Chief Scientist Office 2021 Team Description Paper

Takaaki Numai, Tatsuro Sakaguchi, Joshua Supratman and Airi Yokochi

December 10, 2021

Abstract. This paper details the RoboCup@Home Open Platform League (OPL) league team Chief Scientist Office from SoftBank Corp. Japan, for the participation at the RoboCup@Home 2022 OPL, Bangkok. Our goal is to develop a mobile manipulator that can perform tasks in everyday life. We will compare two types of robotic arms and two types of microphones to study which hardware is more suitable for service robots. Our software is composed of three layers: recognition layer, planning layer, and control layer. Each function is built around the ROS open source package. We will use this software to perform the tasks of door opening, person tracking, and speech recognition in RoboCup. We would like to develop a robot that can freely combine mechanical components and ROS packages to develop useful functions in human life.

1 Introduction

At SoftBank Corp. Chief Scientist Office, we are developing a reconfigure modular robots, similar to that of computer/automobile, and a flexible software framework. Our autonomous mobile robot "Cuboid-kun", equipped with a robot arm, will participate in the RoboCup@Home OPL. We previously participated in the World Robot Summit Future Convenience Store Challenge (WRS FCSC)[1] won 3rd place in 2020. This will be our first time participating in RoboCup@Home OPL.

1.1 Focus of Research

When developing a mobile manipulator, the hardware design changes significantly depending on how much performance is required from the mobile base and manipulator. We participated in this competition to verify the performance required to realize "robots that are useful to humans" and investigate how much

[1] https://wrs.nedo.go.jp/en/wrs2020/challenge/service/fcsc.html

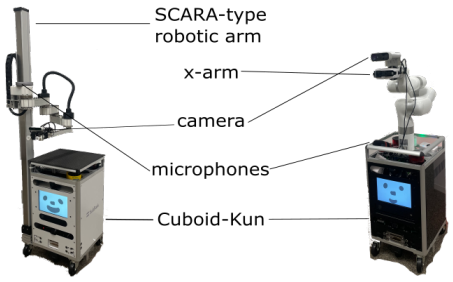

Fig. 1. Image of two robots: "Monar" (left) and "Cuboid-X" (right).

can be achieved with the existing technology. We implemented various functions using existing packages such as MoveIt but found many application problems such as long calculation time and unfinished functions. Thus our focus is to investigate and verify methods to solve the problems mentioned above and develop a mobile manipulator that can be used as a service robot.

2 Hardware

For RoboCup@Home OPL competition, we will prepare two robots, each with a different robot arm, for performance comparison. Figure 1 shows the image of the two robot hardware: "Cuboid-X", the robot equipped with xArm 7[2] robot arm, and "Monar", the robot equipped with SCARA-type robotic arm. Both robots share the same mobile base, "Cuboid-kun".

2.1 Mobile Base

"Cuboid-kun" is a self-navigating autonomous mobile robot that our team is developing. It is equipped with a computer, a power supply, and multiple sensors for obstacle avoidance. It is operated with differential drive and can carry a payload of 20 kg. The mobile base is intended to be used in houses and office buildings, and therefore has a small footprint of 370 mm inside a square. In addition, the hardware was designed to be reconfigurable, and therefore we can easily attach external modules[3] to the robot, such as a robotic arm.

2.2 Robotic Arm

"Monar" is equipped with a SCARA-type robotic arm designed by our team. The robot arm has 8 degrees of freedom and can carry a small payload of 500 g. The end-effector is equipped with a vacuum gripper with two suction cups and a compact RGB-D hand camera (RealSense D415). The robot arm was designed

[2] https://www.ufactory.cc/pages/xarm
[3] https://www.signagekun.com/cuboid

Fig. 2. Sound source localization.

for WRS FCSC, where one of the key tasks is to rearrange multiple objects on a shelf. To handle objects within a small space, we designed the robot arm to have a large operation space with thin and compact end-effectors.

"Cuboid-X" is equipped with xArm 7, a commercially available robot arm with 7 degrees of freedom. This robot arm is composed of large actuator modules and can carry a payload of 3.5 kg although the robot arm has small operation space compared to "Monar". The end-effector is equipped with a two-finger gripper and a high-performance RGB-D hand camera (Azure Kinect).

2.3 Microphones

We are currently experimenting with two different types of microphones, and plan to use the one with the better performance in the competition. One is a highly directional microphone, which is often used in RoboCup@Home. When the speaker is in front of the microphone, it is easier to recognize the speaker's voice. The disadvantage of using this type of microphone is that the speaker must stand in front of the microphone.

