# BahiaRT@Home 2015 Team Description Paper

Josemar Rodrigues de Souza and Deivite Guimarães, Fagner Pimentel, Flávio Sapucaia, Camila Laranjeira, Leone Jesus, Marco A. C. Simões, Diego Frias

> [State University of Bahia (UNEB/ACSO), Salvador, BA, Brazil], http://www.acso.uneb.br/bill/

Abstract. The center for Computer Architecture and Operating Systems (ACSO) at the State University of Bahia (UNEB), represented by Bahia Robotics Team (BahiaRT), introduces BILL (the Bot. Intelligent, Large capacity and Low cost). Aiming to develop solutions for service and assistive robotics, RoboCup@Home works as the perfect test environment to validate such solutions. This paper describes the development of BILL's basic features, such as object recognition, voice recognition and synthesis, mapping, navigation, people tracking, object manipulation and face recognition. The main goal behind this research is to develop efficient algorithms to perform all functionalities proposed. This paper presents satisfactory results using simple and robust methods.

# 1 Introduction

In 2008 ACSO has begun to do research related to service robots, which led up to the robot named BILL (the Bot. Intelligent, Large capacity and Low cost). BILL was built to meet the requirements of the RoboCup@Home league as part of the RoboCup initiative. Figure 1 shows BILL's current look and his conceptual art, which we plan to finish in time for the 2015 edition of RoboCup@Home.

In the RoboCup@Home league, a set of benchmarks is used to evaluate the performance of the robots in a non-standardized and realistic home environment. Thus, the tasks are focused on the areas of interaction and cooperation between Human-Robot, navigation and mapping in dynamic environments, recognition of objects and faces in natural lighting conditions, artificial intelligence, standardization and system integration, etc.

This is BILL's first attempt to attend RoboCup@Home. Therefore, the objective of this TDP is to show the work developed by ACSO in preparation for BILL's first participation and its main functionalities developed until today.

Section 2 will present a historical background of ACSO and its participation in Robocup since its begining. Then section 3 will describe the main functionalities currently developed for BILL. At section 4 some experiments and results will be presented. Section 5 presents our conclusions and future work. And finally, at the end of this TDP there is a brief description of each hardware and software component used in BILL's development.



Fig. 1. BILL's current look on the left and his conceptual art on the right.

# 2 Group Background

BahiaRT is a scientific cooperation group which aims to add the state of Bahia in scientific research in Robotics and Artificial Intelligence. The initiative was created by ACSO in August 2006 by researchers and students of UNEB and other institutions.

The main objective of BahiaRT is to actively participate in the international research initiative known as RoboCup. The goal is to strengthen the Brazilian participation in this important initiative in Robotics and Artificial Intelligence.

At this time, BahiaRT competes in two RoboCup Leagues: 3D Soccer Simulation and RoboCup@Home. BahiaRT has attended in RoboCup since 2007. In the first year, we have participated in 2D Soccer Simulation League and also in the Mixed Reality demonstration competition (formerly Physical Visualization). In Mixed Reality, BahiaRT won the third place in RoboCup 2009 and the fourth place in RoboCup 2010. BahiaRT also has developed the MR-SoccerServer: the main module of MR software infrastructure. In 3D Soccer Simulation, BahiaRT won the fifth place in RoboCup 2014.

# 3 Functionalities

This section will present the basic functionalities implemented in BILL and its development. The functionalities are: (1) Object Recognition; (2) Face Recognition; (3) Navigation; (4) People Detection and Tracking; (5) Speech Recognition and Synthesizing; and (6) Object Manipulation

#### 3.1 Object Recognition

This subsection presents the pipeline for object recognition used in BILL and the training of the object classifier based on color and shape features.

The pipeline of objects recognition is divided into four steps: (1) Acquisition of RGB-D sensor information; (2) Segmentation of the table in front the robot; (3) Segmentation of the objects on the table; and (4) Classification of each segmented object. In the first step, the color and depth informations are extracted from the RGB-D sensor through the ROS [1] publiser/subscriber system and stored in a point cloud structure using the Point Cloud Library (PCL) [2].

The second step is performed to segment the table in front the robot. First, the distant points from the point cloud are removed (over 1 meter away). This is followed by a search in the point cloud for planar structures using the Random Sample Consensus (RANSAC) method [3], native on PCL. The major planar structure found is then segmented and classified as the table. Finally all the points that compose this structure are removed from the point cloud. The remaining points are classified as potential components of target objects.

