Pumas@Home 2017 Team Description Paper

Jesus Savage, Marco Negrete, Hector Arce, Reynaldo Martell, Hugo Estrada, Julio Cruz, Manuel Pano, Jaime Marquez, Edgar Silva, Edgar Vazquez, and Jose Luis Cruz

National Autonomous University of Mexico, México DF 04510, MEX,
http://biorobotics.fi-p.unam.mx

Abstract. This robot is based on the ViRbot architecture, implemented by several modules that perform different tasks, using the ROS platform, together with a cross-platform system called Blackboard developed by us. This year, several improvements in hardware and software of Robot Justina has been made. Currently, our research is mainly focused on task planning using different approaches and the development of a framework to test and debug the execution of the plans.

1 Introduction

The service robots are hardware and software systems that can assist humans to perform daily tasks in complex environments. To achieve this, a service robot has to be capable of understanding commands from humans, avoiding static and dynamic obstacles while navigating in known and unknown environments, recognising and manipulating objects and performing several other tasks that a person might request. This paper describes the autonomous service robot Justina and the research of the Team Pumas, which has been participating in RoboCup@Home league since 2007. It is organised as follows:

In section 2 the ViRbot architecture is described as the platform for the robot software system. Section 3 enumerates the hardware and software components of robot Justina, and the functionality of each of them is described. Section 4 is an abstract of the latest research developed in our laboratory in order to improve the robot's performance and finally, in section 5, conclusions and future work are given.

2 Background: Platform and Architecture

This robot is based on the ViRbot architecture for autonomous mobile robot operation [1], which provides a platform for the design and development of software for general purpose service robots.

The ViRbot architecture defines a human-robot interaction interface made of three main layers (see figure 1):

- **Input layer:** Includes all algorithms related to the acquisition of data from the environment.

– **Planning and Knowledge:** This layer performs most of the AI algorithms.
– **Execution:** Includes low-level control and supervision.

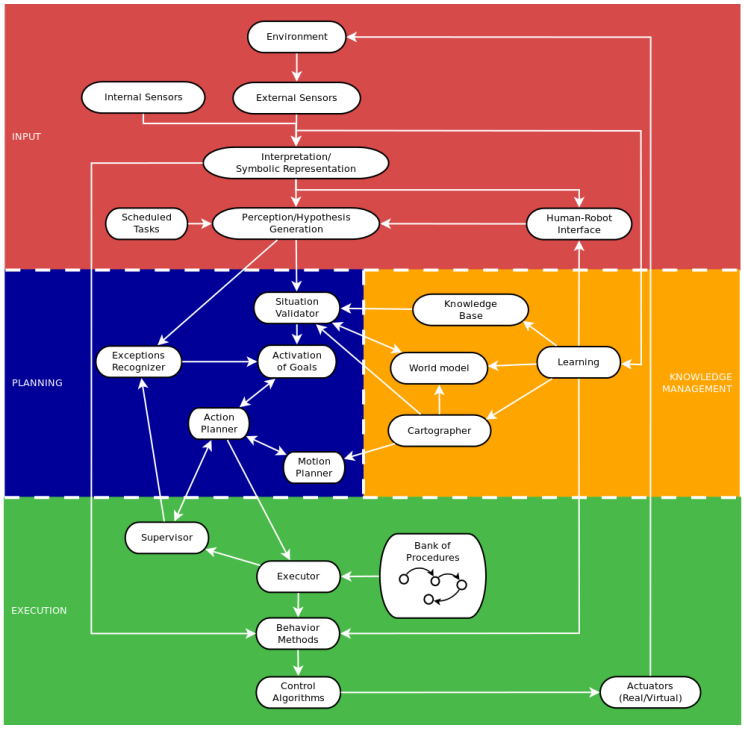


Fig. 1: Block diagram of the ViRBot architecture.

The ViRbot architecture is implemented in our robots through several modules that perform well defined tasks, with a high level of interaction between them. The principal framework used for interaction is ROS, where a Module is represented by one or several ROS's nodes.

Also, we make extensive use of our self-made Framework for modules interaction called Blackboard (BB). In BB, the information exchange is made through a central module which supports shared variables, publisher/subscriber pattern, and message passing. BB was developed mainly in C# and the .NET framework, with APIs to build BB modules for C#, C++ and Python 2.7. It runs on Windows and Unix-based systems (with the help of Mono, an open source implementation of Microsoft's .NET framework). Currently, we had tested BB modules running on Windows, Ubuntu, MacOS, Android and Rasbian

BB's commands and shared variables have a certain equivalence to ROS's services and topics respectively. Integration between ROS nodes and BB modules

has been accomplished through the implementation of the BlackBoard Bridge module, which is a ROS node written in C++ with the BB Module API.

