

# KameRider UTHM @Home Education 2024 Team Description Paper

Jeffrey Too Chuan Tan, Mohammad Afif Bin Ayob, Abu Ubaidah Bin Shamsudin,  
En. Hazwaj Bin Mhd Poad, Herdawatie binti Abdul Kadir, Abishai Asir A/L  
A.Thederajan, Lee Zhan Xiong, Yong Huei Jean ,Ahmad Adham Bin Azhar and  
Ahmad Idris Yakubu

University Tun Hussein Onn,  
Malaysia

ubaidah@uhm.edu.my

[https://sites.google.com/view/robocup  
uthm/home](https://sites.google.com/view/robocup<br/>uthm/home)

**Abstract.** This document is the team description paper of the KameRider UTHM team for the participation of @Home Education in RoboCup 2024, Netherlands. This paper describes an artificial intelligent robot that is focusing on an indoor environment that can map the environment using a ros gmapping package. Using the recorded map, a navigation algorithm is used to move to a given goal. The navigation algorithm is combination of Advance Monte Carlo for localization, A\* path-planning for global planner and Dynamic Window Approach for local planner. Python google speech recognition package is used to convert word to text and python google speech synthesis is used to convert text to sound. For recognition, YOLO 7 real-time object detection is used to recognize objects. The robot is designed as an autonomous robot that is complete with the contribution of combining the MNRS which are mapping, navigation, object recognition and speech recognition. Our research also provides software system and hardware system design development that are implemented in our robot.

## 1. Introduction

### a. Team

As an introduction, our group is from Malaysia that represents University Tun Hussein Onn Malaysia (UTHM). We consisted of five group members. First of all, our team leader serves as the person who will install all the drivers into the robot. He is also the one who will lead our team to do all the tasks. Then, the other four group members serve as the people who will solve the coding and hardware parts assembly for the robot. The assigned tasks will be divided among the team members and should be completed according to the stipulations. If there is a problem, other team members will try to help and solve it, so that the task can be completed before the set date.

Our team also has a website as a platform for anyone to see our project and activity. Our team joined the last competition which is RoboCup @ Home Education 2023 Bangkok, Thailand (online challenge) and awarded for the First Place out of eight competitor. Our team also participated in (on-site challenge) for RoboCup @ Home category and ranked as fifth place out of tenth finalist.

## 2. Project Planning

### a. Overall project plan

Our robot design uses Turtlebot Kobuki as the mobile robot platform. Turtlebot Kobuki is Turtlebot2 type robot consists of Yujin Kobuki base, a 2,200mAh battery pack, realsense depth camera D435, orbecc Astra Mini S, an ROG Srtix G51 laptop, fast charger, OpenMANIPULATOR-X and a hardware mounting kit attaching everything together and adding future sensors. Turtlebot2 was released in Oct 2012 [1].

Our group started to create the coding for the three tasks from September 11, 2023 until November 23, 2023. The competition was made to select the team that would represent UTHM. For now, our team has managed to run all the tasks but we are still making some improvements to make it run smoothly either in the simulation or real world. However, our team still faced some constraints sometime but with the analysis done during the simulations in the real world and the gazebo world, it can help our group in making improvements. Thus, the robot can be adapted in various environments and able to do the task as well as possible even if unexpected things happen.

### b. Integration plan

To make things run accordingly, our team has developed our own Gantt chart. This chart allows us to review the overall plan to make this robot more efficient and organized. The first thing our team does is try to do a simulation on Gazebo World. This allows us to detect flaws in the robot at an early stage of development. After we were done with the simulation, we started to apply the program to the real-world robot by installing all the required drivers in order to operate the robot. Finally, we test the simulation program on our real-world robot and try to do a stress test to allow us to detect any problem during performing the task.

### c. Testing

In this challenge, our group conducted several simulation tests using Gazebo to make sure that all the programs that we created are running smoothly. We can gather the data of the robot by using RViz to know what the robot sees and where the data comes from. Also, we conduct many tests on python files in order to make robots move autonomously and smoothly.

