Programming of a Mobile Robotic Manipulator through Demonstration

Lorenzo Peppoloni and Alessandro Di Fava
PERCRO Lab, Scuola Superiore S. Anna, Pisa 56100, Italy

Keywords: Programming by Demonstration, Manipulation, Autonomous Navigation, Semantic Learning.

Abstract: This paper presents an integrated robotic system capable of learning and executing manipulation tasks from a single human user's demonstration. The system capabilities are threefold. The system learns tasks from perceptual stimuli, models and stores the information in the form of semantic knowledge. The system may employ the model achieved to execute task in an way similar to the example shown and adapt the motion to robot own constraints in terms of physical limits and interferences. The system integrates perception and action algorithms in order to autonomously extrapolate the context in which to operate. It robustly changes its behavior according to the environment evolution. The performances of the system have been verified through a series of tests. The tests run on the Kuka youBot platform and all the tools and algorithms are integrated into Willow Garage "Robotic Operating System" (ROS).

1 INTRODUCTION

Robot programming by demonstration (PbD) is a very large research topic, it includes areas such as human-robot interaction, machine learning, machine vision and motor control. Here we present an approach to perform domestic activities teached through examples. We focused on the following aspects: the ability to learn manipulation activities, to detect relevant information and to figure out a related symbolic model; the integration of algorithms for perception and action into a single architecture; to map tasks execution to robot operational space; to manage and balance the robot constraints (obstructions, manipulation and visual field) with respect to the environment changes. To achieve this functionality we integrated several components, such as: object recognition (Lai et al., 2011), grasping (Ciocarlie et al., 2010), laser range localization and navigation (Marder-Eppstein et al., 2010), computer vision (Malbezin et al., 2002), skeleton tracker and perception (Beetz et al., 2010). We dedicated a particular effort to achieve an effective integration that manages all reciprocal constraints. We chose to adopt an interoperable platform among the set of available platforms (Pangercic et al., 2010; Rusu et al., 2009). We use the software ROS (Robot Operating System) (Quigley et al., 2009), a couple of Microsoft Kinect cameras (Khoshelham, 2011) and the Kuka youBot (Bischoff et al., 2011). The system observes a human setting and clearing a table up and learns how to do it autonomously. It creates a symbolic deterministic model of the shown activities and uses this model to replicate similar tasks in different contexts. Our system learns through One-Shot-Learning (only one demonstration).

2 PLATFORM DESCRIPTION

The platform employs a Kuka youBot opportunely modified. Figure 1 shows the overall setup of the robot. The platform consists of an omni-directional wheeled base **(1)**, an onboard pc and a 5-DOF arm **(2)** with a two-fingers gripper. The platform is integrated with laptop **(3)** and two Microsoft Kinect cameras.

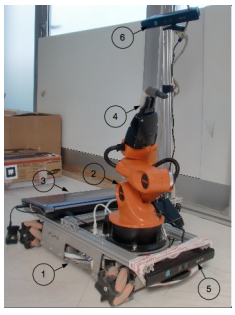

Figure 1: Photo of the modified platform.

The gripper shape **(4)** was modified to improve the grasp of the objects and to maximize the reach-

Peppoloni L. and Di Fava A..
Programming of a Mobile Robotic Manipulator through Demonstration.
DOI: 10.5220/0004041004680471
In *Proceedings of the 9th International Conference on Informatics in Control, Automation and Robotics* (ICINCO-2012), pages 468-471
ISBN: 978-989-8565-22-8

able workspace. A bottom camera **(5)** operates as a laser to locate the robot in the environment and to avoid the obstacles. A top camera **(6)** detects objects, user motions, and table markers. A laptop was added to increase computing power. The laptop communicates with the platform via EtherCAT and with the two cameras via USB connections.

Figure 2: Photo of the robot working environment.

Figure 2 shows the robot working environment. The edges of the environment are real walls that limit workspace and facilitate the laser mapping task. The map of the work environment is provided to the robot as 2D format. In the map there are operating surfaces, the one in the top left represents the "table" e the other in the bottom right represents the "sink". The box in the bottom left is an additional barrier inserted to improve the asymmetry of the scenario and to make more robust the localization algorithms which will be described later. The household objects that will be manipulated to perform the actions are the bowl, and the glass.

