

AAST-CAI@Home 2022

Team Description Paper

Farah El Soussi, Nour eldin Morsy, Nada Samir, Yasser Salem, Amira Abu El Yazed
Mostafa Magdy, Adham Sakr, Maria Roshdy, Ahmed Abdelhady, Abed Ibrahim

Abstract. This paper is about an intelligent robot called AAST-CAI. In the near future we can see robots everywhere to help humans either in houses or in public; that is what inspires us to develop AAST-CAI. To clarify more, AAST-CAI is implemented to be a home assistant, to work in a domestic environment with autonomous navigation and smart-simple interaction. Aiming to participate in Robocup@home competitions, that made it more motivating for us. And to compete and perform well, this paper will specify our effort in software and to overcome the problems faced.

1. Introduction:

Our team "AAST-CAI" is an Egyptian team from AASTMT University College of Artificial Intelligence. We have been working hard to improve our "TurtleBot" robot for the next Thailand RoboCup@Home 2022, which will be our first participation in the RoboCup@home competition. The team used the ROS "Robot Operating System" to improve the functionality of the "TurtleBot" by developing new applications for the robot to make it a human assistant that works in specific human environments and can assist them with specific tasks, which it can do through navigation, manipulation, computer vision, and speech recognition.

This TDP is part of the qualification requirements for Thailand's robocup@home 2022.

2. Tasks

We needed to complete the carry my luggage task that would allow the robot to transport a specific bag that is being carried by the operator from one location to another. The robot will carry the bag and follow the operator outside the arena to the car, then the robot will hand in the bag to the operator and go back home after performing the task. Operator uses voice commands to control the robot, and by using navigation, robot will be able to navigate the arena in the fig.1.

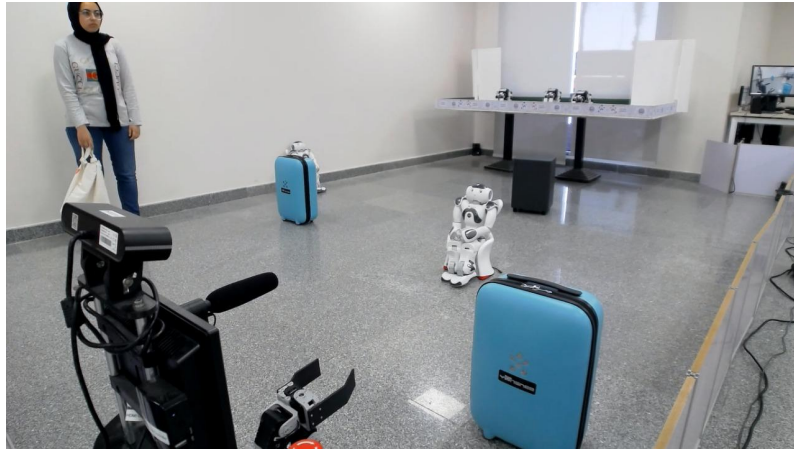


Fig. 1: The RoboCup@Home Arena at AAST-Alamean.

3. Developed Solution

We developed an algorithm to solve the given tasks. In the architecture of the algorithm, it includes google_sr, arm, partybot, navigation, face_detection, and follow nodes that covers our five main tasks: speech, manipulation, navigation, face detection, and following. We created script files including all the terminal ROS commands to construct the packages for each node. We will talk about every node in detail.

This the script that initializes all the five scripts in fig.2 and each script represents the sub components in ROS:

```
#!/bin/sh
# export TURTLEBOT_GAZEBO_MAP_FILE=/home/nvidia/catkin_ws/src/worlds/kinect-map.yaml
xterm -e "~/catkin_ws/sh/face_detection.sh" &
sleep 20
xterm -e "~/catkin_ws/sh/speech.sh" &
sleep 20
xterm -e "~/catkin_ws/sh/navigation.sh" &
sleep 20
xterm -e "~/catkin_ws/sh/arm.sh" &
sleep 20
xterm -e "~/catkin_ws/sh/follow.sh"
```

Fig. 2: Screenshot of the script that initializes all scripts .

