CATIE Robotics @Home 2023 Team Description Paper

Sébastien DELPEUCH, Boris ALBAR, Alban CHAUVEL, Christine JAUREGUIBERRY, Clément LAIGLE, Sébastien GAMARDES, Clément PINET, Stéphane POUYET, Logan SAINT-GERMAIN, Florian LARRUE, Ludwig JOFFROY, and Sébastien LOTY

Centre Aquitain des Technologies de l'Information et Électroniques (CATIE)
1 Avenue du Dr Albert Schweitzer, 33400 Talence, France
s.delpeuch@catie.fr
https://robotics.catie.fr/

Abstract. This paper provides an overview of the CATIE Robotics Team's activities for participating at the RoboCup@Home 2023 in Bordeaux. A TIAGo[1] called Epock is used as the main robotic platform. Safe and autonomous navigation is achieved by integrating ROS compatible components. Macarena, an offline tool developed by CATIE using a Nvidia conformal model and an open source nlu from snips, is used for speech recognition and natural language understanding. State-of-the-art neural networks are used for face recognition, person tracking and object detection. Problem-specific developments are successfully tested for object localization. A human-centered approach is used to enhance Epock's interactions. An environment perception-based grasping pipeline using octomaps is implemented. This document describes selected components, shares some feedback and discusses possible improvements.

1 Introduction

The CATIE Robotics Team was formed at the beginning of 2018 and is part of CATIE, a RTO (Research and Technology Organization). CATIE is a non-profit organization supported by the Nouvelle-Aquitaine French region whose mission is to assist companies willing to adopt and integrate digital technologies in their technological and economic development. By creating a RoboCup@Home team, our ambition is to be part of an expert community, nurture our robotic knowledge and share it to foster progress towards a tangible goal. The competition offers an objective benchmark to measure progress and is a stimulating means to unite efforts locally.

In 2018, we focused on integrating proven technologies and achieving simple but robust behaviors in key fields such as safe navigation, Simultaneous Localization And Mapping (SLAM), force based grasping, person and object recognition. TIAGo, an off-the-shelf service robot, was chosen as our main development platform and Robot Operating System (ROS) as our middleware framework. This

[1] https://tiago.pal-robotics.com/

approach worked well in our first international competitions: we ranked second in GermanOpen 2019 RoboCup@Home[2], third in Sydney 2019 RoboCup@Home and second in SciRoc Episode 7 challenge. As we enter year four, our goal is to achieve high-level, integrated, robust, modular, and safe behaviors for our robot. We are currently working on improving our MoveIt! based grasping pipeline and making it shareable with the community. We are also exploring the servoing issues of the mobile base to provide a universal motor control core which is independent of the hardware base. Moreover, we have a particular sensitivity to possible cyber-physical improvements on the platform allowing us to unlock technological barriers.

In this paper, developments related to the navigation, perception, communication and object manipulation capabilities of Epock, as well as hardware modifications are presented. For each section, an overview of the approach, the results and the future work is given.

2 Navigation and SLAM capabilities

A robust and reliable navigation has always been our priority. We have designed a complete and functional pipeline from map creation to autonomous navigation and localization. No collisions occurred during the overall long distance covered by the robot in different environments throughout the last year. We use AMCL for localization and ROS move_base for navigation. Furthermore, we have added remote encoders to the robot, allowing us to calculate the robot odometry which will not be impacted by wheel skidding. It enables us to improve our localization through a Kalman filter. We are working on an approach using AMCL and Gmapping/Cartographer simultaneously. The task of the SLAM algorithm is to perform a low frequency mapping to update the map during navigation. Moreover, since this year, we started to work on the local control of the robot through the implementation of a mobile base control. This allows us to locally deal with problems for which the ROS native navigation stack is not sufficient, such as going through a door or moving in a confined area.

