UT_MIL@Home DSPL
2018 Team Description Paper

Yujin Tang, James Borg, Yusuke Kurose, Francesco Savarese, Takayoshi Takayanagi, Mohammad Reza Motallebi, Antonio Tejero-de-Pablos, Yingyi Wen, Toshihiko Matsuura, Jen-Yen Chang, Li Yang, Yoshitaka Ushiku and Tatsuya Harada

Machine Intelligence Laboratory,
Department of Mechano-Informatics,
Graduate School of Information Science and Technology,
The University of Tokyo,
7-3-1 Hongo, Bunkyo-ku, Tokyo 113-8656, Japan,
https://mil-tokyo.github.io/robocup/

Abstract. UT_MIL@Home is a team based in the Machine Intelligence Laboratory at The University of Tokyo. We focus on frontier research in various fields of artificial intelligence (AI) and our desire to translate research into applications has encouraged us to participate in the RoboCup@Home competition. In this paper, we briefly introduce the design of our team's current system, the technologies we have used as well as ongoing research elements we plan to incorporate in the near future.

1 Introduction

UT_MIL@Home is based in the Machine Intelligence Laboratory at the University of Tokyo. This lab is specialized in machine learning and our members are skilled in areas such as object recognition and natural language processing. By participating in the RoboCup@Home Domestic Standard Platform League (DSPL), we wish to put into practice our knowledge in machine learning, developing a home-service robot which serves as both proof that artificial intelligence is ready to be applied into daily lives and as a test of where its limitation lies. In spite of being the first time we participate in this competition, we aim to achieve state-of-the-art performance in high-level tasks such as navigation and human-robot interaction.

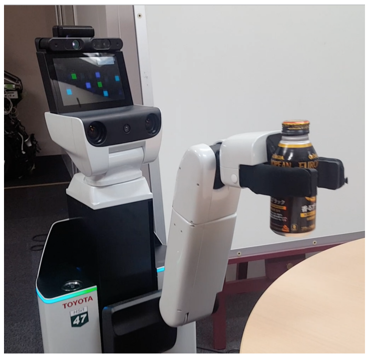
This paper overviews the basic hardware and software, as well as the basic modules, used in our robot for the competition. Besides the basic features of our Human Support Robot (HSR [1]), in order to execute the tasks of the competition, we have developed five modules: person identification, manipulation, navigation, object recognition and speech. To make scientific contributions we are exploring and developing algorithms that combine multi-modal object recognition (E.g., detect object pose with RGBD images) and control (E.g., path planning and deep reinforcement learning), and shall publish our methods and results to relevant conferences/journals.

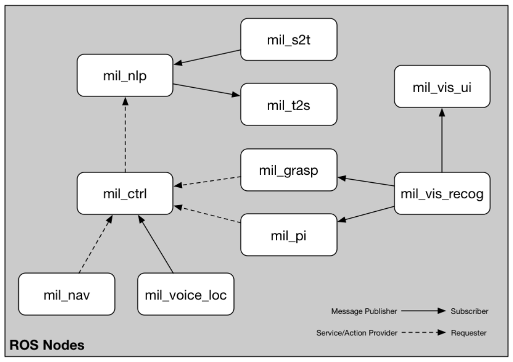
The rest of the paper is organized as follows. In Sec. 2 we show an overview of the current system, after which we give some details of the core components for some of the RoboCup@Home competition tasks in Sec. 3. Finally we conclude in Sec. 4.

2 System Overview

RoboCup@Home DSPL requires the use of an unmodified HSR from Toyota. Fig. 1 (left) demonstrates our robot executing a "fetch me the black bottle" task[1], and Fig. 1 (right) shows the current design of our system.

Fig. 1: Demonstration of a task and current system design.