3.1 Speech:

It represents interaction between the person and the robot. It is divided into two parts: google speech recognition for speech recognition and partybot for chatbot.

- **Speech script:**

This is the script file in fig.3 for initializing the google_sr and partybot nodes from “rchomeedu_speech” and “rchomeedu_partybot” packages .

```
#!/bin/sh
# export TURTLEBOT_GAZEBO_MAP_FILE=/home/nvidia/catkin_ws/src/worlds/kinect-map.yaml
xterm -e "roscore" &
sleep 5
xterm -e "roslaunch rchomeedu_speech google_sr.py"
sleep 5
xterm -e "roslaunch rchomeedu_partybot partybot.py"
sleep 5
xterm -e "roslaunch rchomeedu_partybot partybot1.py"
```

Fig. 3: Screenshot of Speech script.

- **google_sr node:**
This node is considered as master node as it publishes commands to all the other nodes, can send to multiple nodes at the same time, for example when it detects a “Put” command it publishes to partybot to answer back, and it also publishes to the arm to put the object down.
- **partybot node:**
This node is divided into two parts: the first part subscribes from “google_sr” the detected speech commands then it answers back with the suitable answer from a set of conditions in the “partybot.py” file, the second part subscribes from “face_detection” the detected face name then it greets the person.

3.2 Manipulation:

This part is manipulating detected objects by the robot arm depending on the voice command.

- **Arm script:**
This is the script file in fig.4 for initializing the arm node and it activates the 6 DOF servo motors from the “rchomeedu_arm” package.

```
#!/bin/sh
# export TURTLEBOT_GAZEBO_MAP_FILE=/home/nvidia/catkin_ws/src/worlds/kinect-map.yaml
xterm -e "roslaunch rchomeedu_arm arm.launch" &
sleep 5
xterm -e "roslaunch rchomeedu_arm arm3.py"
```

Fig. 4: Screenshot of Arm script.

- **Arm node:**
It subscribes from google_sr node. We created a dictionary including the set of coordinates of the servos of some certain positions such as “Grab” and “Put” in fig.5.

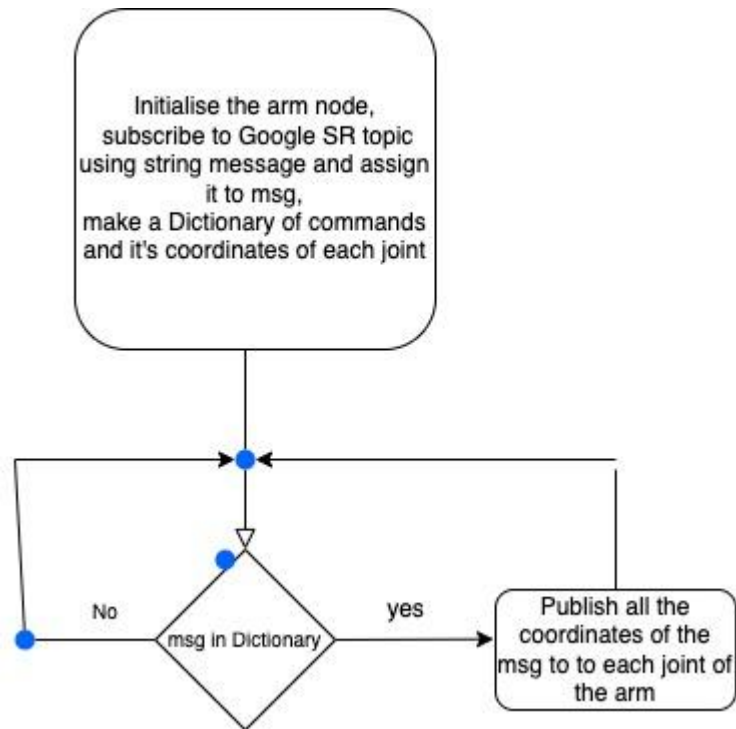


Fig. 5: Flow Chart of Arm algorithm.

