PQMEC@HOME Team Description Paper

Alexandre C. F. Filho, Eduardo A. S. Garcia, Gabriel M. Marques, Guilherme S. O. de Brito, Gustavo R. da Silva, Heinz F. C. Rahmig, José R. R. Teles, Lucas B. Rodrigues, Lucas R. S. Gris, Luis F. F. de O. Freitas, Márcio G. B. Junior, Myrella A. Bordado, Rafaela M. Silva, Rodrigo M. de Carvalho, Vinícius A. R. de Souza, Victor G. Pimenta, Vinicius A. R. de Souza, Werisson E. S. Pereira, Yuri G. Lima

June 30, 2022

www.overleaf.com/project/62bf067a16b4e458a15aa481

Abstract

This paper describes the service robot Zordon of team Pequi Mecânico that will participate on the Robocup@Home competition which takes place annually in Brazil. This competition has influenced the development of research in natural language processing, computer vision, robotic manipulation, simultaneous localization and mapping.

1 Introduction

Pequi Mecânico robotic team exists since 2011 and took part in Latin American and Brazilian Robotics Competition in various categories: IEEE Standard Educational Kit (SEK), IEEE Open, RoboCup Small Size Soccer (F180), IEEE Humanoid Robot Racing (HRR), IEEE Very Small Size Soccer (VSSS) and RoboCup Soccer Simulation 2D. In 2019 we decided to compete for the first time in the Robocup@Home league.

Service robots are hardware and software systems that assist humans to perform daily tasks in complex environments. In order to achieve this, they have to be able to understand spoken or gesture commands from humans; to avoid static and dynamic obstacles while navigating in known and unknown environments; to recognize and manipulate objects and performing several other tasks that a person might request. Our robot's name is Zordon (see Figure 1).

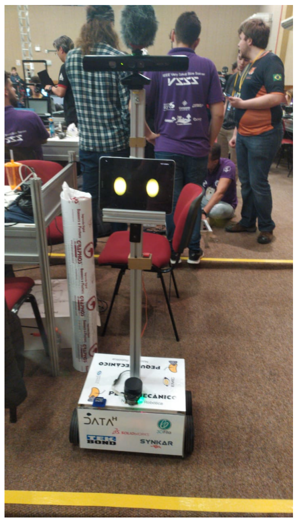

Figure 1: Zordon in the LARC 2019 in Rio Grade/RS - Brazil

2 Zordon's Robotic Architecture

2.1 Manipulator

The robotic manipulator it's created with eight XYZRobot A1-16 Servo motors with the arm structure design by the team, to control the arm we use the software Moveit! In combination with an octomap calculated from the RGB-D camera for motion planning and obstacle avoidance.

2.2 Human Interaction

The main way to interact with the robot is through voice commands, to do the speech recognition we use the *Google Cloud Speech-to-Text API*[3] that is able to provide state-of-the-art performance and is flexible enough to support additional contextual information for processing of the audio to make it more precise on the context of a personal domestic robot.

Aside from the voice command is possible to input commands through it's Graphical User Interface (GUI) on his main screen in a simple text format. The GUI is in a web-based format that allow user's to access it remotely on any mobile device, it provides control and feedback over the actions of the robot through text and audio, the interface also show a friendly animated face that reacts to the user input.

3 Current Research

3.1 Natural Language Understanding

For the robot perform an action, it needs first to comprehend the command received by the speech recognition or by the Graphical User Interface (GUI), to achieved that we choose a deep learning approach using an open-source natural language processing tool called *Rasa NLU*[12], it can be configured to use a recurrent neural network inspired by *Facebook's StarSpace paper*[20] to do intent classification and entity extraction from texts.

This network as trained on a text command generator build by the team that is based on the tasks that the robot is able to perform. To improve the precision of this network is necessary to acquired a list of commands given by a human user's instead of computer generated ones, thus we are storing all commands received by the robot and this will improve his capability of understanding natural language over time.

3.2 Computer Vision

3.3 Human detection

At the moment our main vision sensor is the Intel realsense D435i camera. Although there is the OpenNI API for gestures recognition and human body movements for a sensor, we decide to use the OpenPose [2] algorithm for these tasks.

The OpenPose it's a Deep Learning based human pose estimation, providing real-time multi-person detection. The model takes as input a color image and produces, as output, the 2D locations of keypoints for each person in the image. The authors also provide two pre-trained models, one trained on the Multi-Person Dataset (MPII) [1] and the other trained on the COCO dataset [7], producing 18 and 15 points, respectively.

In order to get a better frame rate in the Jetson Xavier GPU, we have adopted the Tensorflow implementation provided by [6]. The solution provides several variants that have some changes to the network structure for real-time processing on the CPU or lower-power embedded devices. We achieve good accuracy and real-time performance, approximately 8 FPS.