The following is a list of ROS packages/nodes in our current system. We shall briefly sketch the responsibility of each here and give detailed descriptions in the following sections when we talk about the core components for the RoboCup@Home tasks.

mil_s2t Speech to text module. HSR's internal Python interface is adopted.
mil_t2s Text to speech module. Google Cloud Speech API, Deep Speech [2], WaveNet [3], etc are being considered and deployed.
mil_nlp Natural language processing module. It consists of various processing methods. For instance, at the time of writing, we simply use regular expression matching for user intention understanding and we use seq2seq [4] for QA tasks.
mil_ctrl Task control module. Though shown as a single node in Fig. 1 (right), this is a group of nodes each of which is responsible for a high level task such as "fetch me the black bottle" and "introduce me to the others".

[1] Check out our qualification video at https://mil-tokyo.github.io/robocup/

mil_nav Navigation module. It is responsible for map construction and locomotion of the robot.

mil_voice_loc Voice localization module. In some tasks, HSR responds to sounds (E.g., rotate to face the person giving the command). We use hark [5] to carry out the task.

mil_grasp Manipulation module. We exploit the Python interface provided by Toyota to move around the end-effector and grasp/release objects once we have detected the objects' poses.

mil_pi Person identification module. Our HSR robot can perform person identification based on its vision. The supporting technology is a combination of [6] and [7].

mil_vis_recog Visual recognition module. Rather than making a node for each task involving visual recognition (E.g., YOLO for object detection, Faster R-CNN for face detection, etc), we make this module responsible for all tasks the output of which are bounding boxes and class predictions.

mil_vis_ui Visual recognition UI module. This is simply a visualization node for debugging and behavior explanation.

3 Core Components

3.1 Person Identification

In this section we outline the preliminary person identification system we have built for our initial video submission. Following this, possible improvements and areas of research regarding person identification we plan to undertake before the competition are detailed.

Our preliminary person identification system consists of a three-stage pipeline.

1. Face Detection
2. Align and Encode
3. Classification

Face detection is performed with a widely used deep learning architecture for object detection, faster R-CNN [6]. Going forward we plan to consolidate this network and the network being used in the object detection module. Face landmark estimation [7] is performed on detected faces to warp the face image such that eyes and lips are always in the same position. The features of this realigned face are embedded with another deep network and classification is performed with K nearest neighbours. This pipeline gives reasonable results, however, we believe there is several improvements that can still be made.

One issue with the current implementation is that identification relies exclusively on facial features. In situations where the subjects face is occluded, out of view, or the subject is facing away from HSRs camera, identification cannot be performed. To remedy these issues our team plans to extend our current identification module with a voice recognition system. Furthermore, we would like to augment the classification network to also predict age and gender. Understanding the age and gender of a subject can be useful in managing appropriate responses and interactions with subjects.

3.2 Manipulation

Our current implementation for manipulation is an independent module that exposes an action accepting a 6D pose vector ([x, y, z, roll, pitch, yaw]) to the rest of the world. The object recognition module (described in Sec 3.4) is responsible for estimating the pose of the target object. To be concrete, a high level control component (E.g., mil_ctrl) requests the object recognition module for all the bounding boxes and class labels from the current observation (E.g., RGB image). If the desired object is in the detections (in the case where there are many detections, our robot takes the leftmost one in the current implementation), the control component computes the position of the central point (the first 3 entries in the pose vector) by extracting the corresponding point cloud data from the same camera. It then fills in the last 3 entries of the pose vector with $[0, 0, -\pi/2]$ (because we assume the object is placed in a pose that is always "pointing upward" at the time of writing), and then command the manipulation module to move the end-effector to be in the desired pose.

