RoBorregos Team Description Paper for RoboCup @Home 2024

Adan Flores Emiliano Flores Ivan Romero Alexis Chapa
Francisco Salas Alejandra Coeto David Vazquez
Diego Hernandez Oscar Arreola Yair Reyes
Alejandro Guerrero Marina Villanueva Angel Cervantes
Alberto Munoz Carlos Vazquez

November 27, 2023

Instituto Tecnologico y de Estudios Superiores de Monterrey
C.P. 64849, Monterrey, N.L., Mexico
`https://docs.rbrgs.com/home`

Abstract. This paper serves to present the current status of *RoboTeus*, a service robot developed by the RoBorregos team at ITESM. Our robot is assembled with a mobile base and a 6-DOF robotic arm, integrated with custom electronics. Our current approach focuses on developing functional pipelines for each module, implementing stable libraries with state-of-the-art technologies and optimizing and developing features with language-level solutions. Then, each pipeline is integrated through a Main Engine that receives and send tasks through a network of different embedded systems. The main contributions of the team this season, are the development of the Object Recognition pipeline, with a complete proccess from dataset automatic generation, to accurate vision models and the HRI pipeline integrating generative AI. Also, the developments in manipulation with the hardware modifications made, are being used in our Institution for teaching in classes.

1 Introduction

The RoboCup @Home league aims to develop service and assistive robot technology with high relevance for future personal domestic applications. Our team, RoBorregos, was created in 2015 as the robotics team for our institution integrated by undergraduates, and it has been developing for this competition for almost five years. Since its inception, the team has a focus on autonomous competitions, it has won national and international awards in non-service robot competitions. However, the biggest project from the last years has been the service robots, making two appearances in the national competition @Home Beginners (2021 and 2022), earning recognition for the best human-robot interaction and object recognition systems, in 2021, the award for the most consistent solution was won at the international "AirLab Stacking Challenge", in 2022 it was achieved a 5th international place at IROS @Home Simulation category and recently, the team participated in the national Robocup @Home OPL.

Furthermore, the team has an active collaboration with our Institution School of Engineering, digesting software and hardware developments to its application in academic courses and industrial context. Two papers have been published for academic conferences such as EDUNINE and EDUCON [Avi+22] and [AVV22] describing the impact of these technologies in the learning proccess.

In this Team Description Paper, we present the recent research carried out by the team, directly related to achieving the skills required by a service robot and the tasks proposed by the RoboCup@Home competition.

2 Research focus and interests

This year, the team is composed of 15 B.Sc. students developing for the robot and the research, and 5 professors as advisors for the team in the research aspect. Our current development platform, with a mobile base (EAI Dashgo B1) and a 6-DOF robotic arm (xArm 6) was acquired at the beginning this year, hence, this year efforts were focused on developing a fully functional stable platform, adapting some previous advancements but mainly reformulating each software submodule. The principal achievements which are already being used in other robotics applications and for educational purposes, and are the Object Detection pipeline, as well as the electronics for the Base and the Manipulation Pipeline, further described in section 4.

3 Robot Architecture

Our robot's software architecture was developed in Robot Operating System using the Noetic distribution.

It is a behavior-based architecture centered in the *Main Engine* node. This node represents the decision level in the overall proccess, receiving commands from the voice input and directing specific actions to achieve the tasks to each submodule. The *Main Engine* is responsible for analyzing every task that needs to be executed and put a priority on them depending on the most important job on the moment (i.e. if the robot needs to navigate a room looking for a person, the navigation module has a priority to move the robot first and then look for the person using the Person Recognition module).

Each submodule pipeline operates with a principal Action Server, which can be stopped while in execution, and it provides useful feedback. Internally, there are additional libraries and scripts run for particular functions.

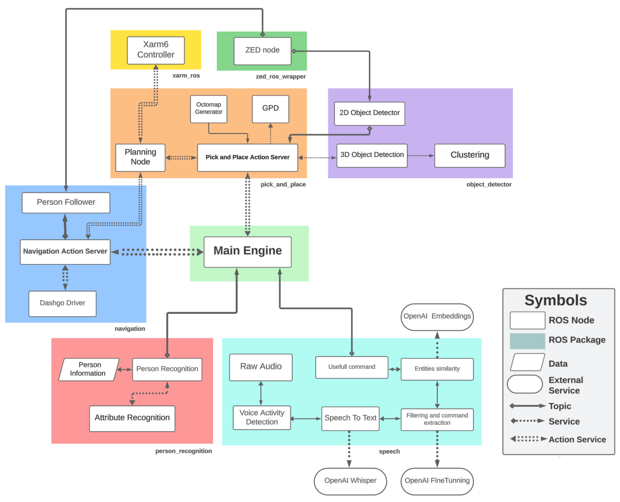

Fig. 1. Diagram of generalized interactions between software modules

4 Scientific contributions and current work

This section delves more into detail about the highlights of developments and achievements per area.