3.4 Human Tracking

With the OpenPose algorithm, we can detect multiple persons in the image, however, it does not provide means for people identification. In order to track the operator in the follow-me task, the time diced to use feature descriptor for person classification.

The OpenPose provides a set of body keypoints which can be used to create a local representation of the texture. This local representation is constructed using Local Binary Patterns (LBPs) [10]. With a neighborhood of size r surrounding the keypoints we compute the LBPs. Then we compute a histogram that tabulates the number of times each LBP pattern occurs. So, we treat this histogram as our feature vector. To handle multiresolution grayscale and rotation invariance, we use the extension to the original LBP proposed by [11].

In the follow-me task when the person stands in front of the robot we compute the LBPs histogram of the operator and register the vector in the memory. During the process we use a similarity metric to find the best match.

3.5 Navigation

In order to navigate through robocup@home environment, Zordon uses ROS *navigation stack* [17]. Thus, the robot can move from wherever it is to a desired point avoiding obstacles in its map using three layers of costmap2d ros package[15]:

- obstacle_layer: uses data from LIDAR as laser_scan and point cloud provided from kinect's sensors as sources of observation;
- static_layer: map acquired from mapping stage (explained in next subsection);
- inflation_layer: propagating cost values out from occupied cells that decrease with distance as stated by environment.

To perform the best movement according to Zordon's structure constraints (e.g. differential driver, acceleration limits, minimum velocity), we adopted a Timed-Elastic-Band local planner, as described in [13]. An implementation of this approach in Robot Operation System is available in [18]. We are using it to achieve optimization of global planner at during runtime and minimizing the trajectory execution time.

3.5.1 Mapping

Previous to the task's execution moment, the robot Zordon will navigate the entire environment for the purpose of tracking objects and mapping the house. This way, using a Lidar as laser scanner and dead-reckoning sensors as odometry and IMU combined by Extended Kalman Filter, we can create a 2D map of occupancy performed by a *Simultaneous Localization and Mapping* approach.

By this day, we are searching for the best SLAM algorithm which attempts hardware limitations and sensors uncertainty. Gmapping [4] is one provided as a CreativeCommom licensed solution of concurrent localization and mapping problem. It is accessible as a ROS package in [16]. Other solution to SLAM problem that has been appreciated by our team is vinySLAM [5]. Even though in tests executed recently both SLAM solutions have achieved good performance, with vinySLAM having accomplished better CPU usage, we did not reach all results to claim which is the best one. Thereby, gmapping has been adopted as standard solution to the problem of mapping.

3.5.2 Localization

An implementation of Extended Kalman Filter (EKF) available in ROS is [8]. By using that, we can combine odometry of wheels, visual odometry and IMU data to provide more accurate relative position measurements. Collecting absolute position measurements like laser_scan data from Lidar, we merge information and pass it through a Localization based on map approach.

By now, ORB-SLAM is a good solution to acquire visual odometry data, and we are using a self-adapting ROS package inherited from RGB-D application [9]. Our solution changes some stacks of algorithm to obtain odometry messages and parameterize the code for realsense sensor.

There are a lot of solutions to the problem of localization with a map (e.g. EKF-Localization, Multi-Hypothesis Tracking, Grid Localization, Monte Carlo Localization explored by [19]). Nevertheless, each of them has great complexity measures in practice. Therefore, an Adaptive Monte Carlo Localization (AMCL) has been used as state-of-art localization problem. We are working with an AMCL available in [14] and parameterizing the code to hitch our sensors.

3.6 3D Modeling and Simulation

After the development of the first 3d modeling software, the sketchpad, in 1963, the creation of solid objects through the representation of a volumetric object became possible and increasingly used - in fields ranging from cinema to robotics.

The modeling in its most common aspect is performed by creating a mesh of segments that will give shape to the object, this is developed by several techniques, the most common being the polygon technique, the vertex technique and the edge technique.

In Robotics, the robot geometry description is based on the Unified Robot Description Format (URDF), which is a package with XML format to represent the robot model.

4 Conclusion and future work

This is Pequi Mecânico's second attempt to compete in this category. Our robot was designed based on other teams' projects, information available at ROBOCUP @home wiki, and our own insights on the competition's challenges. The Deep Learning approaches we use will allow the robot to learn throughout the course while it receives commands and perform tasks. We already have several versions of Zordon which perform different sets of tasks, and, until the competition, we will have many more.