There are three tasks in this challenge. Firstly, in Carry My Luggage, the robot must demonstrate bag detection and recognition capability. The robot will be guided by an unknown user who can command the robot by speech or gestures. Then, the robot will bring the bag and follow the user to the car. As a bonus reward, the robot needs to re-enter the arena and back to the starting point. If the robot can avoid any obstacle there, it will also take the bonus reward. Here, the robot will be tested whether it can follow the speech and gesture command correctly or not. Next, in Find My Mate, the robot needs to find a guest. When the robot finds the guest, the robot needs to recognize where a guest is. Then, the robot will go to the operator and tell the robot where the guest is. Here, the robot will find two mates for the user and as a bonus reward, the robot needs to find another one mate. So, the robot will be tested whether it can recognize where the guest is or not.

In the last task, Receptionist, the robot needs to ask the guest's name and drink. The robot also needs to introduce the guest to other guests and pointing at the guest when introduced. In addition, the robot needs to detect an empty seat and the robot needs to be pointing at an empty seat while offering it. In this task the robot needs two guests. For bonus reward, the robot needs to open the entrance door to a guest and sit the oldest person on a sofa.

All the tasks make us explore more about speech recognition and YOLO V8 detection. It is very useful to test it on our robot. Our group also needs to analysis the test results. The test needs to be done several times to ensure that we get the desired output. Sometimes during the test run, it cannot obtain the same output due to the low internet connection or having other problems, thus, will affect its development. It is important for our team to make a test and analysis the robot.

### 3. Software

#### a. General software architecture

##### 1. Mapping

**Gmapping** is one of the algorithms under the SLAM method that are widely used in mapping an environment [2]. Mapping starts when we run *gazebo*, *keyop*, *gmapping*, and *RViz* command. When mapping our environment, we moved the robot using the *keyop* command. To get an approximate map, we need to be careful in this step so that the Turtlebot 2 does not hit any objects or wall on their way in mapping. To see what the robot sees in its camera we can run *RViz* command. After the mapping is complete, we need to save the map so that the Turtlebot 2 can move autonomously in thenavigation stack.

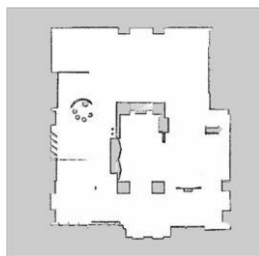


Figure 3.1 shows the mapping using LIDAR laser scan

##### 2. Navigation

**AMCL (Adaptive Monte Carlo Localization)** is one of the other algorithms under SLAM method in mapping and navigation. AMCL is the expectation of exact location for the robot to move in 2 dimensions [3]. **A\*** is route planning for the robot to avoid hitting with the obstacles from the starting point to the end point [4]. **DWA (Dynamic Window Approach)** a path planning based on robot dynamic motion [5]. The Turtlebot 2 navigation is used for robot navigation with map building using *gmapping* and *amcl* while running the navigation stack in ROS. With a complete map in the *gmapping* we can predefine location point that the robot need to move when instructed.

In *RViz*, when starting up with the navigation stack, we need to localize the Turtlebot 2 using *2D Pose Estimate* button because the Turtlebot 2 does not know where are the location of their actual point. Thus, we need to provide the robot its approximate location on the map using this first step. When the Turtlebot 2 is localized, it can move autonomously through the map using *2D Nav Goal*.

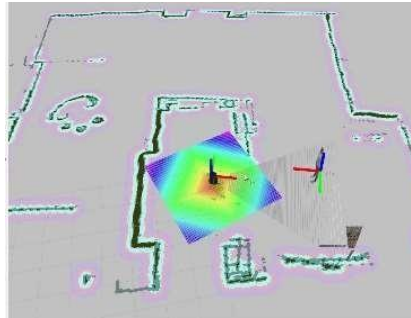


Fig. 3.2. shows the navigation of Turtlebot 2

### 3. Object Detection

**YOLO Real-Time Object Detection** is a state-of-the art, real-time object detection [6]. It is one of the fastest and most accurate object detection frameworks. It applies a single neural network to the full image. This network divides the image into regions and predicts bounding boxes and probabilities for each region. These bounding boxes are weighted by the predicted probabilities. The pre-trained model of the convolutional neural network is able to detect pre-trained classes including the data set from VOC and COCO, and also can create a new network with new detection objects. Meanwhile, this object detection can be applied to robots by using yolo V8. It provides 2d bounding boxes and depth of the objects contained in an objects list, where the 3d position of each object is specified [7]. This program will subscribe to the camera depth topic of robots in order to obtain data. To start object detection, simply run the program to begin object detection. The yolo V8 will provide the location of the detection and classes of the detection.