3 SOFTWARE OVERVIEW

The several modules, that compose our platform, need to communicate one another. To satisfy these needs we use ROS. The platform organizes its activities into phases. Two major phases have been identified: Learn and Perform. In the Learn phase the robot observes the user and learns how to perform tasks. Voice commands are used to instruct the robot which elements to focus on. In the Perform phase the robot executes a goal. The Figure 3 shows how, using ROS, these phases have been integrated into a unique architecture.

In the Learn phase, several modules provide information about the objects, the tables and the human interaction. The Audio module **(7)** gets voice commands given during the demonstration. The Learner module **(a)**, in the Cognition manager, stores action parameters, such as position, object type, etc. into the objects' Knowledge DB **(b)**. The Learner removes details from demonstration that are not relevant for the execution of the task. The result is represented as a simple map encoding actions that the platform can perform. The Learner scans the demonstration and creates a symbolic sequence of actions. To facilitate the recognition of actions, the Knowledge DB **(b)** also provides the Learner with information on the objects' meshes. User inputs are only necessary for the initial demonstration.

In the Perform phase, the Performer **(c)** instantiates high level solutions that solve a given goal. It uses the perception to determine initial conditions and plans the best policy to perform actions. Then the Sensorimotor manager maps these solutions on the platform, by transforming the requested actions into executable motions. For implementation details see (Di Fava et al., 2012).

At lower level, several modules concur to furnish two basic functionalities: Perception and Action. Each module comprises different nodes.

4 IMPLEMENTATION

Navigation. During the initial environmental configuration the robot can recognize the context (tables positions, robot start position and objects on furniture) and adapt its behavior to diminish fault risk, it is done in the starting procedure. First, the robot relocates itself (through marker in the low camera field of vision). Then it locates the furniture and save its position, following the explorer algorithm: 1. Navigate to the workspace center; 2. start pivoting until a table/sink marker is detected; 3. if a marker is found, navigate to the table/sink; 4. approach table/sink; 5. return furniture position in map frame. The navigation is used to autonomously navigate among table and the sink. This process is composed of four phases: 1. localization: the robot uses a hybrid system combining AMCL (Fox et al., 1999) and a vision-aided localization system based on landmarks; 2. global navigation with path planning, combining A* and DWA algorithms (Fox et al., 1997), and obstacle avoidance based on a costmap; 3. second localization to improve accuracy; 4. table/sink approach: implemented through a simple visual control loop, during this phase the platform is moved in a known position in the table/sink marker frame.

Grasping and Manipulation. We implemented ROS services to pick, place and store objects onboard. This gives the possibility to move objects among the furniture with the arm free and without obstruing the vision system. On the other hand, the robot is blind when manipulating objects on its body, this makes

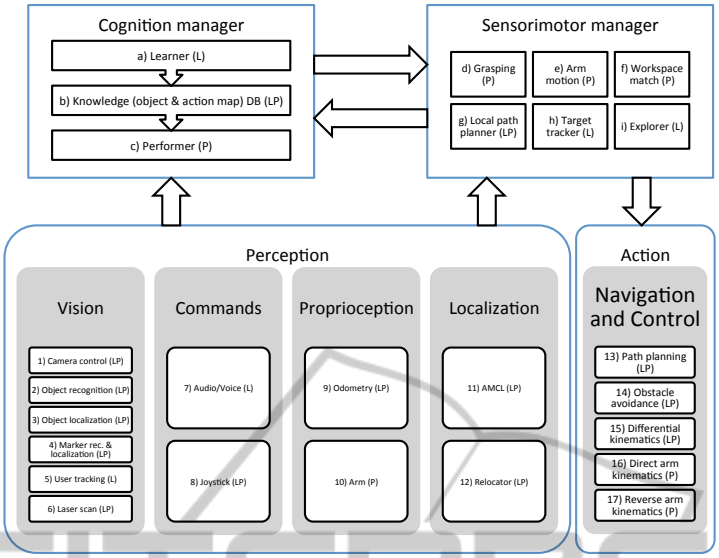

Figure 3: Implemented system architecture. Four macro-blocks interact each other. Each one containing modules, and relative nodes. The two blocks on the top (Cognition manager and Sensorimotor manager) represent the high-level functionalities of control. These blocks interact each other and control the low-level modules of Action and Perception. The active phase determines which module at low-level has to operate. The phase association is reported in parentheses: Learn **(L)** and Perform **(P)**.