3.3 Navigation:

This part will make us control the robot to navigate on a pre-built map in fig.6 by using the lidar and base motors.

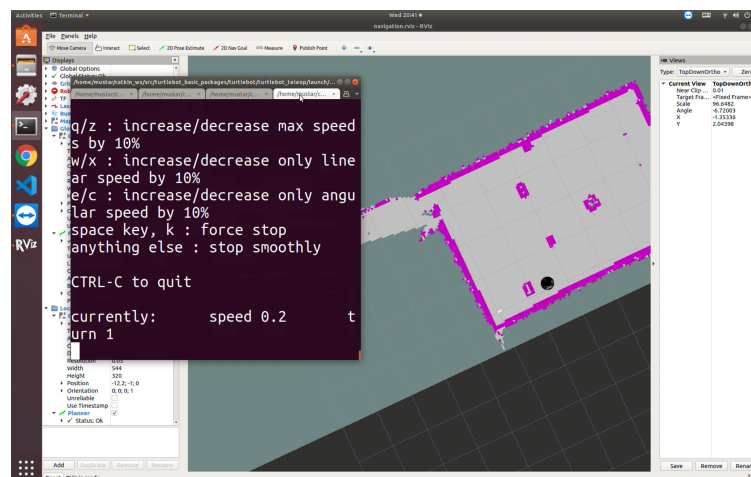


Fig. 6: The RoboCup@Home Arena in RVIZ.

- Navigation script:**
 The script in fig.7 that initializes the navigation node and brings up the turtlebot base, the lidar, and the pre-scanned map.

```
#!/bin/sh
# export TURTLEBOT_GAZEBO_MAP_FILE=/home/nvidia/catkin_ws/src/worlds/kinect-map.yaml

xterm -e " roslaunch jupiterobot_bringup jupiterobot_bringup.launch" &
sleep 5
xterm -e " roslaunch jupiterobot_navigation rplidar_amcl_demo.launch map_file:=/home/mustar/catkin_ws/maps/challenge.yaml" &
sleep 5
xterm -e " roslaunch turtlebot_rviz_launchers view_navigation.launch" &
sleep 5
xterm -e " rosrun rchomeedu_navigation navigation7.py "
```

Fig. 7: Screenshot of Navigation script.

- **Navigation node:**

It subscribes from google_sr, the navigation source code was implemented to take the point coordinates from the terminal, but instead, we created a dictionary including a set of coordinates of the map of some certain positions such as “Home” and “Base” in fig.8.

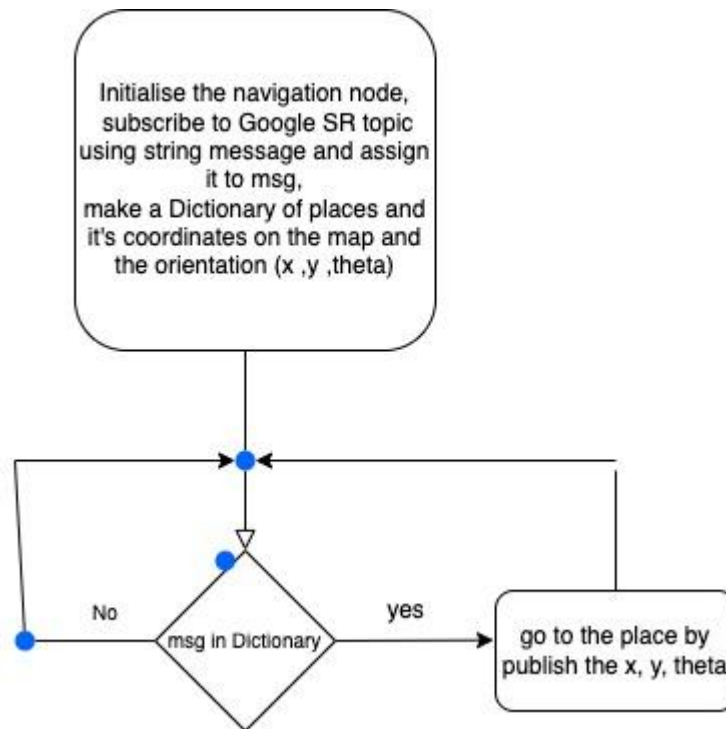


Fig. 8: Flow Chart of Navigation algorithm.