3 Perception capabilities

Perception problems are mainly tackled by several state-of-the-art neural networks, detailed below. For each neural network, we have tried to find the best compromise between accuracy and performance. We mostly rely on siamese networks. This type of neural network can learn a similarity measure between objects [1]. Every object is mapped by the network to a vector. Object comparison can then be achieved using the distance between vectors. This enables these networks to naturally generalize to unknown instances. We also sometimes use some more problem specific approaches, described in dedicated subsections.

[2] https://www.youtube.com/watch?v=7Y4RjxWRqxE&t=5s

3.1 Object Recognition

Object recognition is mainly done using YOLO. We trained our own PyTorch implementation of this neural network from scratch on the COCO dataset. Using $320 \times 320px^2$ images as input, we were able to obtain a mAP_{50} of 50 without overfitting, close to the 51.5 value indicated in the original YOLO paper [2].

A new dataset is created for each new object we have to deal with and only the last YOLO layer is retrained. The dataset creation process is pretty heavy, so we designed a mechanical system to take photos of the objects, consisting of a rotating podium and a chroma-key background (green or blue) to partially automatize the process. The objects are then included in various fake backgrounds and augmented in several ways (luminosity, chromaticity, etc.). A COCO-like dataset is then generated and used for training. But the result of object recognition during the inference in the real environment could be improved. We therefore considered the solutions like Generative Adversarial Networks or Domain Randomization to overcome the gap between the synthetic and real data. These approaches may increase the quality of the generated dataset and increase performances in object recognition.

We recently added support for the MaskRCNN architecture, which enables the object mask extraction and not only a bounding box. Training is done using the same dataset described above. Recent work tries to extend this architecture in a siamese-like way. If it is successful, this approach could be used for one-shot object recognition and could enable to skip retraining the neural network each time an object is added.

Pragmatic approaches sometimes work better than neural networks. In the *SciRoc Pick and Pack* challenge, we used point cloud depth filtering to detect objects, based on the fact that they are on the shelves. We also took advantage of objects not being too close to one another to identify distinct blob of points for each object, with basic color filtering. This code was written with OpenCV.

3.2 Face detection and recognition

Face detection and recognition is done by a combination of two neural networks. The first one is a network that is used out of the box and that performs the face detection using a state-of-the-art Multi-Task Cascaded Convolutional Neural Network (MTCNN). Once the face bounding boxes are extracted as well as face features (eyes, nose, ...), the image is cropped and normalized. Faces are then processed through a siamese network. This network has been trained on more than 8000 different identities using the publicly available VGG 2 dataset. The distance between the faces and the previously detected people are computed and the closest person according to the L2 distance is returned if the value is below a predefined threshold. Examples of uses include checking if the right person is following Epock or during the *Find my mates* task in RoboCup@Home for identifying people faces.

3.3 Person attributes detection

We trained a neural network to identify a person's attributes from a picture. The network was trained on the CelebA public dataset using a resnet architecture. The training of this network relies heavily on the described above face detection process for normalization. Attributes include glasses, hat, hair color, face shape, etc. This is used to generate descriptions for the *Find my mates* task in RoboCup@Home. We wish to add attributes like hair size, skin color and some other useful features that are missing from the dataset using transfer learning from the current network.

3.4 Person tracking and *follow me* behavior

We worked on a solution to track people using a siamese network combined with a Kalman filter. First, we use YOLO to extract the person to be followed. We then get a vector representing the target vector using a siamese neural network trained on the Market1501 public dataset. This dataset contains a large amount of identities with images from six different cameras. During the tracking phase, several different people can be present. Each vector corresponding to a detected person is compared to the target vector in order to keep following the right person. We currently use a moving average of the previously detected vectors as a reference for distance calculation. This way, the target person's orientation or size can change during long tracking sessions without disturbing the system. We combined a Kalman filter[3] to use the available detections and the previous predictions to obtain the best guess of the person's current position. Assuming constant velocity motion, we can easily predict the target person's next position. With the measurements and the guess of the position given by the Kalman filter, we can filter the wrong people with more confidence. This approach could be used to track any type of object by generalizing the network to learn features independently of the object type.