Another method we are trying is to use a microphone array module, such as XMOS, for source separation. We use a software called ODAS [1] to perform the source separation. This feature enables us to recognize speech in noisy environments by separating the speaker's sound source. Furthermore, it can distinguish between two or more people speaking at the same time. Figure 2 shows the sound source localization using ODAS_WEB.

3 Software

All our "Cuboid-kun"-based robots, such as "Monar" and "Cuboid-X", run on a common system. The overview of our software system, shown in Fig. 3, is comprised of three main layers:

1. the perception layer, collection of software modules that takes robot's sensor data and process them to create symbolic representations of the environment,
2. the planning layer, collection of software modules that takes the state of the environment to evaluate and generate a sequence of appropriate robot behaviors, and

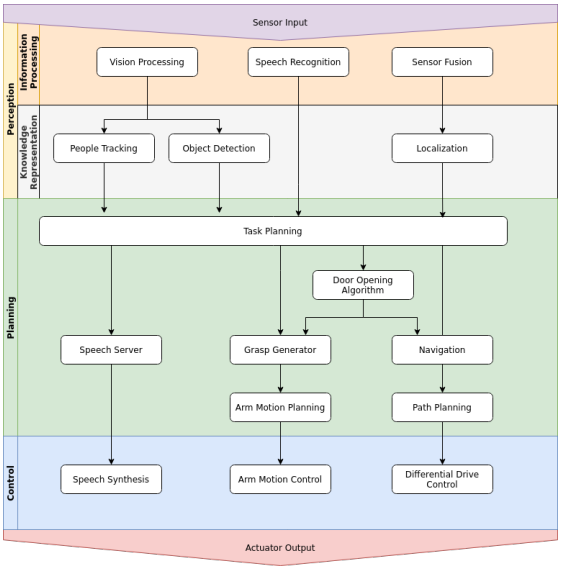

Fig. 3. "Cuboid-kun" software overview.

3. the control layer, collection of software modules that translates the planned behavior and sends appropriate commands to the actuators for the robot to interact with the environment.

The primary framework for software module interaction is Robot Operating System (ROS), with each software module representing one or several ROS nodes.

3.1 Object Classification and Pose Estimator Algorithm

We mainly use ClusterPointIndices, a function of jsk_pcl_ros[4] ROS package, to extract the point cloud data from the object on a plane and estimate the object's pose and collision shapes. However, for YCB objects used in RoboCup@Home league open source dataset, we used PERCH 2.0 [2] to estimate more accurate poses, as shown in Fig. 4.

We use YOLO, trained with MSCOCO dataset, for object classification. However, since the RoboCup@Home league requires more detailed recognition results than the MSCOCO dataset, we also use a classifier trained using GoogLeNet [3] for unknown objects.

3.2 Grasping Object Algorithm

We designed the object grasping algorithm in the following sequence. We first estimate the object poses and collision shapes using the object pose estimator

[4] https://github.com/jsk-ros-pkg/jsk_recognition

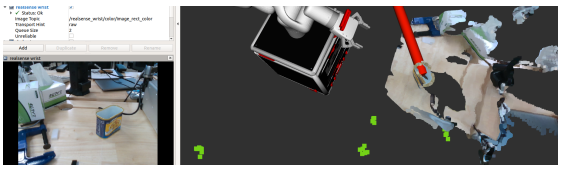

Fig. 4. Spam can pose estimation with PERCH 2.0.

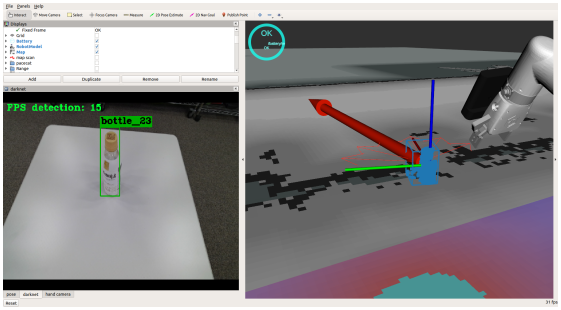

Fig. 5. Visualizing grasping object algorithm for grasping a pet bottle. The left panel shows YOLO detecting a pet bottle. The right panel shows the grasping object algorithm clustering the pet bottle's point cloud (blue) from plane top points, estimating the pose (rgb axis), and generating the grasp poses (red arrows). The robot then selects the best grasping pose (large red arrow) to pick the pet bottle.

algorithm as described in section 3.1. We also estimate the support table's (the area where the object resides on) collision shape and pose. After estimating the object's pose and collision shapes along with the support table, we generate the grasp candidates using HandleEstimator, another function of jsk ROS package. Finally, we provide the calculated information to MoveIt's[5] planning scene and use MoveIt's pick and place pipeline to compute the manipulator's trajectory path to grasp the object. Figure 5 visualizes the grasping object algorithm for grasping a pet bottle.

