KameRider OPL @Home 2019 Team Description Paper

Jeffrey Too Chuan Tan[1]

[1] Nankai University, China
i@jeffreytan.org
http://openbotics.org/kamerider/

Abstract. This document is the team description paper of the KameRider OPL team for the participation of @Home Open Platform League in RoboCup 2019 Sydney, Australia. KameRider team is a collaborative effort that aims to develop an open robot platform for service robotics. This paper describes the motivation of this effort, the hardware and software of the robot developments, and the scientific contribution and social impacts of our work via the educational initiative – RoboCup @Home Education.

1 Introduction

KameRider team is a collaborative effort that aims to develop an open source robot platform for service robotics. Started from 2013, the limited development resources and manpower team condition had urged a strong motivation to develop a more affordable yet functional solution to take part in RoboCup @Home league and service robot development.

The current team objectives are as follows:

A. Utilize open source solutions for both hardware and software to develop an open source robot platform that is affordable (low cost) and with large community support.
B. Development and participation in RoboCup @Home to benchmark the open source robot platform performance.
C. Support the educational initiative – RoboCup@Home Education[1].

2 Background and Motivation

2.1 The challenges of RoboCup @Home

Starting from 2006, RoboCup @Home [1] has been the largest international annual competition for autonomous service robots as part of the RoboCup initiative. The challenge consists of a set of benchmark tests to evaluate the robots' abilities and performance in a realistic non-standardized home environment setting [2]. It has greatly fostered artificial intelligence development in various domains including human-robot

[1] http://www.robocupathomeedu.org/

interaction, navigation and mapping in dynamic environments, computer vision, object recognition and manipulation, and many more developments on robot intelligence.

However, it is observed that the development curve of the RoboCup @Home teams have a very steep start. The amount of technical knowledge and resources (both manpower and cost) required to start a new team has made the event exclusive to only established research organizations. For this reason, our team had initiated the development of an open source robot platform for RoboCup @Home in 2013. The goal of the project is to develop a basic robot platform to facilitate startup team for the participation in RoboCup @Home. It is developed based on open source solutions for both hardware and software developments for low cost and large community support to facilitate startup of the novice teams.

3 Robot Developments

3.1 Open source robot platform development

The open robot platform has a current basic robot hardware configuration (Fig. 1) for fundamental robot platform and add-on modular component systems for customized applications. For example in Fig. 2, a manipulator system (with top vision) and an extended top vision system are added to the hardware configurations during RoboCup Japan Open 2015 and RoboCup 2015 Hefei for the applications in *Restaurant* task and *Follow Me* task.

TutleBot as the basic robot hardware platform. TurtleBot[2] is a low cost (basic kit is approximately USD 1,000), personal robot kit with close integration to popular open source software, ROS[3] (Robot Operating System) [3]. The open source robot kit is adapted as the basic mobile platform for this development. The vertical range of the mobile manipulation can be adjusted with an elevated arm with linear motor, a secondary vision system is paired with the robotic arm for object recognition in the manipulation tasks, and 3D printed parts for component systems. An interactive interface with speech and facial expressions is in development for human-robot interaction. A general laptop PC (currently working on a single board computer system) with speakers and microphone is served as the main robot controller.

ROS as the robot software framework. ROS (Robot Operating System) is an open source robot software framework with a large community to provide huge collection of robotic tools and libraries. With ROS as the fundamental software framework, this work will adapt and assemble ROS packages and stacks to realize the navigation, manipulation, vision and speech functions of the robot in order to perform the tasks in RoboCup @Home.

Cloud-connected. The robot system is controlled by an onboard computer system as the main robot controller to ensure stable low-level controls. Furthermore, the

[2] http://www.turtlebot.com/
[3] http://www.ros.org/

computer system can be connected with cloud systems for extra computing (e.g. image processing), knowledge database (e.g. dialogue engine) and online resources (e.g. wearable data).