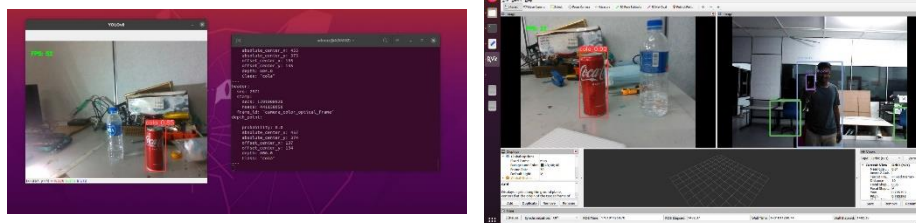
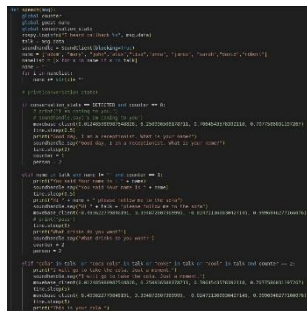


Figure 3.3 show the image in front Kobuki TurtleBot2's camera

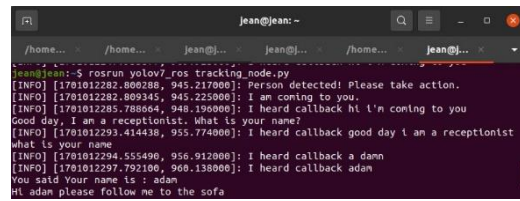
## 4. Speech Recognition

The sound is recorded and saved into speech. By using this speech, ambient noise had been adjusted to get a better sound before we put them into audio. Next, the audio was sent to Google Speech recognition. Language used in our audio was 'en- US'. We started to run the speech by using *python3 trackingnode.py*. At this point, the detected audio will print out as a sentence as shown in Figure 4.



```
python3 trackingnode.py
[INFO] [171012282.809345, 945.225000]: I am coming to you.
[INFO] [171012285.788064, 948.196000]: I heard callback hi l'm coming to you
Good day, I am a receptionist. What is your name?
[INFO] [171012293.414338, 955.774000]: I heard callback good day t am a receptionist
what is your name
[INFO] [171012294.555490, 956.912000]: I heard callback a damn
[INFO] [171012297.792100, 968.130000]: I heard callback adan
You said Your name is : adan
HI adan please follow me to the sofa
```

Input

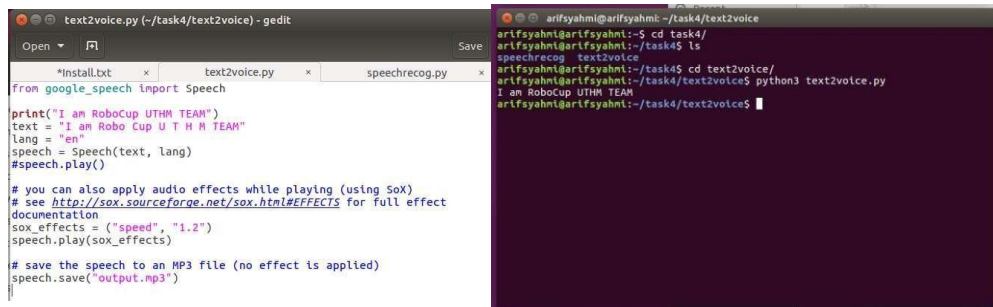


```
jean@jean: ~
/home... /home... jean@j... jean@j... /home... jean@j...
jean@jean:~$ rosrun yolov7_ros tracking_node.py
[INFO] [171012282.809345, 945.225000]: Person detected! Please take action.
[INFO] [171012285.788064, 948.196000]: I am coming to you.
[INFO] [171012293.414338, 955.774000]: I heard callback good day t am a receptionist
what is your name
[INFO] [171012294.555490, 956.912000]: I heard callback a damn
[INFO] [171012297.792100, 968.130000]: I heard callback adan
You said Your name is : adan
HI adan please follow me to the sofa
```