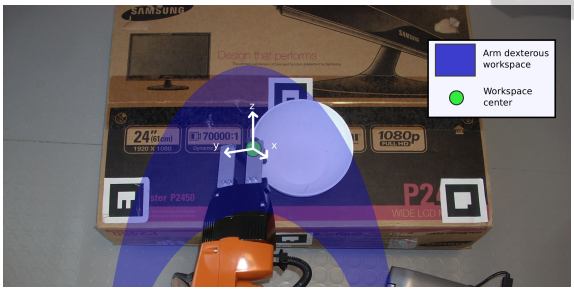

Figure 4: Dexterous workspace with its center is shown in blue. The bowl pre-grasp point on the edge is shown in green. It can be seen how the two points are superimposed, combining arm and base movements. The gripper frame is also drawn with x axis pointing downward.

the system chattering sensitive during navigation. We chose to divide the grasp procedure in three phases (poses for every phase has been computed with the Graspit! Simulator (Miller and Allen, 2004)): approach (pre-grasp position), alignment (grasp pose) and grasp (gripper closure). We assumed only a movement along the gripper x axis among pre-grasp and grasp positions. To study the grasp problem, we computed the manipulator dexterous workspace and we performed cuts at grasp and pre-grasp heights for every object, obtaining the planar workspaces. The actual grasp movement is implemented as a point-to-point trajectory, with the arm controlled in position.

Approaching for Fine Manipulation. Using pre-grasp and grasp information, we implemented an arm-base coordination policy. The pre-grasp pose is computed in the frame of the robot arm using a visual-aided algorithm. For what concerns the base movements during manipulation, we implemented a Maximum dexterity policy, by moving the base to have the pre-grasp pose in the center of the dexterous worspace (Figure 4).

5 TESTS

We validated the platform behavior with different types of tests. In all experiments we estimated the objects pose, without making any assumption on their orientations or positions.

Coming to the goodness of the explorer algorithm, we computed the probability of success. We carried out 10 tests placing the table in different positions. The robot also started from different positions, always with a visible marker. The system detected the table in 8 of 10 tests (80%). The failures are due to the table closer than the camera minimum range or to too much inclination between the markers and the camera.

After having determined the prerequisite for success, we arranged a new set of experiments to estimate the accuracy of localization. Only the robot initial position changed in these tests, while the table remains in the same position. The results of 20 tests show a

variance of 8×10^{-4} $[m^2]$ for x, 2.3×10^{-3} $[m^2]$ for y, 2.9×10^{-3} $[rad^2]$ for the yaw. Even having collected few samples, error distribution fits quite well with a Gaussian ($p \leq 0.2$). Considering a range of values $\mu + 3\sigma$ for the localization, with accuracy ($p \leq 0.003$), the platform has to maintain a distance greater than 9 cm along x and 15 cm along y from the target position to avoid collisions.

Then we examined the accuracy of the table approach procedure. Both table and robot position changed between tests. The results of 10 tests show a variance of 1.12×10^{-4} $[m^2]$ for x, 1.14×10^{-4} $[m^2]$ for y, 4.4×10^{-4} $[rad^2]$ for the yaw. Considering a range of values $\mu + 3\sigma$ for the localization, the platform has to maintain a distance about 3 cm along x and y, taking into account a yaw error slightly greater than 3 degrees, to approach the table without collisions. An error of 3 cm does not invalidate the platform capability of grasping (Figure 4).