During the *follow me* phase, we control the robot's head to always be looking in the direction of the target's last known position. Using this position as target position and depending on the situation, we use either:

- Dijkstra's algorithm to create the path to the target by taking into account the obstacles recorded on the cost map by the LIDAR and the RGB-D camera. The velocity commands are calculated by the local planner.
- the robot local servoing of the robot allowing it to evolve in a more restricted environment.

3.5 Pose Recognition

The pose recognition is done with the Posenet library[4] which computes face, body and hands 2D keypoint detection in real-time if equipped with a recent graphics card.

[3] https://en.wikipedia.org/wiki/Kalman_filter

[4] https://github.com/rwightman/posenet-python

Posenet gives the Body-Foot Estimation (default skeleton), but in the @Home competition, we need a more high-level information like "the person is seated" or "the person is pointing to the left". This is determined by simple rules based on the skeleton, such as: "If the knees are approximately at the same height than the hips and the neck is significantly higher, then the person is sitting". We wrote this code on top of Posenet. It can categorize the person's pose (sitting, standing, raising arm, etc.). A way to improve the pose recognition would be to use 3D keypoints.

3.6 Precise object localization

When Epock has to interact with an object at a known position in the environment, it cannot only rely on its own localization.

Sometimes the actual object position can be slightly different from the theoretical one and Epock's localization alone is not accurate enough for precision tasks such as grasping.

For example, in the *take out the garbage* challenge from RoboCup@Home, robots need to detect garbage bins to take trash bags out of them. In *SciRoc Pick and Pack* challenge, robots need to drop the items in a crate. These tasks require a precise localization of both the bins and the crate.

A LIDAR based pattern recognition is used to detect objects with a specific geometry. Indeed, a trash bin seen by the LIDAR is a semicircle. To add more robustness to this detection, the object theoretical position is also taken into account. An object matching the geometry, but located far from its expected position is probably not the seeked one.

4 Communication capabilities

For speech-to-text and natural language understanding (NLU) capabilities, Macarena has been developed by CATIE and uses an Nvidia conforming model as well as an open source nlu from snips. Macarena is an end-to-end speech-to-text model based on a conforming model [5] to which a GPT2 language model may or may not be added in order to select the most relevant transcriptions. Snips-nlu is a free, off-the-shelf solution for non-commercial software to extract structured intents from raw text written in natural language. In addition, a speech-to-text core has been developed. Macarena has been creating combining these two technologies.It is an NLP solution with offline operation and low latency, which is essential for the RoboCup competition. In contrast, the speech-to-text module requires to learn each sentence pattern in order to detect them afterwards. Otherwise it will not be able to understand the new ones on the fly. Each step of the Macarena process is linked to a specific ROS node. This enables both to control the execution stack and to bring modularity to the code. Each step can easily be replaced by a better performing algorithm.

[5] https://arxiv.org/abs/2005.08100

To fix some problems encountered during previous competitions and match its needs, Macarena has been designed to get some kind of continuous speech-to-text. Listening is programmatically triggered every few seconds and the robot tries to match each phrase heard with what it has learnd. The confidence score must be high enough to avoid triggering false positives. The continuous text-to-speech is quite experimental.

One observed issue in real conditions was some variability in speech recognition quality. In a robustness driven approach, we scripted automatic record of every audio interaction to double check every fail. This enabled us to detect some noises by hardware at different levels of the sound acquisition pipeline.

For text-to-speech, we are using TIAGo's default text-to-speech module: Acapela[6].

The different challenges encountered during our competitions were an opportunity to see how Epock interacts with people. We noticed that some people had difficulties understanding when to speak to Epock or what Epock was doing. To address this issue, we worked on interactions with a Human-centered approach by adding a screen where the current state of Epock is displayed (help needed, listening, moving, in action). Moreover, the integration of different leds make possible to know that the robot is performing actions (arm movement, waiting for speech, etc.). A user study was conducted to see if these pictograms were well understood and to get some feedback. Impressions and comments of people not involved in the project were collected. Now, with the additional pictograms, users can adapt their behavior according to the robot state shown on the screen as they would with another human. Our interface is based on Bastien and Scapin's Criteria.