4.1 Object manipulation

This pipeline interacts with the 6-DOF robotic arm, and its gripper. Starts with consistent vision information from 2D object detection nodes as well as 3D Point Cloud information being processed for octomap generation. The robot uses *MoveIt!* for motion planning but the action servers for grasping and placing objects were developed from scratch, molded to the robot capabilities.

Picking Objects Grasping Pose Detector (GPD) is used to obtain picking poses, the pointcloud is processed using Point Cloud Library (PCL) to obtain the object as a 3D mesh. RANSAC [Martínez-Otzeta et al., 2022] is then run to find the table the robot is looking at and cutting it off from the pointcloud, which leaves the object meshes available for identification. GPD then obtains possible gripper positions to pick the object matched with the 2D detection position, and MoveIt is used to plan the trajectory considering the robot itself and the environment to avoid collisions.

Placing Objects Considering the possibilities of non-empty tables where the robot should place objects, two different methods have been developed to build possible placing positions. The base for the Pick using GPD and RANSAC still recycled.

The first approach is a language-level C++ algorithm that finds the best spot for a place position without additional libraries, it uses a hash function to map 3D world points into a boolean matrix. RANSAC then identifies the largest table, creating a boolean matrix for potential placements, predefined to consider the robot physical reach. PCL is used to identify object clusters above the table and removes them from the matrix. A prefix sum area matrix is then computed where each position represents the number of slots available for placing before it. As a placing near the center of the robot is preferred, a Breadth First Search algorithm is run, to find a position with area of minimum 1.5x the surface area of the object currently held. A Pose Stamped for the placing position is returned. Then, MoveIt is used for the planning of the trajectory towards the Pose given. When the position is reached, it opens the gripper, placing the object.

The second solution addresses complex scenarios, also considering all points in the point cloud reachable by the Xarm6. RANSAC then isolates the table point cloud, sent to a clustering server, and considers the objects observed in the filtered area sending as well the number of objects present in that area. The server processes the point cloud received as a 2D scatter plot, over which it runs a K-Means clustering algorithm, which resulting clusters are used to evaluate the area with the most available space. A factor for multiplication is added so that, for every object, N number of clusters is generated around each object. Currently, 4 clusters per object are analyzed, representing each cardinal direction.

After every cluster is generated, the one with the largest area is selected, this is calculated through generation of a point density histogram, defining a minimum pixel density threshold to count for occupied and unoccupied areas of the histogram. The largest cluster obtained then has its centroid calculated and is sent as a 3D pose for object placing.

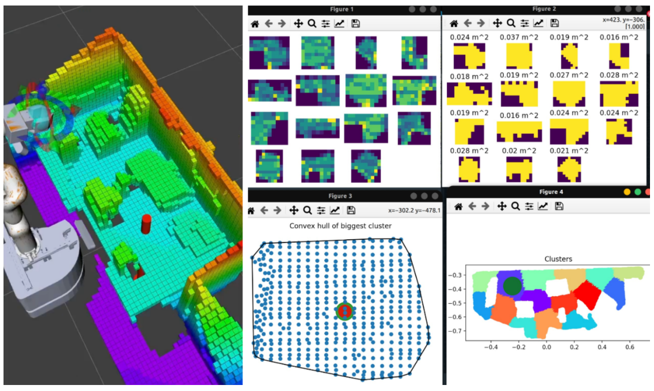

Fig. 2. Place visualization on Rviz and Clusters generation.

4.2 Object detection

Object detection uses a YOLOv8 model trained on custom semi-synthetic datasets. Generation of said datasets involves a three step process which has shown great efficiency both in model accuracy and time spent on creating it. First, photos from the objects the model should be trained to recognize are taken and bounding boxes of their location in the image are obtained, which has been donde through several methods, including external vision models and contour or blob detection.

After the image taken has the object location, a segmentation mask is generated using META's Segment Anything model [Kirillov, 2023] which cuts the object off from its background. In this way, multiple datasets of PNG images of objects without background are obtained. The last step involves pasting the cut images on backgrounds, for which mostly random images from online available datasets have been used.