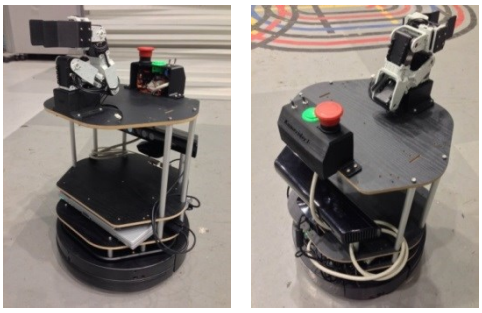

Fig. 1. The current basic robot hardware configuration

Fig. 2. Two hardware configurations during RoboCup Japan Open 2015 (left) and RoboCup 2015 Hefei (right)

3.2 Robot navigation

With the Kobuki[4] and MS Kinect sensor as the mobile base hardware configuration, the TurtleBot navigation package[5] is used for robot navigation with map building using *gmapping* and localization with *amcl*, while running the navigation stack in ROS. With the prebuild map and predefined waypoint locations, we can then instruct the robot to travel to a specific goal location with path planning using *actionlib*[6].

Navigation in known and unknown environments (Help-me-carry). With the top second vision system configuration (Fig. 2), we have developed the navigation system in known and unknown environments for *Help-me-carry* and *Restaurant* tasks. Based on the TurtleBot navigation package, we have combined it with the people tracking package, for online update of the map while following the operator in the unknown environment.

[4] http://kobuki.yujinrobot.com/home-en/
[5] http://wiki.ros.org/turtlebot_navigation/
[6] http://wiki.ros.org/navigation/Tutorials/SendingSimpleGoals

3.3 Speech interaction and sound source localization

For human speech interaction, we use CMU Pocket Sphinx[7] as our robot speech recognizer. It is a lightweight speech recognizer with a support library called Sphinxbase. We build our application with the latest version "sphinxbase-5prealpha". We use *gstreamer* to automatically split the incoming audio into utterances to be recognized, and offers services to start and stop recognition. The recognizer requires a language model and dictionary file, which can be automatically built from a corpus of sentences. For text-to-speech (TTS), we are using the CMU Festival system together with the ROS *sound_play* package.

In order to improve the speech recognition efficiency, we use a strategy for the robot to listen for activation keyword first, and once the keyword is recognized, it switches to *ngram* search to recognize the actual command. Once the command has been recognized, the robot can switch to grammar search to recognize the confirmation, and then switch back to keyword listening mode to wait for another command.

In order to improve the speech recognition accuracy in a noisy environment, we use SphinxTrain tool in the CMUSphinx to train the recordings of sentences taken in the noisy environment. The SphinxTrain tool extracts the sounds of the noisy environment by using a large number of the above recordings as a database. We use the obtained parameters to replace the original parameters for better speech recognition detection.

Sound source localization. Apart from human speech interaction, we have also tested sound source localization using HARK[8] for possible people search when the person is speaking outside of the robot visual perception area.

3.4 Robot vision and person/gender/object recognition system

A second vision system (Fig. 2) is built on top of robot with MS Kinect for people/gender/object detection and recognition. The people tracking package is used to track people in the *Follow Me* task.

Person recognition. We built a Convolutional Neural Network (CNN) with Tensorflow as well as Keras. When we need to add a new person into our database, we use a digital camera to take about 1000 pictures of this person, and then we use Haar-like face detection method to detect human face in each picture and add the face region of each picture to our database with its labels. After that we use a laptop with NVIDIA GTX1070 to run the CNN, so that we can get our own model to recognize the person. The algorithm is explained in the following figure.

Gender recognition using an online API from Baidu-AI. During the competition, we capture and upload the photo to the Baidu-AI cloud server, and we can get the gender recognition results labeled on the photo.

Object recognition system. We use YOLO (You Only Look Once) for object detection. In the *Storing Groceries* task, we use the Kinect sensor for shelf detection,

[7] http://wiki.ros.org/pocketsphinx
[8] http://www.hark.jp/wiki.cgi?page=HARK-ROS-TURTLEBOT

table detection and object detection. Before the competition, we take photos of the predefined object, and then we do labeling by adding annotation labels and bounding boxes for each image. We capture the images in different angle, different light condition as well as different background to ensure suitable generalization of our model to deal with the competition conditions.

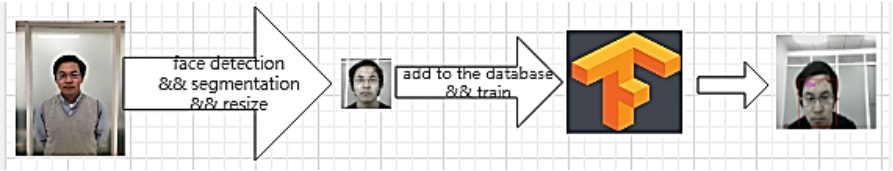

Fig. 3. Person recognition system

Human gesture detection. In our human gesture detection system (Fig. 4), we use CMU OpenPose as our skeleton detector (Fig. 5). It is a real-time multi-person keypoint detection library for body, face, hands, and foot estimation. The OpenPose demo requires a RGB image and then returns the number of people as well as their skeleton positions. To get the human pointing direction, 3D coordinates of the wrist joint and elbow joint are necessary. We combine the OpenPose result and point cloud library (PCL) to get the positions described in the head RGB-D sensor coordinate system, then the TF matrix is used to transform them to the map coordinate system. Additionally, space vector method is used to calculate which point on the ground the human is pointing to.