This first study should help initiate an in-depth reflection on the creation of a global system thought to improve human/robot collaboration in a non-experimental environment.

5 Grasping capabilities

5.1 Arm control

Since the 2020 TDP, most of our effort has been focused on implementing a prehensile pipeline. The aim is to implement a succession of actions allowing the robot to : locate an item in space, create a representation of this space (octomap) and plan a trajectory in this high dimensional space using MoveIt!. Without forgetting our safety and robustness approach. This grasping pipeline is therefore an element of a more general grasping core which, depending on the situation, decides to use one module or the other of the grasping pipeline (octomap generation or not, etc.).

To reach an acceptable security level, we use high level guidelines and low level behaviors. The guidelines include:

- Position the robot in order to maximize the surrounding space.

[6] `http://www.acapela-group.com/`

- Only move the arm if the robot is still.
- After grasping an item, try to retract the arm as best as possible before moving the base of the robot.
- Always announce vocally that the arm will move.

As for the low level, a "zero gravity" behavior was implemented[7] based on the force/torque sensor attached to the wrist. Its principle is simple: if an abnormal force or torque is felt, then the arm is moved in the direction that would reduce that force/torque. Once the disturbance disappears, the arm resumes its movement[8]. When grasping an item, its weight must be accounted for. If the weight is unknown (e.g *Take Out The Garbage* test), this security is turned off when the item is grasped and turned on as soon as the item is weighed with the sensor. In its current state, the arm doesn't take into account the current measures of its motors and therefore won't adapt if an obstacle is touched with the elbow for example, this is a work in progress.

Epock's force sensing capabilities are also used to improve the overall robustness of the grasping pipeline. For example, if the weight measured by the wrist sensor goes below a predetermined threshold, the robot will assume that the item fell off its gripper and try to grasp it again if possible[9]. Also, some items are smaller than the precision we're capable of reaching at the end of the gripper along the Z axis (e.g. a fork on a table). In this case, we'll use a slow descending motion until resistance is felt when the table is touched. The same approach was used to grasp the garbage whose size is unknown prior to the test.

We've tested the impact of using checkpoints to achieve a reproducible behavior and tried to evaluate the appropriate force to apply on objects depending on their resistance and stiffness. Formalizing these methods and making a generic module out of them is a work in progress. Finally, exploratory work is currently being pursued with soft grippers and will be integrated in Epock if the results are good enough.

5.2 Perception challenges

Grasping requires specific information from the item. Such as its position, mask or orientation. We reviewed the state of the art on object recognition for grasping in 2020. Further to this state of the art, the implementation of object detection strategies based on several methods has been performed. On one hand, a MaskR-CNN algorithm has been developed and enables to provide the bounding box and the mask of the different objects. On the other hand, the specific characteristics of the objects (color, height, ...) are identified thanks to custom OpenCV code. Using the latter, we obtained good results in the final task *Pick and Pack* of the SciRoc 2019 competition (6/6 objects chosen, 5/6 objects packed).

Work on point cloud analysis has been recently undertaken. This allows, among other things, to detect flat surfaces (such as a table) where the robot

[7] https://www.youtube.com/watch?v=4auVU5Ifvpw&t=8s

[8] https://www.youtube.com/watch?v=7Y4RjxWRqxE&t=6s

[9] https://www.youtube.com/watch?v=btS7S6dadN4

could drop an object. Performing object detection using the point cloud rather than the 2D color image is then possible.

6 RoboCup experience and community outreach

In addition to the results presented in the introduction, we were exhibitors at 2018 and 2019 Cap Sciences' Village des Sciences, that gathered more than 3000 people over the weekend around robotics and the RoboCup competition[10]. We took part in the following events in 2019: NAIA (Bordeaux) and Vivatech (Paris). We participated in RoboCup@Home Education Challenge @EURCJ 2018 and co-organized a similar workshop in early 2019 that gathered 30 students in Bordeaux.