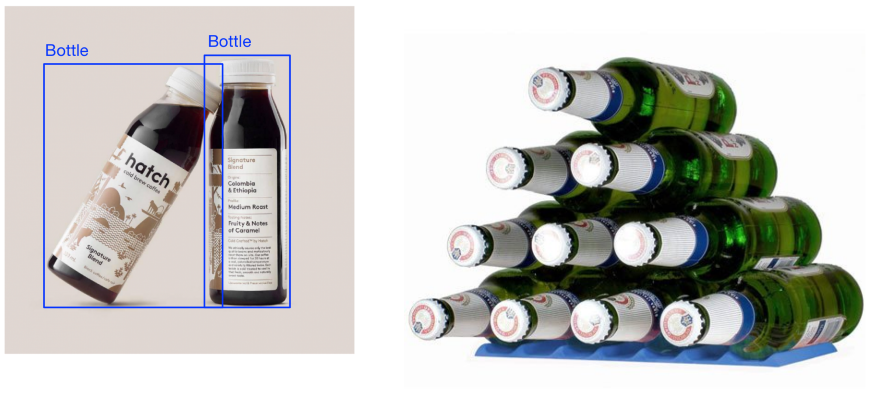
Besides the unrealistic orientation assumption, we are aware of several other deficiencies in our current implementation and are continuously improving our methods. Detailed problem statements and tentative solutions will be published in relevant conferences/journals, but we would like to give a rough description of the problems here. To get rid of the impractical orientation assumption, we could train a neural network from RGB images to determine the optimal grasping point by utilizing open datasets such as [8], and feed the image segment containing the target object to this network to get grasp locations. In this way, we can handle the case shown in the left of Fig. 2. However, this method does not solve the case illustrated in the right of Fig. 2 where the bottles are piled and oriented in a way that is pointing perpendicular to the plane of observation because most of the open datasets for training are images of objects horizontally placed on a flat plane. For this kind of configurations, we think 3D models or multi-angle views (E.g. by moving the robot body up and down or moving the end-effector around the target object and taking pictures using the camera in the gripper) of the scene are necessary. Or we can also integrate vision based deep reinforcement learning into the task although the cost of sample collection is high. Pose estimation is an active research topic in the community and we would like to contribute through our practice in the RoboCup competition.

3.3 Navigation

In this section we describe the current path planning system based on the ROS *move_base* and *navigation_stack* [9]. *mil_nav* is the module responsible for managing robot navigation. Referring to Figure 3 we briefly describe each module and their interaction in the context of path planning. We present 3 main modules:

1. **mil_ctrl** is responsible for receiving instructions from outside and dispatching commands to involved modules
2. **mil_env_und** is responsible for providing *mil_nav* a good description of the environment

Fig. 2: Illustration of grasping method and limitations.



3. **mil_nav** is responsible for processing requests from (1) and utilizing data from (2). This module sends commands to the ROS *nav_stack* and manages the response.

The navigation module is implemented as a ROS server. It acts as a bridge between the control module and the robot itself. *mil_nav* exposes several basic actions for moving the robot. Commands received from *mil_ctrl* are still high level commands and this module is responsible for translating them to suitable messages for the *navigation_stack*. Several clients are launched to serve different kind of requests. *mil_nav* frequently interacts with the *mil_env_und* module, receiving a detailed description of the environment. In this way, messages filtering and preprocessing, as well as semantic navigation are possible. A decision maker is also embedded within *mil_nav* capable of requesting actions and interacting directly with the robot without overloading *mil_ctrl*. Complex choices, however, are always made by the controller, having a richer knowledge of the entire system. It receives continuous feedback from *mil_nav* and uses this data for modifying the current behavior of the robot in real time. When the movement is completed a final message is sent to the controller that can schedule next actions according to the result.