In the third step, the points in the point cloud are converted to an OpenCV [4] image structure, where the removed points are transformed into zero pixels and the rest are converted into pixels with their original color information. In this image, is extracted a binary mask representing the non-zero pixels and then a closing morphological operation is applied aiming to smooth the shape of the extracted objects. In the resulting binary image is performed a search for contours using the algorithm presented in [5], native on OpenCV. Each contour found is stored in a vector.

In the fourth step, the objects are finally classificated. This is done by first extracting the characteristics of each object. Two types of features are used: (1) Color features, represented by the mean of histograms in each channel of LAB color space present in the object and (2) shape features, represented by the Hu's invariant moments [6] that use invariant values to translation, rotation and scale for a given shape. In all, for each object is extracted a vector with 10 features (three for color and seven for shape). From this feature vector, the object is finally classified using a SVM classifier previously trained.

The previous training aims to create a SVM model classifier to be used in the objects recognition, and it is divided into four parts: (1) Creation of the dataset, which is done by extracting images containing the target objects. The Images should contain several objects varying distance, position and angle. (2) Segmentation of objects, made in the same way as shown previously in the pipeline of objects classification. (3) Extraction of features also performed in the same way as shown earlier in the pipeline for the object classification. (4) Manual Classification of segmented objects, where the classes of objects are identified manually by an user. (5) Training, testing and creation of the classifier model using the libSVM library tools. Finally, with the trained classifier, a classification model is generated and it is used by the robot to automatically classify new instances of objects.

#### 3.2 Face Recognition

The module responsible for face recognition was elaborated to enable manmachine interaction, making it possible to perform tasks such as finding an user on a crowd or recognizing their gender. To allow the robot to recognize faces, it's necessary to follow a set of steps. Using a basic form, the first step is to find the face in the image and crop it. This stage is done by the *Haar Cascade* method, which is based in an algorithm of cascade classification [7].

The algorithm Haar Cascade Classifiers runs an image from the pixel (0,0) with sub-windows, passing through each image region using a classification. Thereby, it follows line by line using the X and Y axis, repeating itself everytime a search is completed and restarting with higher sub-window scales until the sub-window meets the size of the entire image. All sub-windows pass through several classificators, only the ones that keep a positive result after each and every analysis, are considered faces. If in any step of the detection until the end of the cascade, one window is rejected, the classificator understands there are no faces. When this algorithm finds a face, it will separate the face from the rest of the image, which will then be normalized to match the characteristics of the images on the face dataset, i.e., it will be resized and discolored. Moreover, the face is submitted to an image processing method, combining the Histogram Equalization and the Gaussian Filter[8] to positively influence the results of the recognition stage.

The next step is a comparison to the existing faces in the picture bank to find a match, highlighting facial attributes that can be used to differentiate between the faces of the bank. In this project, these attributes are not analyzed separately, it uses the facial recognition algorithm Fisherfaces [9], which takes into account the full facial representation to obtain the necessary information, both in the analysis of the captured image, and in the training of pictures in the dataset. Fisherfaces creates a low-level representation of the face, resulting in a linear discriminant analysis on the projection of the image, in the sub-space of images created with the principal component analysis (PCA) [10]. As a result of this comparison, the method returns a number that represents the Euclidean distance between the low-level representation of the captured image, the most similar in the dataset, and a number that represents this associated image.

### 3.3 Navigation System

In order to ensure greater autonomy in performing tasks in unknown and unstructured environments, a robust navigation system is invaluable. It should be able to perform the navigation process efficiently and safely, while new environment data is collected and analyzed. For this purpose, Simultaneous Localization And Mapping (SLAM) approach is used to map the environment and provide self-localization in this map.

There are several steps to compose the process of navigation to explore an unknown environment. First the map is built using the incremental mapping package, Hector Mapping [11], a system for fast online learning of occupancy grid maps. The grid map generated is done by RPLIDAR 360 Laser Scanner sensor, which is able to get the 2D world data. After generating a two-dimensional world representation, the next step is to create the path planning based on the occupancy grid map. In order to provide a path clear of obstacles, the occupancy grid is updated based in Dynamic Voronoi ROS package approach [12], useful to compute the corresponding Euclidean distance maps (DM) and Euclidean Voronoi diagrams (GVD) with points that mark newly occupied or freed cells, so the grid map can be updated to reflect the changes in the environment. Then the shortest path to the goal is computed by means of D\* Lite algorithm [13] to ensure obstacle avoidance over incremental mapping.