3.4 Face detection:

The part where the robot will recognize faces that was trained before by using the astra camera.

- **face_detection script:**

The script in fig.9 of face_detection node: activates the astra camera from “rchomeedu_vision” package and initializes the face recognition node from “opencv_apps” package.

```
#!/bin/sh
xterm -e "roslaunch rchomeedu_vision multi_astra.launch image:=/camera_top/rgb/image_raw" &
sleep 5
xterm -e "roslaunch opencv_apps face_recognition.launch image:=/camera_top/rgb/image_raw"
```

Fig. 9: Screenshot of Face detection script.

- **face_detection node**

It subscribes from google_sr node. We trained the model to recognize some of our team members and added their names in a dictionary so after that it could greet them once as soon as the robot detected their faces in the partybot node in fig.10.

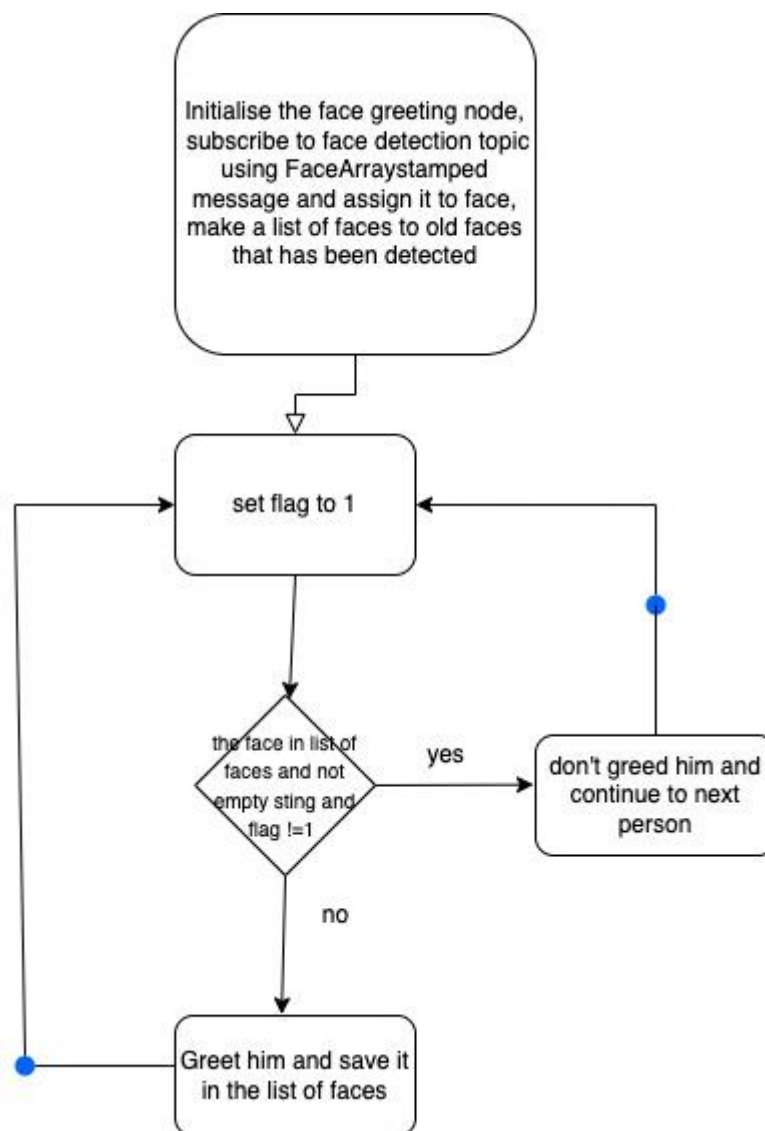


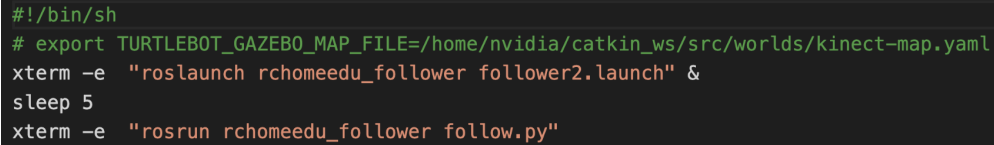
Fig. 10: Flow Chart of Face detection algorithm.

3.5 Following:

This part where the robot starts following and tracks the person in front of him by using the astra camera.

- **The follow script:**

The script in fig.11 initializes the following node and brings up the astra camera.



```
#!/bin/sh
# export TURTLEBOT_GAZEBO_MAP_FILE=/home/nvidia/catkin_ws/src/worlds/kinect-map.yaml
xterm -e "roslaunch rchomeedu_follower follower2.launch" &
sleep 5
xterm -e "roslaunch rchomeedu_follower follow.py"
```

Fig. 11: Screenshot of Follow script.

- **The follow node:**

It subscribes from google_sr node, it has two different commands either to follow or to stop following.

4. Results

The Robot starts listening when activated. If it hears a certain line it will do the command that maps to the said line. These commands can access each part of the Robot (Navigator - Movement - Arm - Dialog - Camera).

4.1 Navigation - Movement

If the Robot Hears "Follow me", it will activate the upper camera and start navigating towards the closest person.

If the Robot Hears "Go to [a point on the map]", it will be able to move Toward that exact point, using the lidar to avoid any obstacle.