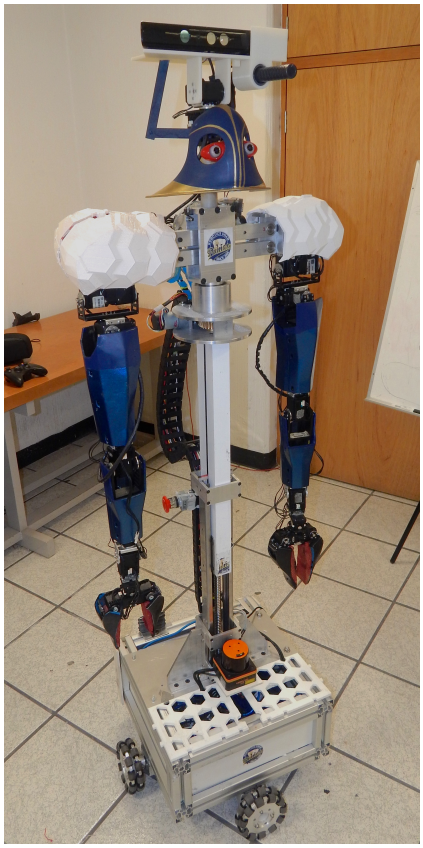


Fig. 2: Robot Justina

3 Robot Justina

3.1 Actuators

- **Mobile base:** Omnidirectional through differential pair configuration and omnidirectional wheels (self design).
- **Manipulators:** 2 x 7-DOF anthropomorphic arms with 10 Dynamixel servomotors each.
- **Head:** 2-DOF (Pan and tilt) built with Dynamixel servomotors.
- **Torso:** 3-DOF (Elevation, pan, and shoulders tilt) through a worm screw and a configuration of gears.
- **Remote:** Xbox wireless remote controller.

3.2 Sensors

- **RGB-D Camera:** Microsoft's Kinect sensor
- **RGB Camera:** Logitech Pro C920 Full HD.
- **Microphone:** Rode NTG2 directional microphone.
- **Laser:** Hokuyo rangefinder URG-04LX-UG0.

3.3 Software

Robot Justina uses two computers: one running Ubuntu 14.04 and one running Windows 7. Modules are programmed in C#, C++, Python or CLIPS [2]. Modules currently running on Justina are:

Action Planner Details are given in subsection 4.3.
Simple Task Planner Bank of procedures that involves simple and repetitive tasks easily achieved by state machines, e.g. grasping an object, searching for a face, aligning with an edge, etc.
Motion Planner Path planning for the non-linear control of the mobile base. Also, some obstacle avoidance behaviours are implemented using point cloud data of the scene.
Object Finder Details are given in subsection 4.4
Person Finder Multiple face detection using a Haar cascades like algorithm. Face recognition and the online training is performed using 2D and 3D data from the Kinect sensor.
Speech Recognition Throws hypothesis (text strings) of recognized voice with a confidence ranking using Microsoft SAPI 5.3 [3] (commercial software).
Speech Generator Voice synthesizer using Microsoft SAPI 5.3 [3] with the Loquendo Susan voice (commercial software).
Cartographer Map building and localisation based on GMapping node.
Leg Finder Legs of pedestrians in a scene are detected and localised using the laser readings.

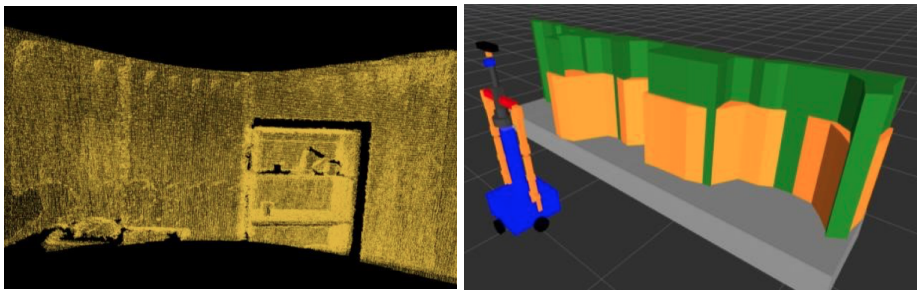
4 Current research

4.1 Surface reconstruction on virtual environments

This research has as main purpose to build a 3D representation of an environment, in which a service robot can interact, using the data obtained by a wide variety of sensors.

Particularly, the data that we use in our experiments are point clouds sets. The capture of this data is performed by the Kinect sensor of robot Justina. To align several captures of point clouds of a scene, an implementation of the ICP (Iterative Closest Point) algorithm is used. Also, a sampling step is implemented using Vector Quantization [4]. The virtual environment builded can ve visualised using the simulator described in subsection 4.2 .