At last, the motion planning in charge of getting the path planning and relating linear and angular motion is triggered, which applies the kinematics control law, and sends a message to low-level control.

### 3.4 People Detection and Tracking

During Human Robot Interaction (HRI) people detection and tracking has a crucial role for service robots like BILL.

The kinect OpenNI library [14] provides position identification of key points on the human body, such as head, torso, knees, etc. This representation resembles a human skeleton, and it allows to obtain a person's position relative to the camera on the robot. The library also assigns an ID for each person it identifies, allowing to track a specific person while he or she moves in front of the camera. That feature is used as an input for the navigation system, which will plan a path to follow a specific user in cluttered and dynamic environments.

### 3.5 Speech Recognition and Synthesizing

The voice is the form of man machine interaction most used to give commands to the robot in a more natural way, either through specific commands or natural language.

For recognition, we use the CMU PocketSphinx[15], which features greater flexibility for adaptation and personalization, allowing to adapt the dictionary and acoustic models to the problem of context. After the speech recognition, use the output of pocketsphinx to feed the state machine using the boost Regex library. With that we construct a grammar able to interpret the commands and fulfill the assigned tasks.

Pocketsphinx uses a statistical approach based on hidden Markov models (HMM)[16], and its architecture is defined in 5 modules: front-end, phonetic dictionary, acoustic model, language model and decoder.

In the process of synthesizing, we use Festival[17], a software that allows us to change the various aspects of voice, such as tone, speed of speech, among others, in order to ensure better understanding by the listener, allowing us to generate better interaction experience.

#### 3.6 Object Manipulation

Object manipulation plays an important role to interact with a home environment. To meet that requirement, BILL was given an arm based on the TurtleBot Arm [18], composed of 5 degrees of freedom (DoF) including a gripper, which currently allows the robot to grab lightweight objects. Since the developers provide an open source code for the TurtleBot Arm as a package of ROS, our team has integrated it with BILL's software. We are currently working on the improvement of its manipulation capabilities, enabling it to carry heavier loads in the future.

# 4 Experiments and results

In this session will show the experiments carried out in stages of object recognition and face recognitin.

In order to validate the development of object recognition, we conducted a training with fifteen different objects and generated a classifier model. For that, were taken 150 pictures of objects (10 for each object). It was also performed a variation on the SVM kernel in Linear, polynomial, Radial Basis and Sigmoid with a parameter optimization presented in [19]. The result of accuracy for each kernel was respectively 78.8%, 84.8% 78.8% and 72.7%. The polynomial kernel was chosen to present the best accuracy rate, and was used in the robot for object classification.

To improve the accuracy of the face recognition, the image of the faces are submited on a image processing method, using Histogram Equalization. The method was chosen after a experiment using a dataset of 20 faces of 62 people: 10 to the Fisherfaces training, 10 to the tests. Before the recognition process, the faces were submited to the methods Histogram Equalization, Gaussian and Bilateral Filter, and their combination. The amount of right identifications of the algorithm without methods were 43%. The results of the abstraction filters were less than 40%, and the Equalization Histogram and its combinations achieve results over than 50%, showing the impact of the adjustment on bad lightness from images in natural environments, as the image dataset used.

## 5 Conclusions and future work

This TDP presented the main functionalities of robot BILL developed by ACSO. Results were presented for object recognition using color and shape features with a 84.8% accuracy rate and for face recognition using FisherFaces with accuracy rate of 57%, in environments on bad conditions.

As future work, the authors of this TDP aim to improve some functionalities of BILL, including the object recognition with modeling more features on SVM classifier; face recognition using others paradigms; and implement a objects system localization in unknown environments using saliency and color features.

# Module Description

In order to provide completely autonomous operation, BILL owns two main modules of control: The High-level control, which includes algorithms to solve functionalities such as global task planning, navigation and tracking, recognition of objects and faces, user-interaction, among others. And a low-level to control sensors and actuators in the real world.

## **BILL Hardware Description**

The low-level module, running on Arduino ATmega 2560, is capable of receiving messages coming from high-level for transform in an action, like the desired speed and steering angle to the due motion control, with a classical PID controller. In addition to this, the low-level control is designed to supervise information about temperature and battery power to ensure the BILL's health. The hardware specifications are as follows:

- Base: One Arduino Mega 2560; Four motors IG32P 24VDC 190 RPM Gear Motor with Encoder; One Notebook Dell Inspiron 15R-4470, intel core i7; One digital buzzer. One RPLIDAR 360 degrees laser scaner. Three Sabertooths controlers. One LM35 linear temperature Sensor. three batterys 11.1 volts and 2800 mAh; One digital push button;
- Torso: Mini actuator Firgelli Automations; One Emergency switch;
- Arm: five Dynamixel-ax-12A; One ArbotiX-M; Maximum load: 1kg.
- Head: One Dynamixel-ax-12A; One Microsoft Kinect sensor; Two Microsoft life Cam HD-3000; One Rode Videomic Pro.