We carried out demonstrations of the RoboCup tests at NAIA.R (Bordeaux) in 2021 and during the SIDO exhibition (Paris) in 2021 and 2022. Moreover, since the RoboCup 2023 is in Bordeaux, we collaborate to promote national and regional events. This includes a RoboCup booth at the Maker Faire Lille, for example. One of our members is vice-president of an association organizing the Robot Maker Days in Bordeaux (EirLab Community).

7 Conclusion

In this paper, we have given an overview of the approaches used by the CATIE Robotics team for the RoboCup@Home competition. We have detailed our approaches for navigation, detection, communication and grasping. In all these areas, we have made significant improvements, but we are still building a robust basis for the competition by catching up to the state of the art, which consumes most of our time.

References

1. Gregory Koch, Richard Zemel, and Ruslan Salakhutdinov. Siamese neural networks for one-shot image recognition. In *ICML deep learning workshop*, volume 2, 2015.
2. Joseph Redmon and Ali Farhadi. Yolov3: An incremental improvement. *arXiv*, 2018.

[10] `http://www.cap-sciences.net/au-programme/evenement/village-des-sciences-2018`

Epock - Robot TIAGo Hardware Description

Robot TIAGo has been selected and is being customized for the @home competition purpose. Specifications are as follows:

- Base: differential drive base, 1m/s max speed.
- Torso: lifting torse (35cm lift stroke)
- One arm with a gripper (7 DoF). Maximum load: 2kg.
- Head: 2DoF (pan and tilt)
- Robot dimensions: height: 1.10m - 1.45m, base footprint: 54cm diameter
- Robot weight: 72kg.

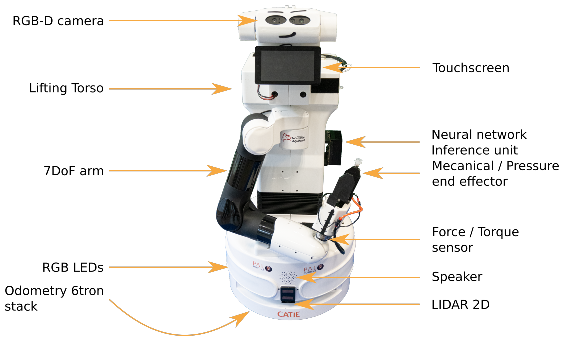

Fig. 1. Robot TIAGo

Our robot incorporates the following sensors:

- RGB-D camera
- 2D LIDAR
- Stereo microphone
- Speaker
- Sonars
- IMU
- Motors current feedback
- Wrist force and torque sensor

Robot's Software Description

For our robot we are using the following software:

- OS: Ubuntu 18.04
- Middleware: ROS Melodic
- Simulation: Gazebo
  http://gazebosim.org/
- Visualisation: RViz
  http://wiki.ros.org/rviz
- Localization: AMCL
  http://wiki.ros.org/amcl
- SLAM: Cartographer and GMapping
  https://github.com/googlecartographer/cartographer
  http://wiki.ros.org/gmapping
- Navigation: move_base
  http://wiki.ros.org/move_base
- Arms control: moveIt! and play_motion
  http://moveit.ros.org/
  http://wiki.ros.org/play_motion

Robot software and hardware specification sheet

- Face recognition: custom siamese neural network
- Object recognition: custom maskrcnn neural network
- Pose detection: PoseNet
  https://github.com/rwightman/posenet-python
- Speech recognition: custom solution (macarena, see dedicated section)
- Speech generation: Acapela
  http://www.acapela-group.com/
- Task executor: SMACH
  http://wiki.ros.org/smach

External Devices

Our robot relies on the following external hardware:

- Rode Videomic Pro external microphone
- External laptop
- 2 touch screens
- 2 6TRON stack developped by CATIE
  https://6tron.io/