4.2 Arm - Camera

If the Robot Hears "Pick up [object name]", it will activate the top camera and use the YOLO-CNN to detect-recognise the given object in fig.12, and it will manipulate the arm in a way to pick it up.

If the Robot Hears "Drop" or "Give", it will open the arm and drop the item.



Fig. 12: Object recognition using YOLO.

4.3 Camera

If the Robot Hears "Who is this", it will activate the top camera and start recognising faces, if the face is recognised it will say "This is [name]". If the Robot Hears "Add To Faces", it will take pictures of the detected face and add them then ask for the name.

4.4 Dialogue

The Robot can answer questions like: "What time is it" - "What is the weather like today".

The Robot is connected to the web to get word meanings and general information.

5. Conclusion

In this TDP, we described the features of our team "AAST-CAI" and all the efforts we have put into our goal of making the turtlebot robot capable of being an assistant robot that can perform many tasks, considering safety and ease of use in mind. We have chosen the robocup@home competition to showcase all the work we've done and to share the experience with other teams. Our goal in the future is to further improve our robot, "making it a great helper for those in need." In the end, we are very proud to be associated with this global competition that pushes people to work harder and be more innovative in technology.

6. Future work

Our team "AAST-CAI" have an ambition that will begin to develop on "Pepper Robot" for the next RoboCup@Home competition and also we will start to build our own Robot for the next Robotcup@Home Open platform.

References

1. Robots/TurtleBot - ROS Wiki, <https://www.turtlebot.com>

2. NAvigation - ROS Wiki, <http://wiki.ros.org/navigation>
3. Yolo - darknet, <https://pjreddie.com/darknet/yolo/>
4. Google speech recognition -Google API, [Google APIs Explorer](#) | [Google Developers](#)
5. pocketsphinx - ROS Wiki, <https://cloud.google.com/speech/>
6. pocketsphinx - ROS Wiki, [pocketsphinx · PyPI](#)