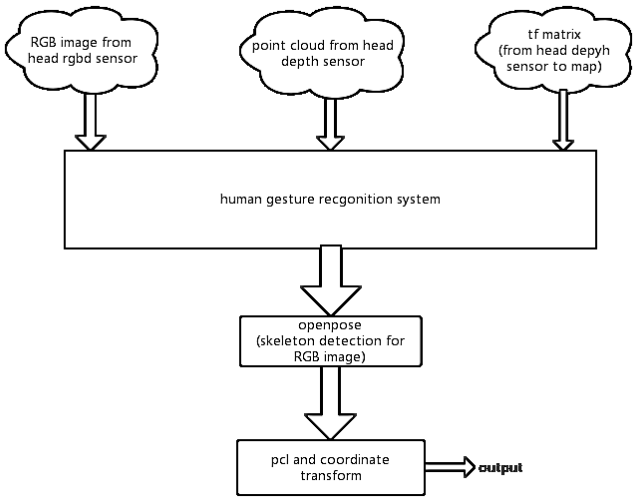

Fig. 4. Human gesture detection system

Fig. 5. CMU OpenPose skeleton detector

3.5 Robot arm and object manipulation

We are using TurtleBot Arm[9] for object manipulation (Fig. 6). It consists of 5 Dynamixel AX-12A servo motors, controlled by an *ArbotiX-M controller board/USB2Dynamixel*. We use *MoveIt!* as the arm software framework, and we have integrated the arm control with object detection by color detection[10] and object recognition by image processing for object manipulation. Once we recognized the object, we perform object localization by 3D point cloud to obtain the position of the object and calculate the inverse kinematic to make the movement of the arm to grasp the object. Also, by using *MoveIt!* we can plan the arm movement including obstacles avoidance to avoid collision with the surrounding objects.

Elevated arm. An elevated arm (Fig. 7) is developed for flexible height manipulation. The current design is target to enable object manipulation at the height ranges from 0.3m to 1.8m.

Fig. 6. Robot arm for object manipulation

Fig. 7. Elevated arm for flexible height manipulation

[9] http://wiki.ros.org/turtlebot_arm
[10] http://wiki.ros.org/cmvision

3.6 Simulation Development with SIGVerse

In order to speed up our robot development, we are also developing robot simulation with SIGVerse[11]. SIGVerse is a robotics simulator that can simulate human-in-the-loop human-robot interaction, capable of representing various task scenarios in RoboCup @Home.

Refer to Fig. 8, we have developed a virtual Handyman task that resembles the GPSR task in @Home. We develop and improve our gesture detection system in the virtual Interactive Clean Up task for better human gesture recognition. In the virtual human navigation task, we train our robot system to understand the sentence generation in GPSR. We also use the power of simulation system to conduct repetitive robot learning of huge amount of data via crowdsourcing.