3.4 Object recognition

This section details the object detection and segmentation algorithms proposed for supporting object manipulation and path planning tasks of the competition. To improve upon efficiency and accuracy of other detection methods our proposal is to utilize:

1. Object detection trained with RGB-D data, leveraging transfer learning and domain adaptation.
2. Propagation of key frame feature maps with optical flow [10] [11]

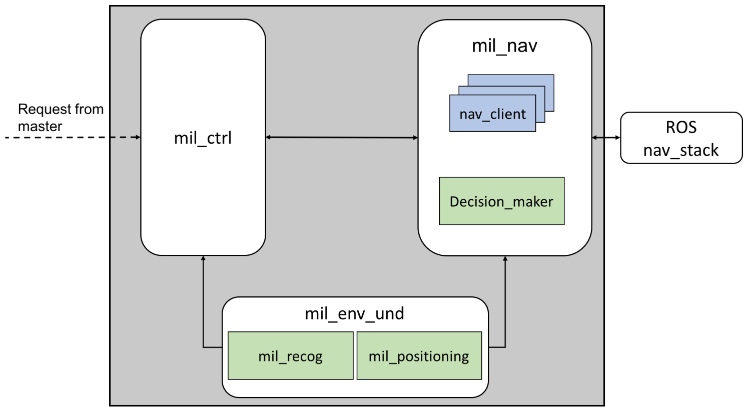


Fig. 3: Path Planner Structure

Many modern object detection techniques handle RGB data exclusively. Robocup competition provides our team an opportunity to investigate object detection techniques utilizing RGB-D data. Work by [12] demonstrates that general improvements can be achieved with added depth information. Building upon this work we intend to apply transfer learning and domain adaptation, taking advantage of the wealth of RGB data for object detection whilst exploiting the added depth information available to the HSR.

Performing object detection in real-time can be difficult especially with limited resources and a large set of possible objects. To ensure efficiency of our algorithm we propose a network employing feature propagation. Deep feature maps, acquired with computation heavy networks are retrieved at key frames only. Features maps at intermediate frames between key frames are acquired by propagation. This propagation can be achieved using optical flow (as was done in Deep Feature Flow [10]), however our team hopes to compare a number of propagation methods.

3.5 Speech

The Speech sub-system handles the audio Input/Output of the system. It consists of three modules (nodes): Speech2Text, Text2Speech and natural language processing (NLP). The Speech2Text module does the job of transcribing the words spoken to HSR and publishing the text on a dedicated Topic. As of the time of writing this paper, this module mainly uses the Google Cloud Speech API. The current implementation sends clips of voice recordings and receives the text after 1-2 seconds. As a next step, we plan to implement other Speech Recognition solutions, such as Deep Speech [2], and wav2letter [13,14] to provide an offline alternative. To this end, we intend to train our model with public data such as Mozilla Common Voice (https://voice.mozilla.org/) and also integrate training data that we think might be used more for the scenarios in this competition.

The Text2Speech module on the other hand does the job of outputting the audio of the text provided to it via a dedicated topic. Currently we rely on the HSR speech synthesis module provided for this.

The NLP module interprets commands and instructions provided to HSR in text format (either directly via CLI or through the Speech2Text module) and converts them to the appropriate action exposed in the Controller module. As of the time of writing, the instructions and the format they should be in are limited, however, we intend to expand this to make the interaction more dynamic and natural. In the current setting, NLP module waits for and detects a wake-up word at first (similar to "Siri"). Upon successfully detecting the wake-up word, it responds to the user that it has aknowledged the wake-up word waits for the instruction to follow.