The system was successfully tested experimentally by reconstructing our laboratory. Figure 3 shows an image of multiple point clouds sets captured by the robot and the virtual environment created after the method described above.



(a) Several point clouds captures of a scene.

(b) Virtual environment build after processing the sets.

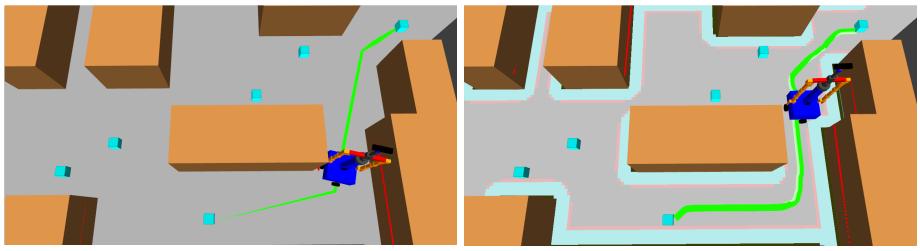
Fig. 3: Example of the test performed in a real scene.

4.2 Service Mobile Robot Simulator

We developed a new software for simulation and graphical representation of algorithms and behaviours used in autonomous service robot navigation in dynamic environments. This simulator allow us and new robotics students to test and improve behaviours for the navigation module of robot Justina.

This new simulator has implemented several features such as laser range finder simulations, obstacle representation and rendering using point clouds sets, collision detection, kinematics models for certain mobile base configurations, etc, polygons growing. Also, a graphical user interface has been created and is fully integrated within the ROS framework

Using this simulator, we have successfully tested well-known navigation algorithms such as artificial potential fields, occupancy grid exploration, topologically map routing using graph search methods, and some hybrid behaviours that combines reactive models and hierarchical models.



(a) Path planning using an algorithm for a Topological Map.

(b) Path planning using an algorithm for an Occupancy Grid.

Fig. 4: Test of a navigation algorithm using the graphical interface of the simulator.

Figure 4a shows the robot simulated, the virtual representation of our laboratory and a path created using Dijkstra Algorithm based on a topological representation of the environment. In contrast, Figure 4b shows a path created using the same robot and virtual environment, but using an Occupancy Grid instead for representing the map.

Thanks to the integration with ROS, after a simulation has been run, it allow us to test and visualise, in real time, the algorithms in robot Justina for result analysis and comparison. Different algorithms in a wide-range of environment configurations has been modelled and tested, both in simulation and in the real robot.

4.3 High level task planning using conceptual dependency

This year, a new knowledge based system for high level task planning has been developed. This system is designed for interpretate a command in a natural language format and for generate a sequence of high level actions that have to be completed to accomplish the command. With this system, the robot generates behaviours that can drive his own world representation, similarly to a human being.

The system is composed by two principal modules:

Communication Module In this module, the grammar structure of the commands is defined. Performs a syntactical analysis and a semantic interpretation using inferences and Conceptual Dependency, in order to find a meaning for the command. Finally, this module returns a formal expression that describe a sequence of high level actions that the robot have to execute.

Execution Module Consist in a set of rules for infer and determinate the sequence of high level tasks as basic and atomic predefined actions that represents the plan that will allow the robot to accomplish the command that has been requested. This module was developed using CLIPS, an expert system designed to represent and model the human knowledge.

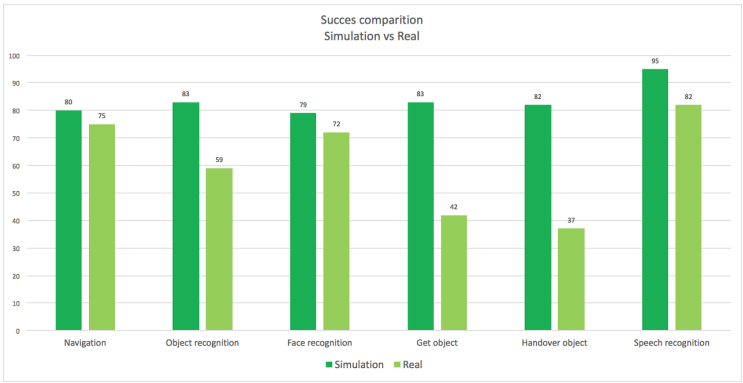


Fig. 5: Comparison of success between simulated plans and real plans.

We have compared the number of successfully executed plans, both in simulation and with the real robot, as can be seen in Figure 5, for 100 executions for each action.

So far, we have modelled actions like Navigation, Object Recognition, Face Recognition, Object Taking and Object Leaving.