3.3 Door Opener Algorithm

We designed the door opener algorithm in the following sequence. First, we use YOLO to recognize doors and handle positions. The door dataset and the trained model of YOLO are described in [4]. Figure 6 shows the results of door detection by YOLO. YOLO can even detect a double-door as two doors along the center slit. Next, we estimate the pose of the door using jsk ROS package as described in section 3.1. In Fig. 7, the blue arrows represent the normal vector of the door, and in Fig. 8, the red bounding box indicates that the handle is

[5] https://github.com/ros-planning/moveit

Fig. 6. Door detection with YOLO.

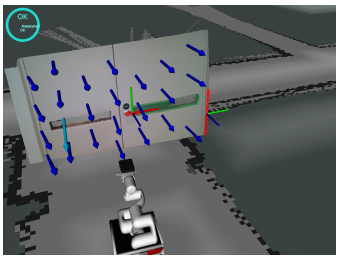

Fig. 7. Door plane estimation.

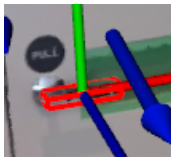

Fig. 8. Handle pose estimation.

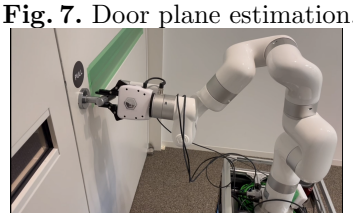

Fig. 9. Result of grasping the handle.

detected on the plane of the door. After estimating the handle pose, the robot grasps the handle using MoveIt as described in section 3.2. The robot achieved door opening by moving backward in the right direction based on the position of the handle and the door. Figure 9 shows how the robot grasped the handle based on these estimation results.

3.4 Person Tracker Algorithm

For detecting and tracking people we use the ROS package for spencer's robot [5]. The package can fuse several different detection results such as leg detection by 2D LiDAR and upper body detection by RGB-D. We also developed a tracking system that is even more robust to lost targets by integrating pose estimation results by OpenPose and image recognition results by YOLO as shown in Fig. 10.

For pose estimation, we considered HRNet [6] and CenterNet [7] as well as OpenPose. CenterNet is a method for anchor-less object detection similar to OpenPose. CenterNet can achieve the same processing speed with about $\frac{1}{3}$ of the GPU memory usage of OpenPose, but it assumes that all key points are always present in the bounding box. In robotics, this result is not desirable because the whole body is often not within the angle of view of the camera. On the other hand, HRNet provides higher performance by maintaining high resolution even during the process. However, the processing speed is about 1 Hz, inferior to OpenPose. HRNet is preferable to estimate poses in a static environment, but we needed to use these results in a dynamic environment for tracking, so we adopted OpenPose.

Fig. 10. People tracking system overview.

Furthermore, these tracking methods are prone to lose or swap tracking targets when the person being tracked hides in the shadows or overlaps with others. To avoid that, we have introduced an algorithm for re-identification using Deep SORT [8].

3.5 Speech Recognizer Algorithm

We use the Google Speech API and Dialogflow to recognize speech phrases and responses. We also use the snowboy[6] to detect hotword at the start of a conversation. Hotword triggers contribute greatly to recognizing conversational content with high accuracy, but without a sufficient number of training samples, snowboy cannot handle all of the people. Therefore, we use hotword detection only for conversations with operators.

4 Contribute

We have conducted several demonstration experiments with "Cuboid-kun" in multiple environments and scenarios and have provided "Cuboid-kun" as a research base to several external Japanese companies.[7][8][9].

[6] https://github.com/Kitt-AI/snowboy
[7] Meet the Robots of Smart City Takeshiba, Part 1: A Delivery Robot that Obeys Traffic Signals. https://www.softbank.jp/en/sbnews/entry/20210716_01
[8] SoftBank R&D: Evaluating the Latest Technologies with a View to Commercialization. https://www.softbank.jp/en/sbnews/entry/20210716_01
[9] Developed "in-building mobile environment measurement / notification system". https://www.toda.co.jp/news/2021/20211108_002989.html

5 Conclusions and future work

In this paper, we presented how we are developing our robots for the RoboCup@Home. Using voice recognition, navigation, person tracking, and object recognition, we have implemented several tasks for RoboCup@Home. Each of these functions has its challenges in terms of speed and success rate. We will be comparing algorithms, adjusting parameters, adding functions, and debugging for the RoboCup@Home. We will also compare two different configurations of microphones and arms, and adopt the better method. What to use as a benchmark is still an issue. It is also our goal to improve the quality of the robot so that it can be commercially available.