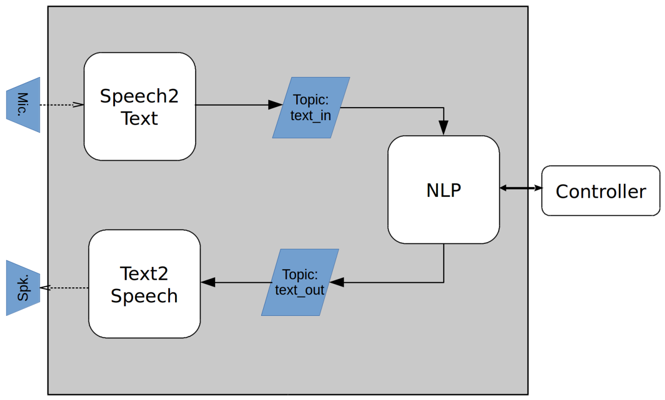


Fig. 4: Speech Sub-System

4 Conclusions

In this paper, we summarized the technical details of our Human Support Robot (HSR), with which we plan to participate in the RoboCup 2018 Montreal. Besides the original hardware and software, in order to tackle the RoboCup@Home DSPL tasks, we developed five modules: person identification, manipulation, navigation, object recognition, and speech. These modules allow our HSR to successfully perform basic tasks. In the future, we will apply our knowledge in machine learning to achieve advanced behavior (i.e., navigating through different classes of obstacles and manipulating a great variety of objects) and make our results and methods public to the community.

Acknowledgements

This work was partially supported by NVIDIA NVAIL program, and JST CREST Grant Number JPMJCR1403, Japan.

References

1. Toyota Motor Corporation. Human support robot. http://www.toyota-global.com/innovation/partner_robot/family_2.html. Online; accessed 15 January 2018.
2. Awni Y. Hannun, Carl Case, Jared Casper, Bryan Catanzaro, Greg Diamos, Erich Elsen, Ryan Prenger, Sanjeev Satheesh, Shubho Sengupta, Adam Coates, and Andrew Y. Ng. Deep speech: Scaling up end-to-end speech recognition. *CoRR*, abs/1412.5567, 2014.
3. Aäron van den Oord, Sander Dieleman, Heiga Zen, Karen Simonyan, Oriol Vinyals, Alex Graves, Nal Kalchbrenner, Andrew W. Senior, and Koray Kavukcuoglu. Wavenet: A generative model for raw audio. *CoRR*, abs/1609.03499, 2016.
4. Ilya Sutskever, Oriol Vinyals, and Quoc V. Le. Sequence to sequence learning with neural networks. In *Proceedings of the 27th International Conference on Neural Information Processing Systems - Volume 2*, NIPS'14, pages 3104–3112, Cambridge, MA, USA, 2014. MIT Press.
5. Honda Research Institute Japan Audition for Robots with Kyoto University. Hark. http://www.hark.jp/. Online; accessed 15 January 2018.
6. Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster r-cnn: Towards real-time object detection with region proposal networks. In C. Cortes, N. D. Lawrence, D. D. Lee, M. Sugiyama, and R. Garnett, editors, *Advances in Neural Information Processing Systems 28*, pages 91–99. Curran Associates, Inc., 2015.
7. Vahid Kazemi and Josephine Sullivan. One millisecond face alignment with an ensemble of regression trees. In *Proceedings of the 2014 IEEE Conference on Computer Vision and Pattern Recognition*, CVPR '14, pages 1867–1874, Washington, DC, USA, 2014. IEEE Computer Society.
8. Ian Lenz, Honglak Lee, and Ashutosh Saxena. Deep learning for detecting robotic grasps. *Int. J. Rob. Res.*, 34(4-5):705–724, April 2015.
9. ROS.org. ROS navigation. http://wiki.ros.org/navigation. Accessed: 2018-01-11.
10. Xizhou Zhu, Yuwen Xiong, Jifeng Dai, Lu Yuan, and Yichen Wei. Deep feature flow for video recognition. *CoRR*, abs/1611.07715, 2016.
11. Philipp Fischer, Alexey Dosovitskiy, Eddy Ilg, Philip Häusser, Caner Hazirbas, Vladimir Golkov, Patrick van der Smagt, Daniel Cremers, and Thomas Brox. Flownet: Learning optical flow with convolutional networks. *CoRR*, abs/1504.06852, 2015.
12. Judy Hoffman, Saurabh Kumar Gupta, Jian Leong, Sergio Guadarrama, and Trevor Darrell. Cross-modal adaptation for rgb-d detection. *2016 IEEE International Conference on Robotics and Automation (ICRA)*, pages 5032–5039, 2016.
13. Ronan Collobert, Christian Puhrsch, and Gabriel Synnaeve. Wav2letter: an end-to-end convnet-based speech recognition system. *CoRR*, abs/1609.03193, 2016.
14. Vitaliy Liptchinsky, Gabriel Synnaeve, and Ronan Collobert. Letter-based speech recognition with gated convnets. *CoRR*, abs/1712.09444, 2017.