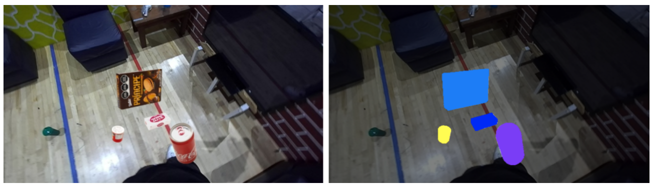

Fig. 3. Objects pasted over a background image, with their segmentation labels.

This dataset generation process has been used both for regular testing models, as well as on several competitions. These include the national 2023 tournament in which a 6 class model of previously unknown objects with around eight thousand images was generated in under 8 hours, including time spent taking pictures of objects and backgrounds. It was also used on a more complex vision task for the 2023 Latin American Robotics Competition, for building a vision model to recognize white cubes with different letters on them. Factors that have been tested and evaluated for the models include the use of random datasets instead of plain colors or only images from the competition arena), on which models trained with random backgrounds have shown an increased efficiency on visually noisy environments. Another factor tested was the use of generated models to redo the first stage of the pipeline, now being able to see the objects and generating more diverse datasets with and without the synthetic generation step, to mostly successful results.

4.3 Human Robot Interaction

Speech Detection The chosen implementation strategy involves dividing tasks into nodes for Voice Activity Detection (VAD) and Speech To Text (STT), fostering decoupling for easy node replacement and versatile code reuse. This modular approach offers a configurable system, providing the flexibility to select nodes based on precision and computational resource considerations. The VAD module, utilizing Silero VAD or webrtcvad, filters input data for the STT node, reducing overall computational resources. OpenAI Whisper, a versatile speech recognition model, is employed for STT, offering various precision and computational resource options. Alternatively, the system can configure Speech Service from Microsoft Azure for resource efficiency. This streamlined approach ensures adaptability and efficiency in speech interpretation while accommodating diverse precision and resource requirements.

Speech Processing The NLP Pipeline in the speech module is the main interpreter of what makes the contents of the given message useful to the main parts of the *Main Engine* of the system, determining useful actions for the robot to actuate. The main challenges for this system are how to get a useful command out of the given messages, and how to relate each part of the command (action, complements) to a given set of possible commands, implementing a consistent and robust solution that works mainly with the extended GPSR command generator.

Filtering and Command extraction The filtering and command extraction section aims to simplify user input messages, making them easily manageable and abstracting individual commands. Leveraging OpenAI Fine Tuning for the *3.5 turbo* model, we utilized the GPSR command generator to gather over 200 inputs, labeling them for training. The first iteration of the fine-tuned model revealed issues of irrelevant text in the command extraction process. To address this, a new dataset, incorporating random coherent text alongside commands, was created. This refined model yielded comparable results while enhancing the capability to filter and extract useful commands from the given text. This ongoing work contributes to optimizing command interpretation in the system, advancing the efficiency of the filtering and extraction processes.

Entities Similarity Once the simplified command was given with the fine tuned model, the Entities Similarity module is the next robust layer of the command for getting the correct command in the way to relate similar words that relate semantically but are not our “key values” for the actions, object, persons or places. (i.e. go is related semantically with move, leave or pass by). Not only this but in cases of generalizing qualities of objects. (i.e. Beverage: Fanta, Soda, Coffee, Drink). For this we built a pipeline for each pair of action and complement. We built a database of actions and complements with the embedded value given by the OpenAI Embeddings model `text-embedding-ada-002`, then for each command input got the right action key, and a list of most similar complements depending on the action. The GPT text model created in the two steps above is further refined using reinforcement learning with human feedback (RLHF) to achieve better results [Hongmei H., 2023].

Interface To enhance direct interaction with the robot and streamline feedback, a Next.js web app using TypeScript and React.js was developed for its accessibility. Roslibjs facilitates the interface's communication with ROS. The interface features a navigation bar for accessing various functionalities. The initial section displays real-time images from the depth camera accessed through a rosbridge. Leveraging the *main engine's* development, the interface showcases control commands such as Go, Grab, Put, Find, Introduce, and Stop, each offering relevant options or locations. Selected options trigger topic publication through roslibjs, transmitting the action and item or location (i.e. Grab Soda). To display navigation status, RVIZ web is employed.

Face detection and recognition Several face detection and recognition models were tested, but face-recognition with DLib was chosen due to its fast processing. Additionally, face encodings can be found once and then re-used in the recognition phase. In general terms, the recognition node loads previous detections, so for each image, the face encodings are found and appended to the array of people saved. For the detection phase, each frame obtained from the camera is processed, finding the encodings and locations of each face. For the recognition, each encoding and location from the frame is compared to all of the encodings from the array to determine if there is a match or not. Faces are tracked according to whether their position is within a threshold of their previous position.