Output

Figure 3.4.1 shows the input and output from audio to sentences

The next stage is converting text to voice. The python3 had been installed from the attached file named text2voice from the same google drive link. From the same file we opened the programmed file and imported *google speech* into speech. The languages used are in English, "en" and the speed adjusted to "1.2". The voice then can be heard clearly. Lastly, we saved the speech to an MP3 file named output.mp3.



```
text2voice.py (-/task4/text2voice) - gedit
*install.txt x text2voice.py x speechrecog.py x
From google_speech import Speech
print("I am RoboCup UTHM TEAM")
text = "I am RoboCup U T H N TEAM"
lang = "en"
speech = Speech(text, lang)
#speech.play()

# you can also apply audio effects while playing (using Sox)
# see http://sox.sourceforge.net/sox.html#EFFECTS for full effect
documentation
sox_effects = ("speed", "1.2")
speech.play(sox_effects)

# save the speech to an MP3 file (no effect is applied)
speech.save("output.mp3")

arifsyahmi@arifsyahmi:~/task4/text2voice
arifsyahmi@arifsyahmi:~$ cd task4/
arifsyahmi@arifsyahmi:~/task4$ ls
speechrecog text2voice
arifsyahmi@arifsyahmi:~/task4$ cd text2voice/
arifsyahmi@arifsyahmi:~/task4/text2voice$ python3 text2voice.py
I am RoboCup UTHM TEAM
arifsyahmi@arifsyahmi:~/task4/text2voice$
```

Input

Output

Figure 3.4.2 shows the input and output from text to voice

## b. Flowchart

Figure 3.5 (a) shows the flowchart for task 1, which is Carry My Luggage. The first thing a robot must demonstrate is bag detection and recognition capability where the robot starts searching the bag using YOLO 5 object detection. If the robot finds the bag, the robot arm will start the pick and place pose to pick the bag. Then, the robot will bring the bag and start following the user to the car. The robot then re-enters the arena and go back to the starting point if the user says “Stop” indicates the task has been completed.

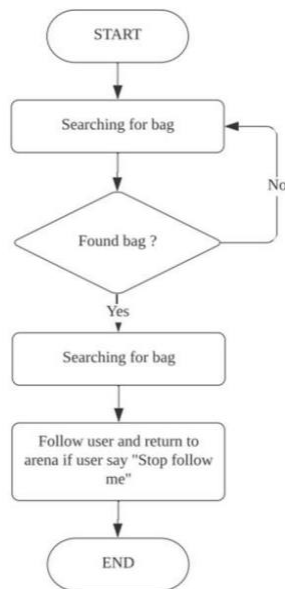


Figure 3.5 (a) shows flowchart for task 1

Next, for task 2, Find My Mate, the robot needs to find a guest. By referring to Figure 3.5 (b), it starts by searching for the people on the checkpoint. When the robot detects a person, it starts to move toward the person and begins asking for their personal information such as name, age, and etc. After that, the robot will go to the next checkpoint and repeat the same task by asking for personal information until it finds a 3rd person or goes to the last checkpoint. After doing that, the robot will go back to the user and tell all the information the robot has gathered including their location.

