Navigation stack [6]

3.4 Movement learning

One of the easiest way to teach a movement to a robot is to use learning by demonstrations as shown in Figure 5. We use kinesthetic demonstrations where a human move the robot's arms. We implement Gaussian Mixture Model (GMM) and Gaussian Mixture Regression (GMR) [7][8]. These probabilistic models generalize a movement from multiple demonstrations. Each demonstration is aligned temporally with the first demonstration. The initialization of the means is done with K-means algorithm and the selection of the number of Gaussians with the BIC score. This module was written to run with various robots.

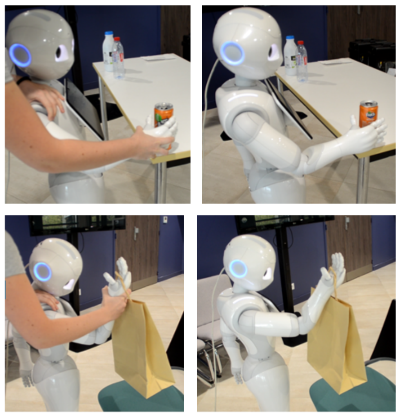

Fig. 5. Learning by demonstration

4 Conclusion

This paper describes the RoboBreizh's team approach to the RoboCup@Home Education challenge. We propose a ROS architecture to handle the competition's tasks using a Pepper robot. Notably, our robot is able to move to a destination, point to an empty seat, take a bag in hand, compute a person's position, talk with someone and detect a waving hand. Our proposed architecture is also flexible and can be easily implemented on other robots. We are still working on additional features to enhance Pepper and wish to increase its usability in person-friendly environments.

References

1. Maxime Busy and Maxime Caniot. qibullet, a bullet-based simulator for the pepper and nao robots. *arXiv preprint arXiv:1909.00779*, 2019.
2. Erwin Coumans and Yunfei Bai. Pybullet, a python module for physics simulation for games, robotics and machine learning. `http://pybullet.org`, 2016–2019.
3. Joseph Redmon and Ali Farhadi. Yolov3: An incremental improvement, 2018.
4. Zhe Cao, Gines Hidalgo, Tomas Simon, Shih-En Wei, and Yaser Sheikh. Openpose: Realtime multi-person 2d pose estimation using part affinity fields. *CoRR*, abs/1812.08008, 2018.
5. Rodrigo Longhi Guimarães, André Schneider de Oliveira, João Alberto Fabro, Thiago Becker, and Vinícius Amilgar Brenner. Ros navigation: Concepts and tutorial. In *Robot Operating System (ROS)*, pages 121–160. Springer, 2016.
6. Vittorio Perera, Tiago Pereira, Jonathan Connell, and Manuela M. Veloso. Setting up pepper for autonomous navigation and personalized interaction with users. *CoRR*, abs/1704.04797, 2017.
7. S. Calinon". *Robot Programming by Demonstration: A Probabilistic Approach.* EPFL/CRC Press, 2009.
8. Maria Kyrarini, Muhammad Abdul Haseeb, Danijela Ristić-Durrant, and Axel Gräser. Robot learning of industrial assembly task via human demonstrations. *Autonomous Robots*, 43(1):239–257, 2019.