Markovito’s Team Description
RoboCup@Home 2014

L. Enrique Sucar, Eduardo Morales, Miguel A. Palacios-Alonso, Patrick Heyer, Irving Vásquez, David Carrillo, Elías Ruiz, Elías Reyes, Alejo Mosso, Rogerio Enríquez-Caldera, and José Tobon

Instituto Nacional de Astrofísica, Óptica y Electrónica,
Computer Science Department,
Luis Enrique Erro 1, 72840 Tonantzintla, Mexico
http://ccc.inaoep.mx/~markovito

Abstract. In this paper we present Sabina, a service robot developed by the Markovito team at INAOE. Sabina is based on a PatrolBot robot platform and incorporates a set of general purpose modules for service robots that achieve basic robot skills, such as map building; localization and navigation; object and people recognition and tracking; an human interaction using facial animation, speech and gestures and manipulation. All these modules are integrated in a layered behavior-based architecture using shared memory for communication and coordinated by a decision-theoretic planner. As an addition to Sabina’s capabilities, we also describe a novel approach to autonomous 3D object reconstruction. Markovito team have participated in the Robocup@Home category in previous Robocup competitions; in Turkey 2011 our team qualified for the second stage of the competition. In 2013, Markovito team won the Mexican RoboCup@home competition.

1 Sabina’s Hardware Platform.

Sabina is a service robot developed by the Markovito team (see Figure 1), it is based on a PatrolBot robot platform [1]. The platform has a sonar ring, two wheels, two motors with encoders, a Laser SICK LMS200, one video camera Canon VCC5, speakers, a directional microphone and an integrated PC. The team have added other devices such as an standard Laptop, two web cams, a Katana 6M arm with 5 DOF and a Kinect device.

2 Sabina’s Software Architecture

Sabina’s software architecture has been designed and developed as a layered behavior-based [2] architecture that uses shared memory for communication. As you can see in Figure 2 all the general purpose modules are integrated in this architecture. The architecture has three different levels:

– Functional Level: Here the modules interact with the robot sensors and actuators, relaying commands to the motor or retrieving information from the sensors.
– Execution level: Modules in this level interact with the functional level through shared memory. This level includes the modules to perform basic tasks such as navigation, localization, visual perception, human-robot interaction, etc.
– Decision level: This is the highest level in the architecture. Markov Decision Processes (MDPs) are used as a global planner to coordinate the execution level modules.

The layered structure and a transparent communication mechanism allow different configurations to be defined without modifying the modules and without affecting the rest of the system. In this architecture, a robot behavior is an independent software module that solves a particular problem, such as navigation or gesture recognition. The complete system was developed using mainly C/C++ languages and runs on Linux.

2.1 The Execution Level

We are implementing different general-purpose modules that are common to several services robot’s applications. All of these modules are in the execution
level on the Sabina's architecture. The libraries used in these modules are listed in Table 1.

**Table 1.** Modules and libraries used in Markovito.

<table>
<thead>
<tr>
<th>Module</th>
<th>Source code/Libraries</th>
</tr>
</thead>
<tbody>
<tr>
<td>Navigation</td>
<td>Arla, ARNL, MRPT</td>
</tr>
<tr>
<td>Vision</td>
<td>OpenCV, SIFT algorithm, ORB descriptors, OpenKinect</td>
</tr>
<tr>
<td>Interaction</td>
<td>Festival, Microsoft Speech Recognition, OpenGL Custom Render</td>
</tr>
<tr>
<td>Coordination</td>
<td>SPUDD (MPD)</td>
</tr>
</tbody>
</table>

**Map building and Planning.** The first task that a service robot has to do is to know its environment. For this purpose, a service robot requires a model or map of this environment. Sabina combines information from a laser scanner and odometer to construct an occupancy map. We use the ICP algorithm from Mobile Robot Programming Toolkit (MRPT) [3] to build this model. The ICP algorithm matches a point cloud and some reference. In this case, the point cloud are the current laser readings, the reference are the previous one. Figure 3 illustrates this process.

For planning purposes, a probabilistic roadmap (PRM) [4, 5] is built using a random generation of points in the configuration space. These points are joined if there is a free path between them and information is stored in a graph $G$. Given an initial configuration, $s$, and a goal configuration, $g$, the problem consists of connecting $s$ to $g$ in $G$. This process is illustrated in Figure 4.
Fig. 3. Sabina building a map of the environment using its SLAM algorithm based on MRPT ICP.

Fig. 4. Example of a PRM constructed. The PRM is build using a random generation points. The points are joined if there is a free path between them. Intermediate points in the path are given as goals to the navigation module.

Localization and Navigation. Due to odometric error, the ability for mobile robots to locate themselves in an environment is not only a fundamental problem in robotics but also a pre-requisite for safe navigation tasks. In order to locate itself, Sabina uses laser information. Given a set of laser readings, a MRPT particle filter process is performed to estimate the real robot position. Figure 5 shows the localization process.

Fig. 5. Sabina running the MRPT particle filter localization. a),b) c) Show the algorithm initialization. d) Depicts the algorithm convergence, the mean of the particles is the robot position estimation.

In order to control the robot’s movement to follow a specific path, we have implemented a simple but effective navigation module based on Aria actions [1]. Also we use the integrated ARNL library [1] for navigating in more complex environments.