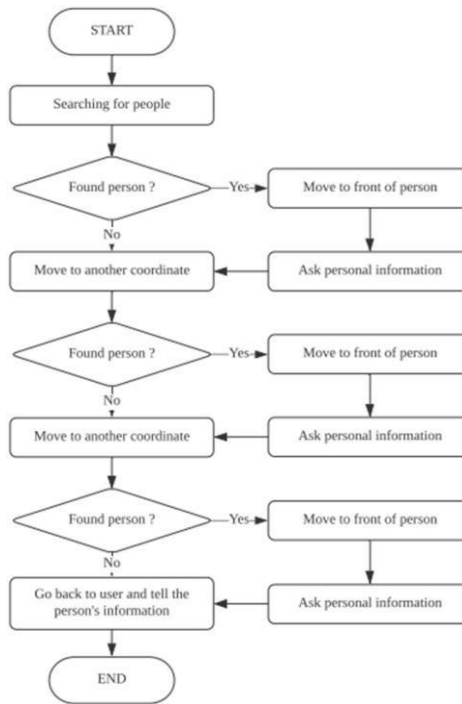


Figure 3.5 (b) shows flowchart for task 2

In the last task, the Receptionist on Figure 3.5 (c), If the robot finds the person, the robot arm will start the interacting with the person and asking “Do u need help?” If the user says “Yes”, the robot will redirect the person to specify location according to that person description. If the user says “No”, the robot then go back to the starting point indicates the task has been completed.

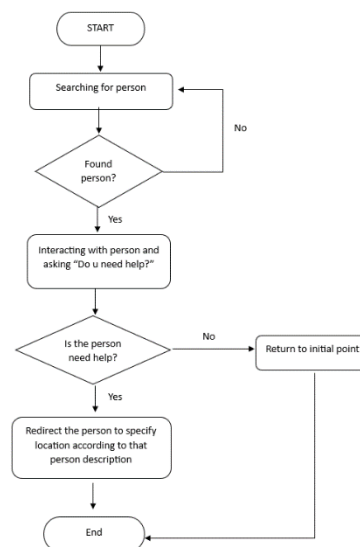


Figure 3.5 (c) shows flowchart for Task 3

## 4. Hardware

To make our robot move is a crucial part of robotic design. Here, mechanical design and manufacture are needed as a main structure for our robot. Even if the designing mechanism is more important in machine design, but by creating a robot also focused on it. Other than that, electronic design and manufacturing are also important. Understanding electronics will allow our group to make the right choice in the electronic part. Both of these things will make our robot work.

### a. Mechanical design and manufacturing

For the main structure of our robot, we use **Turtlebot Kobuki** as shown on figure (a). Its structure is very good for autonomous navigation. It also comes with built in odometry, gyro and cliff sensor. This robot uses a 14.8V, 2200 mAh battery to power its motor. Also, it can operate for 3 hours on a single charge. Meanwhile, the maximum translational velocity can reach 70cm/s and the maximum rotational velocity is 180 deg/s. In addition, it can carry a payload of up to 5 kg on hard floors and also 4 kg on the carpet. The flexibility allows this robot to operate anywhere, especially in the home environment.

For pick and place tasks, the **OpenManipulator-X** as shown on figure (b). OpenManipulator-X has 5 DOF (4 DOF + 1 DOF Gripper). It has a payload of 500g and a repeatability of less than 0.2mm. The robot has a weight of 0.70 kg (1.54 lb) and a reach of 380 mm (14.9 in). The gripper stroke is 20~75 mm (0.79~2.95 in). The robot is compatible with TTL Level Multidrop BUS and supports ROS, DYNAMIXEL SDK, Arduino, Processing.



Figure 4.1 shows the (a) Turtlebot Kobuki (b) OpenManipulator-X

### b. Electronic design and manufacturing

For our robot sensor, we use the Intel RealSense. It is a suite of depth sensing cameras that can be used for a variety of applications, including robotics, 3D scanning, and augmented reality.

RPLIDAR A2 is a LIDAR sensor that is used to perform **Simultaneous localization and mapping (SLAM)**. It provides a 360-degree scan field, 5.5hz/10hz rotating frequency with a guaranteed 16-meter range distance. The robot will read RPLIDAR raw scan results using RPLIDAR's SDK and convert it to ROS LaserScan message. The scan result is much better and accurate than using conventional RGB depth to do mapping.

Orbbec Astra Mini S has a range of 0.35m to 1m, a field of view of 60°H x 49.5°V x 73°D, and RGB and depth image resolutions of 640 x 480 @30fps. This device is also energy-efficient, consuming less than 2.4 W of power. For the brain of our robot, we use a laptop to control all the sensors and arm. Also, it can operate realsense on the laptop and allow use of all provided software.



To power the robot, we use a Lithium Polymer (LiPo) battery. This battery is used to power all components of the robot such as the Turtlebot2 wheel. These batteries provide higher specific energy than other lithium battery types and are long lasting. To suit our robot requirement, we decided to use 11.1 V with 5500 mAh that can last for 3 hours and we need 3 of it.