5 Team Information

5.1 Robot Software Information

Operating System	Ubuntu 18.04
Middleware	ROS Melodic
Navigation	ROS *navigation stack*
Localization	*AMCL*
Mapping	*Gmapping*
Object Recognition	YOLOv4 tiny
Face Detection	YOLOv4 tiny
Human Detection	OpenPose
Gesture Recognition	OpenPose
Face Recognition	Eigenfaces
Speech Synthesis	Google Cloud Text-to-Speech API
Speech Recognition	Google Cloud Speech-to-Text API
Natural Language Understanding	RasaNLU

5.2 Robot Hardware Information

Base Motors	2x Hoverboard Wheel Motor 6,5"
Manipulators	XYZrobot 6 DOF Robotic Arm
Microphone	Rode Videomic Go
Display	Samsung Galaxy Tab S3
RGB-D Camera	Intel RealSense D435i
RGB Camera	Logitech Pro C920
Speaker	JBL Charge 3
IMU	Intel RealSense D435i
LIDAR	RPLidar A2
Embedded System	NVIDIA Jetson Xavier

References

[1] Mykhaylo Andriluka et al. "2d human pose estimation: New benchmark and state of the art analysis". In: *Proceedings of the IEEE Confer-*

ence on computer Vision and Pattern Recognition. 2014, pp. 3686–3693.

[2] Zhe Cao et al. “OpenPose: realtime multi-person 2D pose estimation using Part Affinity Fields”. In: *arXiv preprint arXiv:1812.08008*. 2018.

[3] *Cloud Speech-to-Text - Speech Recognition — Cloud Speech-to-Text — Google Cloud*. URL: `https://cloud.google.com/speech-to-text`.

[4] Giorgio Grisetti, Cyrill Stachniss, Wolfram Burgard, et al. “Improved techniques for grid mapping with rao-blackwellized particle filters”. In: *IEEE transactions on Robotics* 23.1 (2007), p. 34.

[5] Arthur Huletski, Dmitriy Kartashov, and Kirill Krinkin. “Vinyslam: an indoor slam method for low-cost platforms based on the transferable belief model”. In: *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE. 2017, pp. 6770–6776.

[6] Ildoo Kim. *Deep Pose Estimation implemented using Tensorflow with Custom Architectures for fast inference*. `https://github.com/ildoonet/tf-pose-estimation`.

[7] Tsung-Yi Lin et al. “Microsoft coco: Common objects in context”. In: *European conference on computer vision*. Springer. 2014, pp. 740–755.

[8] Thomas Moore and Daniel Stouch. “A generalized extended kalman filter implementation for the robot operating system”. In: *Intelligent autonomous systems 13*. Springer, 2016, pp. 335–348.

[9] Raúl Mur-Artal and Juan D. Tardós. “ORB-SLAM2: an Open-Source SLAM System for Monocular, Stereo and RGB-D Cameras”. In: *IEEE Transactions on Robotics* 33.5 (2017), pp. 1255–1262. DOI: `10.1109/TRO.2017.2705103`.

[10] Timo Ojala, Matti Pietikainen, and David Harwood. “Performance evaluation of texture measures with classification based on Kullback discrimination of distributions”. In: *Proceedings of 12th International Conference on Pattern Recognition*. Vol. 1. IEEE. 1994, pp. 582–585.

[11] Timo Ojala, Matti Pietikäinen, and Topi Mäenpää. “Multiresolution gray-scale and rotation invariant texture classification with local binary patterns”. In: *IEEE Transactions on Pattern Analysis & Machine Intelligence* 7 (2002), pp. 971–987.

[12] *Open source conversational AI*. URL: `https://rasa.com/`.

[13] Christoph Rösmann et al. “Trajectory modification considering dynamic constraints of autonomous robots”. In: *ROBOTIK 2012; 7th German Conference on Robotics*. VDE. 2012, pp. 1–6.

[14] *ROSWiki AMCL*. `http://wiki.ros.org/amcl`. Accessed: 2019-06-17.

[15] *ROSWiki costmap_2d*. `http://wiki.ros.org/costmap_2d`. Accessed: 2019-06-17.

[16] *ROSWiki gmapping*. `http://wiki.ros.org/gmapping`. Accessed: 2019-06-17.

[17] *ROSWiki navigation stack*. `http://wiki.ros.org/navigation`. Accessed: 2019-06-17.

[18] *ROSWiki teb_local_planner*. `http://wiki.ros.org/teb_local_planner`. Accessed: 2019-06-17.

[19] Sebastian Thrun, Wolfram Burgard, and Dieter Fox. *Probabilistic robotics*. MIT press, 2005.

[20] Ledell Wu et al. “StarSpace: Embed All The Things!” In: *CoRR* abs/1709.03856 (2017). arXiv: `1709.03856`. URL: `http://arxiv.org/abs/1709.03856`.