4.4 Low or null texture objects recognition using RGB-D cameras

Currently, several robust technics based on feature extraction and description exist for object recognition. However, if the objects are low textured, only a few number of features can be extracted, making the matching process unreliable. For these cases, we developed a method that combine three characteristics: color, size and shape, combining the color and depth information for the recognition process, after a 3D detection and segmentation in a plane for each object.

Color Information is extracted from the HSV space of the object's pixels and it is represented by the histogram of the Hue component, but only for pixels with Saturation and Value above certain threshold. For pixels below these thresholds, two more bins are added to the histogram. For campaign histograms we used the histogram Intersection [5].

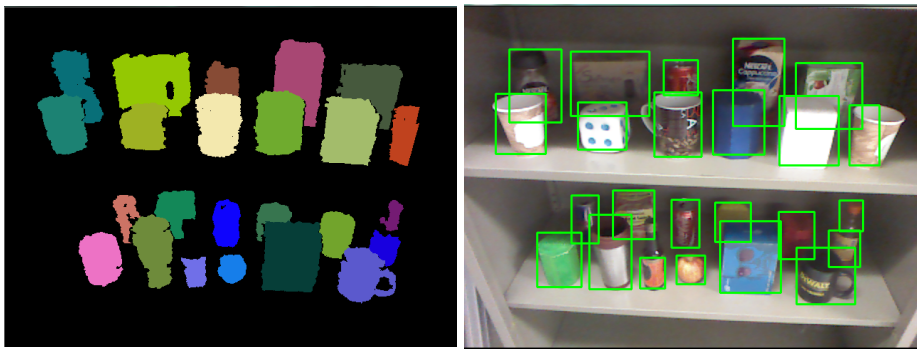
The size and shape is estimated from the object's point cloud, which are obtained using an oriented bounding box (OBB) of the point cloud as follow: the base of the OBB is obtained from the oriented bounding rectangle of the projection of the points in the plane below them. The heights are obtained from the maximum distance of the points to the plane. The shape is characterized using the Hu Moments [6] of the convex hull calculated from the points projected over the plane below them.

For the recognition process we compare in three steps: size, shape and color characteristics, removing candidates below a certain threshold for each step. At the end, from the remaining candidates, we select the best one according to a color-based similarity function.

These method has been tested experimentally in the Rocking robotics competition, where the we obtained a second place in the "object perception test", showing fast and robust results for changes in light , scale, and rotation in a plane parallel to the plane below the object. Figure 6 shows an example of object recognition on a shelf (multiple planes).

5 Conclusions and future work

The ViRbot system and robot Justina has been successfully tested in Robocup in the category @Home since the Robocup competition of Atlanta 2007. The team has get into the finals the last two years (2014 and 2015) in this competition. Also, in 2014, the team participated in the RoCKIn Competition wining a 1st place and a 2nd in different categories. In the 2015 edition of the same competition, the team won the second place in the "Object perception" test. Last year, in Robocup 2017, the team got into the 2nd stage.



(a) Point clouds corresponding to each segmented object.

(b) Objects can be segmented even with occlusions.

Fig. 6: Example of object segmentation on several planes.

In these years, the full system has been improved, both in hardware and software, having reliable performance and showing promising results. Particularly, this year, we have a new omnidirectional mobile base for navigation and a new torso. In terms of software, we have change the way of conceiving the tests of the competition: from state machines to inferences based on rules.

As future work, we are trying to improve the computer vision algorithms to achieve real time detection and recognition of objects and persons. Also, we want to explore some methods for better path planning for object manipulation.

References

1. Jesus Savage and et al. Virbot: A system for the operation of mobile robots. In Ubbo Visser, Fernando Ribeiro, Takeshi Ohashi, and Frank Dellaert, editors, *RoboCup 2007: Robot Soccer World Cup XI*, volume 5001 of *Lecture Notes in Computer Science*, pages 512–519. Springer Berlin Heidelberg, 2008.
2. Gary Riley. *CLIPS Reference Manual Version 6.0. Technical Report Number JSC-25012.* Software Technology Branch, Lyndon B. Johnson Space Center, Houston, TX, 1994.
3. Microsoft. Speech server - microsoft corporation. `http://www.microsoft.com/speech/speech2007/default.mspx`, 01 2011.
4. Yoseph Linde, Andres Buzo, and Robert Gray. An algorithm for vector quantizer design. *IEEE Transactions on communications*, 28(1):84–95, 1980.
5. Michael J Swain and Dana H Ballard. Color indexing. *International journal of computer vision*, 7(1):11–32, 1991.
6. Ming-Kuei Hu. Visual pattern recognition by moment invariants. *Information Theory, IRE Transactions on*, 8(2):179–187, 1962.