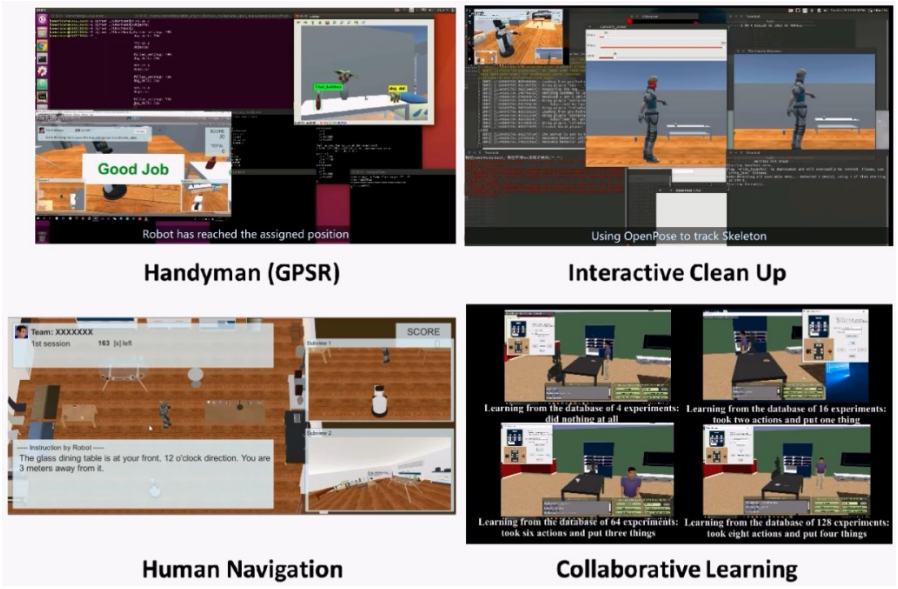

Fig. 8. Simulation development with SIGVerse

4 Conclusions

4.1 Open source robot platform for service robotics

This work aims to utilize open source solutions for both hardware and software to develop an open source robot platform for service robot research and development. The developed robot platform is open sourced with support wiki, source codes on GitHub and 3D printing parts to ensure easy reproducibility, to build up a community-driven development effort for service robots.

[11] http://www.sigverse.org/

- Website: **http://openbotics.org/kamerider/**
- Learning resources: **http://www.robocupathomeedu.org/learn**
- Source codes: **https://github.com/robocupathomeedu/**
- Demo Videos: **https://www.youtube.com/user/kameriderteam**

4.2 Participation in RoboCup competitions to benchmark the open source robot platform performance

On July 2015, we had participated for the first time in international RoboCup @Home in RoboCup 2015 Hefei, China with the open source robot platform (Fig. 9). With the overall ranked 7th result, it has proven the potential of the developed open source robot platform in RoboCup @Home application. Despite the increasing challenging of RoboCup @Home competition each year, we managed to maintain the 7th position in RoboCup 2016 Leipzig, Germany.

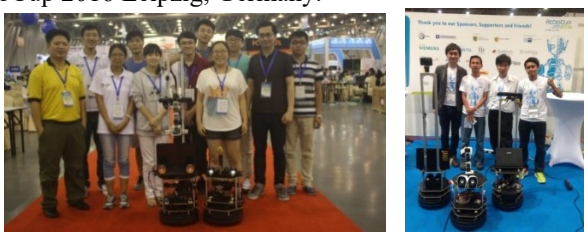

Fig. 9. RoboCup 2015 Hefei, China (left) and RoboCup 2016 Leipzig, Germany (right)

4.3 Continuous Social Contribution: Support the Educational Initiative – RoboCup@Home Education

Along with our robot development, we are supporting the RoboCup@Home Education initiative to promote service robot development to entry level audience via RoboCup@Home. Under this initiative, currently there are 3 efforts in operation:

1. RoboCup@Home Education Challenge
2. Support the development of open source educational robot platforms for RoboCup@Home (service robotics)
3. Outreach Programs (domestic workshop, international academic exchange)

References

[1] Thomas Wisspeintner, Tijn van der Zant, Luca Iocchi and Stefan Schiffer, “RoboCup@Home: Scientific Competition and Benchmarking for Domestic Service Robots”, *Interaction Studies*, Vol.10, No.3 (2009), pp.392-426.

[2] Tijn van der Zant and Luca Iocchi, “Robocup@Home: Adaptive Benchmarking of Robot Bodies and Minds”, *Social Robotics*, (2011), pp.214-225.

[3] Quigley, Morgan, Ken Conley, Brian Gerkey, Josh Faust, Tully Foote, Jeremy Leibs, Rob Wheeler, and Andrew Y. Ng. "ROS: an open-source Robot Operating System." In *ICRA workshop on open source software,* vol. 3, no. 3.2, 2009.

Annex

Name of team: KameRider OPL
Contact Information: i@jeffreytan.org
Website: http://openbotics.org/kamerider/

Hardware:

- TurtleBot 2 (Kobuki base)
- Microsoft Kinect Xbox 360
- Asus Xtion Pro
- Dynamixel AX-12A servo motors (TurtleBot arm)
- Laptop PC with NVIDIA GTX1070 GPU
- PC microphone and speakers
- Digital camera

Software:

- ROS (Robot Operating System)
- CMU Pocket Sphinx
- HARK
- OpenCV
- Baidu-AI
- Tensorflow and Keras
- OpenPose