References

1. François Grondin and François Michaud. Lightweight and optimized sound source localization and tracking methods for open and closed microphone array configurations. *Robotics and Autonomous Systems*, 113:63–80, 2019.
2. Aditya Agarwal, Yupeng Han, and Maxim Likhachev. Perch 2.0 : Fast and accurate gpu-based perception via search for object pose estimation. In *IROS*, 2020.
3. Nizar Massouh, Lorenzo Brigato, and Luca Iocchi. Robocup@home-objects: Benchmarking object recognition for home robots. pages 397–407, 12 2019.
4. Miguel Arduengo, Carme Torras, and Luis Sentis. Robust and adaptive door operation with a mobile robot. *Intelligent Service Robotics*, May 2021.
5. Timm Linder, Stefan Breuers, Bastian Leibe, and Kai O. Arras. On multi-modal people tracking from mobile platforms in very crowded and dynamic environments. In *2016 IEEE International Conference on Robotics and Automation (ICRA)*, pages 5512–5519, 2016.
6. Ke Sun, Bin Xiao, Dong Liu, and Jingdong Wang. Deep high-resolution representation learning for human pose estimation. In *CVPR*, 2019.
7. Xingyi Zhou, Dequan Wang, and Philipp Krähenbühl. Objects as points. In *arXiv preprint arXiv:1904.07850*, 2019.
8. Nicolai Wojke and Alex Bewley. Deep cosine metric learning for person re-identification. In *2018 IEEE Winter Conference on Applications of Computer Vision (WACV)*, pages 748–756. IEEE, 2018.
9. Z. Cao, G. Hidalgo Martinez, T. Simon, S. Wei, and Y. A. Sheikh. Openpose: Realtime multi-person 2d pose estimation using part affinity fields. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2019.
10. Yasemin Bekiroglu, Naresh Marturi, Máximo A. Roa, Komlan Jean Maxime Adjigble, Tommaso Pardi, Cindy Grimm, Ravi Balasubramanian, Kaiyu Hang, and Rustam Stolkin. Benchmarking protocol for grasp planning algorithms. *IEEE Robotics and Automation Letters*, 5(2):315–322, 2020.
11. Konstantinos Chatzilygeroudis, Bernardo Fichera, Ilaria Lauzana, Fanjun Bu, Kunpeng Yao, Farshad Khadivar, and Aude Billard. Benchmark for bimanual robotic manipulation of semi-deformable objects. *IEEE Robotics and Automation Letters*, 5(2):2443–2450, 2020.

"Cuboid-kun" Hardware Description

Shared Base Hardware

- Base: differential drive, 0.8 m/s max speed.
- Dimensions: 370 x 370 x 670 [mm].
- Weight (base only): 30 kg
- Sensors:
  - LIDAR x 5
    - PaceCat x3
    - AkuSense x2
  - HC-SR04 Sonar x12
  - Kinect V2 x1
  - MPU-6050 IMU x1
- Speaker: Arcs X-118 Car speaker
- Microphone: RODE stereo video mic
- Microphone array: XK-USB-MIC-UF216
- CPU: Intel(R) Core(TM) i9-9900 CPU @ 3.10 GHz
- RAM: 32 GB
- GPU: GeForce GTX 1650 (4 GB)

Manipulator

	Cuboid-X	Monar
type	x Arm 7 (7 DOF)	SCARA-type custom (8 DOF)
end-effector	two finger gripper	vacuum gripper
payload	3.5 kg	500 g
hand camera	Azure Kinect	Realsense D415

Robot's Software Description

- Platform: Ubuntu 18.04 ROS melodic
- Navigation: move_base, eband_local_planner
- Arm Control: MoveIt
- Face recognition: Dlib.
- Speech recognition: ODAS, Dialogflow
- Speech generation: Google Speech API, Dialogflow
- Human Tracking:
  - spencer_people_tracking
  - OpenPose
  - Deep SORT
- Object recognition:
  - YOLO v3
  - jsk_pcl_ros
  - GoogLeNet
  - perch 2.0