4.4 Navigation and mapping

The navigation and mapping systems of the robot rely on the SLAM (Simultaneous Localization and Mapping) method, which is implemented using the ROS Gmapping package, and for navigation the robot uses the ROS Navigation Stack

Dynamic goals The navigation system employs an action server to streamline the transmission of navigation goals using a JSON file for predefined poses in specified locations (e.g., Kitchen, Room). Each location is defined by coordinates (x, y, z) and orientation quaternion. The action server reads these poses from the JSON file, utilizing them as goals. This approach ensures modularity, reusability, and seamless integration of diverse goals into the robot's autonomous navigation system.

Person following For this task the point calculated using the depth camera is given to the base in a frame known by both, and the base computer then transforms it to a frame usable by the move base action server, and can now send it as a goal. The process employs the Mediapipe pose landmarker model to derive coordinates for person tracking, incorporating depth information for accurate 3D point calculation. Leveraging the Intel RealSense SDK, the point is transmitted, transformed into a pose, and added to a goal for the move base server, ensuring effective coordination and goal transmission between the two nodes.

Variable height obstacle avoidance The costmaps obtained from the LIDAR in the base, are complemented using data obtained with the Depth camera. To avoid passing great amounts of information between the processors, the point-cloud is converted to a laserscan using the ROS `pointcloud to laserscan` package, which produces a laserscan parallel to the ground that considers the obstacles of a wider range of heights.

4.5 Electronics and hardware

The main PCB for the mobile base was created from stracth in order to upgrade the software of this industrial robot. The computer provided by the manufacturer was replaced by an Nvidia Jetson Nano to migrate this system to ROS Noetic. The main board focused on moving the DC motors of the differential base, and a microcontroller was added with RTOS for the actuators and the IMU.

5 Conclusions and future work

We have developed a robust and functional set of general purpose modules, integrated in a layered behavior-based architecture. These features enable RoboTeus to perform dynamic tasks interpreted and solved with its skills. We believe that the future of robotics depends on platforms integrating reactive and deliberative behaviors, this path towards fully intelligent robots adapting to unexpected situations able to assist people in different scopes. In the following year, we will focus our development in implementing new state-of-the-art technologies for each module, enhancing areas such as Manipulation and Navigation with Reinforcement Learning and other AI methods, and investigating for a potential upgrade of our current architecture.

References

1. A. Kirillov. *Segment anything.* arXiv.org, https://arxiv.org/abs/2304.02643, 2023.
2. J. M. Martínez-Otzeta, I. Rodríguez-Moreno, I. Mendialdua, and B. Sierra. *RANSAC for Robotic Applications: a survey.* Number 23(1). Sensors, https://doi.org/10.3390/s23010327, 2022.
3. E Altamirano-Avila, A. Valdivia, and C. Vazquez-Hurtado. *[AVV22] Work in ´ Progress: Implementation of a Digital Twin as Technology to support Discrete Event Control Teaching.* 2022 IEEE World Engineering Education Conference (EDUNINE), DOI: 10.1109/EDUNINE53672.2022. 9782154., 2022.
4. [Avi+22] Altamirano-Avila E. *A Digital Twin implementation for Mobile and collaborative robot scenarios for teaching robotics based on Robot Operating System.* 2022 IEEE Global Engineering Education Conference (EDUCON), DOI: 10 . 1109/EDUCON52537.2022.9766583., 2022.
5. Hongmei He. *RobotGPT: From ChatGPT to Robot Intelligence.* University of Salford, https://doi.org/10.36227/techrxiv.22569247.v1, 2023.

RoboTeus Hardware Description

A modified EAI Dashgo B1 robot was used for mobility purposes and an uFactory xArm 6 was added for manipulation tasks and to mount the depth camera.