Face Detection and Recognition. We have developed a face recognition system allowing a mobile robot to learn new faces and recognize them in indoor
environments (see Figure 6). First, the image is enhanced by equalizing its histogram and performing a local illumination compensation [6]. Next, we use an object detection scheme based on a boosted cascade of simple feature classifiers [7] to detect eyes, mouth, and nose. For each region, SIFT features [8] are extracted. The features in a sequence of images are compared to the models in a database using a Bayesian scheme.

![Face detection and recognition.](image)

**Fig. 6.** Face detection and recognition.

**Gesture Recognition.** To recognize gestures we are incorporating the Kinect device to take advantage of its depth information. Our approach allows the robot to simultaneously segment and recognize gestures from continuous video [11]. Figure 7a shows Sabina recognizing the attention gesture, Figure 7b shows a graphical description of the approach. We use dynamic windows to segment the video sequence. Each time step, the features belonging to every window is evaluated to get a gesture prediction. Windows that better fit the segment of the gesture, will produce a higher probability and thus their votes will have higher weight.

![Gesture recognition module.](image)

**Fig. 7.** Sabina’s recognition module for continuous video. a) Sabina is capturing a user gesture through the Kinect Device. b)Diagram illustrating the simultaneous segmentation and recognition of dynamic gestures. System has a video sequence as input. Then 5 windows are generated, one every time step. W1 starts at t0, W2 starts at t1, and so on. Each window is evaluated and voted, simultaneously until certain stop criteria is met.
**Human Tracking.** In this module we extract torso boundaries using a histogram and the back projection image [9] coupled with Haar functions [10] with a monocular camera. Our torso detection and tracking system is divided in two stages. The first stage is the torso localization process, that uses a face detection algorithm based on color histograms in RGB. Once the face is detected, the torso position is estimated based on human biometry. The color histogram of the torso is registered by this module. The second stage consists of tracking the torso using the color histogram obtained at the first stage, coupled with detectors based on motion and appearance information. Finally, a distance transformation is applied, considering a pinhole camera model.

**Object Recognition.** Currently we are exploring the use of the ORB algorithm. Basically, it fusions of FAST keypoint detector and BRIEF descriptor. It computes the intensity weighted centroid of patches with located corners at the center. The direction of the vector from each corner point to its centroid gives the orientation. To improve the rotation invariance, moments are computed with $x$ and $y$ which should be in a circular region of radius $r$, where $r$ is the size of the patch. This process is done on pre-processed images where background subtraction is done using depth information from the Microsoft kinect mounted on top of our robot.

**Manipulation and 3D Reconstruction.** For manipulation purposes we have added a Katana arm (see Figure 1b) to the Patrolbot platform. This arm is able of grasping objects which are inside its reachable space. Once that object and robot positions have been obtained inside the environment (point cloud), we plan the controls to reach the configuration which grasps the object by using a Rapidly-exploring Random Tree technique (RRT) [12]. Here our goal state is the grasping configuration (see Figure 8a). The environment where the RRT checks for collisions is given by an octree updated with the point cloud of the environment.

![Fig. 8. a) Sabina Taking an object. b) c) d) Autonomous reconstruction 3D experiment.](image)

Sabina also has the ability of building 3D models from real objects in its environment. We have developed an autonomous 3D object reconstruction algorithm. To generate the model, several scans from different configurations are taken by
the robot. For each configuration, the robot takes a scan, updates the incremental representation of the object and plans the next configuration. To plan the next configuration, we uniformly sample the robot configuration space and test each sample in order to evaluate their utility. The configuration with the highest utility is selected. In addition, the trajectory to reach the planned configuration is calculated with a Rapidly-exploring Random Tree. Figures 8b, 8c and 8d show some simulated experiments of this approach.

**Speech Recognition and Synthesis.** We use Microsoft Speech Recognition[14] engine and Kinect SDK libraries for speech recognition. Different dictionaries or sets of recognizable phrases are defined depending of the task to be performed by the robot. The system can identify only the phrases or words defined in its dictionary. The coordinator (MDP) sets the right set of phrases to be used by the speech recognition module on each task. For synthesis we use Festival [13].

**Facial Animation.** We have incorporated a set of emotions such as happiness, anger and surprise to Sabina by providing it with a friendly animated face (see Figure 9). The animation is done with key-frames interpolation. OpenGL is used to render the 3D model and key postures and timing information are defined *a priori*.

![Fig. 9. Sabina’s face.](image)

2.2 The Decision Level

The behavior modules are coordinated by a decision-theoretic controller based on MDPs [15]. An MDP is specified for each task and solved to obtain an optimal policy. In our current implementation we use a factored representation to specify the MDPs and SPUDD [16] to solve them. The model is specified manually by the programmer according to the task. We use an interactive approach to define the model.

3 Conclusions

Sabina hardware and software architecture have been described. The robot is based on a PatrolBot platform where a Katana arm, a kinect and some cameras
have been added in order to get a better perception performance and interact effectively with people. With this objective in mind we have also developed a set of general purpose modules, integrated in a layered behavior-based architecture. These characteristics allow Sabina to perform different RoboCup@Home tasks. Based on this framework and a PeopleBot platform we have participated in the Mexican Robotic tournament with our robot Markovito since 2007; and in the international Robocup@Home competition in 2009, 2011 and 2012. In 2013 we won the Mexican RoboCup@home competition with our new PatrolBot platform, Sabina.

References