A Team Information

A.1 Team Name

UT_MIL@Home

A.2 Contact Information

robocup2018@mi.t.u-tokyo.ac.jp

A.3 Website URL

https://mil-tokyo.github.io/robocup/

A.4 Team Members

1. Yujin Tang
2. James Borg
3. Yusuke Kurose
4. Francesco Savarese
5. Takayoshi Takayanagi
6. Mohammad Reza Motallebi
7. Antonio Tejero-de-Pablos
8. Yingyi Wen
9. Toshihiko Matsuura
10. Jen-Yen Chang
11. Li Yang

B Robot Information

B.1 Photos of HSR

Fig. 5 shows our team's HSR.[2].

B.2 Hardware Specification

Table. 1 outlines the detailed hardware information of our HSR.

B.3 External Devices

Table. 2 shows the external devices attached to our robot.

B.4 3rd Party Softwares

Table. 3 shows 3rd party softwares used in this project.

[2] For more images of the robot please refer to http://www.toyota-global.com/innovation/partner_robot/family_2.html.

Fig. 5: Our team's HSR.

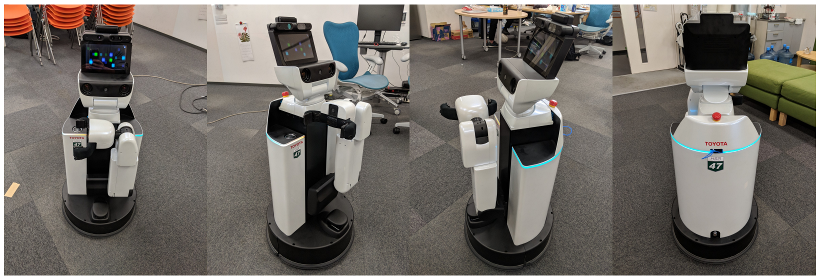


Table 1: Detailed information about the hardware of our HSR.

Name	Human support robot (HSR)
Footprint	430mm
Height(min/max)	1005/1350mm (top of the head height)
Weight	About 37kg
CPU	4th Gen Intel Core i7 (16GB RAM, 256GB SSD)
Embedded GPU board	NVIDIA Jetson TK1
Battery	Lithium-ion battery 25V/9.5Ah
Sensors on the moving base	IMU Laser range sensor Magnetic sensor for stop (N-pole detection type) x2
Arm length	600mm
Arm payload (recommended/max)	0.5/1.2kg
Sensors on the head	RGB-D sensor (Xtion PRO LIVE) x 1 Absolute type joint angle encoder Stereo camera x1 Wide-angle camera x1 Microphone array x1
Sensors on the arm	Absolute type joint angle encoder 6-axis force sensor
Sensors on the gripper	Potentiometer Gripping force sensor Wide-angle camera
Expandability	USB x3 VGA x1 LAN x1 Serial x1 15V-0.5A output x1 TK1 USB2.0 x1 TK1 Serial x1

Table 2: HSR external devices.

#	**Device**
1	External laptop for image and language processing
2	Cloud services for speech to text (E.g., Google Cloud Speech API)
3	External omni-directional microphone array (E.g. TAMAGO-03)

Table 3: Detailed software information used in HSR.

System	OS	Ubuntu 16.04
	Middle ware	ROS Indigo/Kinetic
Image processing	Face recognition	caffe, dlib
	Object recognition	TensorFlow, Keras
Navigation	Path planning	navigation_stack (ROS)
Speech processing	Speech2Text	Google Cloud Speech API
		DeepSpeech
		WaveNet
	Text2Speech	HSR speech synthesis module