#### **BILL Software Description**

The high-level has tools developed by our team and other open source applications already available on ROS. Currently this control run on a laptop, where some strategic and tactical tasks are computed. The communication with the low-level is made through a USB-TTY serial connection. For our robot we are using the following softwares:

- Integration: Robot Operating System (ROS).
- Navigation, localization and mapping: Hector\_mapping.
- Face recognition: OpenCV library.
- Speech recognition: PocketSphinx library; Boost library.
- Speech generation: Festival.
- Object recognition: Point cloud library (PCL); OpenCV library; libSVM library.

#### References

- Morgan Quigley, Ken Conley, Brian Gerkey, Josh Faust, Tully Foote, Jeremy Leibs, Rob Wheeler, and Andrew Y Ng. Ros: an open-source robot operating system. In *ICRA workshop on open source software*, volume 3, page 5, 2009.
- Radu Bogdan Rusu and Steve Cousins. 3D is here: Point Cloud Library (PCL). In IEEE International Conference on Robotics and Automation (ICRA), Shanghai, China, May 9-13 2011.
- Martin A Fischler and Robert C Bolles. Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography. *Communications of the ACM*, 24(6):381–395, 1981.
- 4. Gary Bradski and Adrian Kaehler. Learning OpenCV: Computer vision with the OpenCV library. "O'Reilly Media, Inc.", 2008.
- Satoshi Suzuki et al. Topological structural analysis of digitized binary images by border following. *Computer Vision, Graphics, and Image Processing*, 30(1):32–46, 1985.
- Ming-Kuei Hu. Visual pattern recognition by moment invariants. Information Theory, IRE Transactions on, 8(2):179–187, 1962.
- Michael J. Jones Paul A. Viola. Rapid object detection using a boosted cascade of simple features. Computer Vision and Pattern Recognition. Proceedings of the 2001 IEEE Computer Society Conference, 1:I–511 – I–518, 2001.
- Pedro Pontes. Visage impacto dos filtros de abstração no reconhecimento facial em imagens. Master's thesis, FACULDADE DE ENGENHARIA DA UNIVERSI-DADE DO PORTO, Portugal, 2013.
- 9. Peter N. Belhumeur, João P. Hespanha, and David J. Kriegman. Eigenfaces vs. fisherfaces: Recognition using class specific linear projection, 1997.
- Raul Queiroz Feitosa, Carlos Eduardo Thomaz, and Álvaro Veiga. Comparing the performance of the discriminant analysis and rbf neural network for face recognition. International Conference on Information Systems Analysis and Synthesis, 1999.
- 11. S. Kohlbrecher, J. Meyer, O. von Stryk, and U. Klingauf. A flexible and scalable slam system with full 3d motion estimation. In *Proc. IEEE International Symposium on Safety, Security and Rescue Robotics (SSRR)*. IEEE, November 2011.
- Boris Lau, Christoph Sprunk, and Wolfram Burgard. Improved updating of Euclidean distance maps and Voronoi diagrams. In Proc. of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), pages 281–286, Taipei, Taiwan, 2010.
- James Neufeld. Autonomous Outdoor Navigation and Goal Finding. ProQuest, 2008.
- 14. Openni. Retrieved December 3, 2014, from http://openni.org/, 2014.
- 15. D Huggins-Daines. Pocketsphinx api documentation, 2010.
- Xuedong Huang, Alex Acero, Hsiao-Wuen Hon, et al. Spoken language processing, volume 18. Prentice Hall Englewood Cliffs, 2001.
- Alan Black, Paul Taylor, Richard Caley, Rob Clark, Korin Richmond, Simon King, Volker Strom, and Heiga Zen. The festival speech synthesis system, version 1.4. 2. Unpublished document available via http://www.cstr. ed. ac. uk/projects/festival. html, 2001.
- 18. Turtlebot arm. Retrieved February 3, 2015, from  $http://wiki.ros.org/turtlebot_arm, 2015$ .
- Chih-Wei Hsu, Chih-Chung Chang, Chih-Jen Lin, et al. A practical guide to support vector classification, 2003.