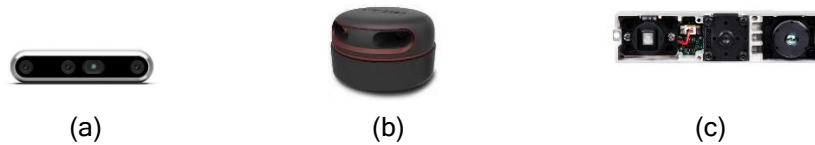


Figure 4.2.1 shows the (a) realsense (b) RPLidar A2 (c) Orbbec Astra Mini S

## 5. Performance Evaluation (Result)

For overall performance of the robot, the robot seems to move accordingly to our plan on the Gazebo simulation. In the real world, all the components and parts can operate and execute well. However, for the autonomous task, it seems that the robot is unable to perform perfectly like the commercial robot. It still needs some improvement on the source code and some humanity value to make the robot more friendly with the human environment. Moreover, the robot must go through multiple stress tests in order to make the robot adapt to multiple environment setups and still can perform a task accordingly.

## 6. Discussion and Conclusion

To sum it up, we presented on how to integrate *MNRS* software to be used for the RoboCup@Home Education Netherlands 2024 competition. We implement Kobuki simulation as a service robot. Throughout this competition, it gives us the experience to think outside the box to troubleshoot unexpected problems that inhibit autonomous programs to perform a task during the competition and give a chance to compete with the other experienced team. For future work, we hope that we can inspire our young generation to get involved in this industry and be able to use robotic knowledge to help grow our nation.

## 7. Acknowledgements

The completion of this undertaking could not have been possible without the participation and assistance of so many people whose names may not all be enumerated. Their contributions are sincerely appreciated and gratefully acknowledged.

Our team would like to express our sincere gratitude to our supervisor, Dr Abu Ubaidah, Dr Herdawatie and Mr. Hazwaj for allowing our team to represent UTHM for this year's competition. It is an honour that our team is able to join this competitive international competition. Also, special thanks to Ts. Dr. Mohammad Afif bin Ayob for providing us 3D printing specialists.

To Universiti Tun Hussein Onn Malaysia (UTHM), we thank the university for providing financial support and encouragement to our team. In addition, providing us a conducive workshop for our team to do research and development of our robot.

Last but not the least, our parents are also an important inspiration for our team members. So with due regards, we express our gratitude to them.

## 8. References

- 1) Evan Ackerman (2012), "What is Turtlebot?", IEEE Spectrum, <https://www.turtlebot.com/about/>
- 2) [Norzam](#), W. A. S., Hawari, H. F., & Kamarudin, K. (2019), "Analysis of Mobile Robot Indoor Mapping using GMapping Based SLAM with Different Parameter", *IOP Conference Series: Materials Science and Engineering*, 705,012037. <https://doi.org/10.1088/1757-899x/705/1/012037>
- 3) Brian P. Gerkey (2020), "amcl", ROS.org, <http://wiki.ros.org/amcl>
- 4) *Path planning*. (2021). Husarion Docs. <https://husarion.com/tutorials/ros-tutorials/7-path-planning/>
- 5) D. Fox, W. Burgard and S. Thrun, "The dynamic window approach to collision avoidance", in IEEE Robotics & Automation Magazine, vol. 4, no. 1, pp. 23- 33, March 1997, doi: 10.1109/100.580977.  
<https://www.trossenrobotics.com/interbotix-turtlebot-2i-mobile-ros-platform.aspx>
- 6) Joseph Chet Redmon (2013), "YOLO: Real-Time Object Detection", Darknet: Open Source Neural Networks in C, <https://pjreddie.com/darknet/yolo/>
- 7) Francisco Martín Rico, Fernando González Ramos (Aug 18, 2020), *gb\_visual\_detection\_3d*, IntelligentRoboticsLabs  
[https://github.com/IntelligentRoboticsLabs/gb\\_visual\\_detection\\_3d](https://github.com/IntelligentRoboticsLabs/gb_visual_detection_3d)