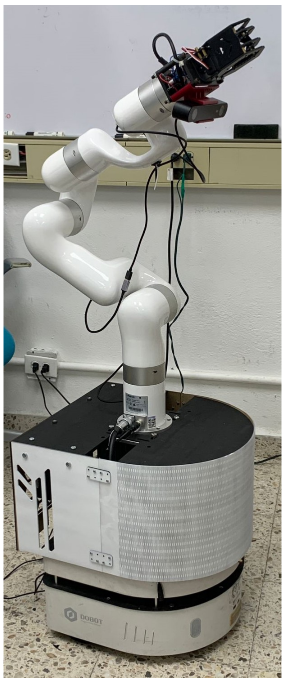

Fig. 4. RoboTeus

– Base: A chassis with a differential configuration of wheels was used as a base, and motors with encoders were used for odometry purposes.
– Computers: Two embedded systems, Jetson Xavier AGX for manipulation and vision tasks and a Jetson Nano connected to it via ethernet through a local network that handles the robot's movement and sensor processing.
– Batteries: 24v batteries for both Xarm 6 and Dashgo (including its Jetson Nano), and 14.8v battery for Jetson Xavier
– Arm: The robot uses an xArm 6
– Arm gripper: Universal custom gripper, interchangeable with variants such as a soft-gripper, a gripper with higher precision for small objects, and a gripper with higher strength using a stepper motor.
– Vision: There's an Stereolabs ZED2 camera attached to the EOF of the Xarm that runs the vision tasks and transmit its data through a USB cable to the Jetson Xavier
– Base cover: Manufactured structure for passive heat dissipation
– Robot dimensions: height: 186 cms (fully extended), width: 42 cms
– Robot weight: 37 kg.

Also our robot incorporates the following devices:

– Power indicator for the Xarm 6 and Jetson Xavier batteries
– 5V switch that enables local data transmission through a ethernet network
– Independent emergency stop devices for Jetson Xavier, Xarm controller and Dashgo
– Customized gripper controller

Robot's Software Description

For our robot we are using the following software:

Robot software and hardware specification sheet

–
- Platform: Robotic Operating System (ROS), Noetic distribution
- Navigation: ROS Navigation Stack
- Face recognition: DLib
- Speech recognition: OpenAI Whisper
- Voice activity detection: Silero VAD
- Speech processing: OpenAI GPT and Embeddings
- Object recognition: YoloV8
- Manipulation planning: MoveIt
- Point cloud proccessing: PCL and RANSAC
- Integration: Multimaster package

Robot performing miscellaneous tasks

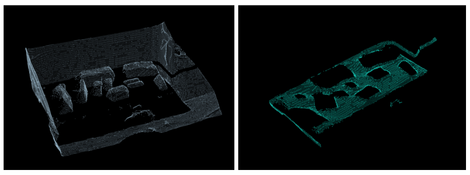

Fig. 5. Table extraction from the full generated point cloud for manipulation.

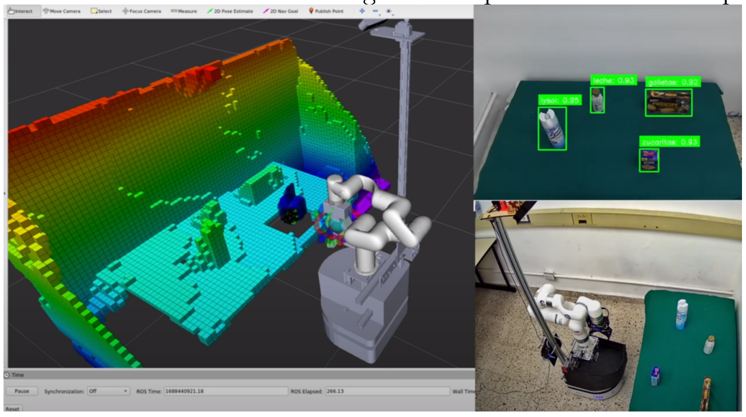

Fig. 6. Generated octomap, object mesh and GPD detected poses, along real-time robot view with 2D object detection and an external view of the scene for the pick action.

Robot software and hardware specification sheet

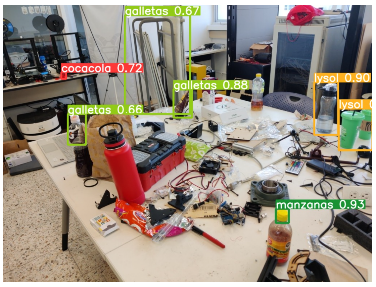

Fig. 7. Trained model results in our laboratory.

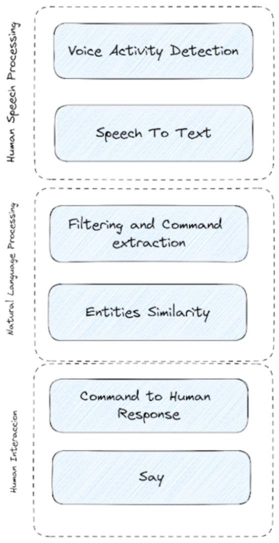

Fig. 8. Human Robot Interaction pipeline overview

Robot software and hardware specification sheet