The last set of experiments determines the goodness and the accuracy of objects grasping. We carried out 10 tests placing the object in different areas of the table and starting the robot always with the table markers visible. The arm grasps the object in 9 of 10 tests (90%). The tests show a variance of 3.2×10^{-7} $[m^2]$ for x, 2.4×10^{-5} $[m^2]$ for y. Considering a range of values $\mu + 3\sigma$ for the object recognition, we can estimate that the error made in recognizing an object falls within the limits of about 0.17 cm along x and 1.48 cm along y. The error obtained, being smaller than the opening wideness of the gripper (2.3 cm), does not invalidate the grasp capability of the platform. No phase is secure/robust, but the consequentiality of phases ensures a progressive refinement that avoids collisions and gives capability to grasp.

6 CONCLUSIONS AND FUTURE WORK

We presented and validated a learning by demonstration system. It integrates action and perception algorithms to learn and execute household tasks adapting them to its physicality. We validated every module efficiency and integration through a series of experimental tests. We plan to adapt the system to be controllerd with biometric readings to use the robot as an auxiliary body for the human user and to have arm movements learned through demonstrated examples (Avizzano, 2012).

REFERENCES

Avizzano, C. A. (2012). Guided latent space regression for human motion generation. In *Robotic and Autonomous Systems, Special Issue on SKILLS*.

Beetz, M., Tenorth, M., Jain, D., and Bandouch, J. (2010). Towards automated models of activities of daily life. In *Technology and Disability*. IOS Press.

Bischoff, R., Huggenberger, U., and Prassler, E. (2011). Kuka youbot - a mobile manipulator for research and education. In *In Proc. of the IEEE International Conference on Robotics and Automation*.

Ciocarlie, M., Hsiao, K., et al. (2010). Towards reliable grasping and manipulation in household environments. In *In Proc. of Intl. Symposium on Experimental Robotics (ISER)*.

Di Fava, A., Peppoloni, L., Avizzano, C. A., and Ruffaldi, E. (2012). A cognitive learning architecture for a mobile robotic manipulator. In *Robot and Human Interactive Communication, 2012. Proceedings. RO-MAN 2012 (to be submitted)*.

Fox, D., Burgard, W., Dellaert, F., and Thrun, S. (1999). Monte carlo localization: Efficient position estimation for mobile robots. In *In Proc. of the National Conference on Artificial Intelligence (AAAI)*.

Fox, D., Burgard, W., and Thrun, S. (1997). The dynamic window approach to collision avoidance. In *in IEEE Robotics and Automation, Vol. 4, No. 1*.

Khoshelham, K. (2011). Accuracy analysis of kinect depth data. In *In Proc. of ISPRS Workshop Laser Scanning 2011*.

Lai, K., Bo, L., Ren, X., and Fox, D. (2011). A large-scale hierarchical multi-view rgb-d object dataset. In *In Proc. of the IEEE International Conference on Robotics and Automation*.

Malbezin, P., Piekarski, W., and Thomas, B. (2002). Measuring artootkit accuracy in long distance tracking experiments. In *In Proc. of the Augmented Reality Toolkit, The First IEEE International Workshop*.

Marder-Eppstein, E., Berger, E., et al. (2010). The office marathon: Robust navigation in an indoor office environment. In *In Proc. of the IEEE International Conference on Robotics and Automation*.

Miller, A. and Allen, P. K. (2004). Graspit!: A versatile simulator for robotic grasping. In *in IEEE Robotics and Automation Magazine, V. 11, No.4*.

Pangercic, D., Tenorth, M., Jain, D., and Beetz, M. (2010). Combining perception and knowledge processing for everyday manipulation. In *In Proc. of Intelligent Robots and Systems (IROS), 2010 IEEE/RSJ International Conference on*.

Quigley, M., Conley, K., et al. (2009). Ros: an open-source robot operating system. In *In Proc. of the ICRA Workshop on Open Source Software*.

Rusu, R. B., Sucan, I. A., et al. (2009). Real-time perception-guided motion planning for a personal robot. In *In Proc. of Intelligent Robots and Systems (IROS), 2009 IEEE/RSJ International Conference on*.