BerlinUnited@Home 2013 Team Description Paper

Adalberto Llarena, Jan F. Boldt, Nikolai S. Steinke,
Heinrich Engelmeyer, Raúl Rojas

Department of Mathematics and Computer Science,
Freie Universität Berlin, 14195 Berlin, Germany
bu_athome@googlegroups.com,
http://athome.berlinunited.org

Abstract. This paper presents the approaches and technology used by the team BerlinUnited@Home with the intention to participate in the Robocup@Home league with a prototype of autonomous electric wheelchair, capable of recognizing spoken commands and performing complex behaviors like searching and grasping objects dynamic environments. The motivation of such work arises from the need of helping handicapped or aged people to perform some tasks through a small autonomous vehicle, capable of driving by itself, even when the user is incapable or in case of danger, based on the experience of Prof. Dr. Raúl Rojas and his colleagues in the field of autonomous vehicles.

1 Introduction

1.1 Robocup@Home

Robocup [1] is an international research effort; where the most recent techniques of robotics and scientific computing join together with the intention of emulate some common tasks for the human beings such as playing soccer or cleaning a table. In this sense, the Robocup@home [2] league has focused in the development of robots capable of solving common task for humans (so called "Service Robots") in a typical human-living environment: it's own house.

Since then, several kinds of robots have participated in this league, including small robots as the four legged Sony® AIBO and outdoor navigating vehicles like Segway®. Mostly, two-wheeled differential drive robots have been the most popular mobile platforms in the @Home league.

Fig. 1. Some robots used in the Robocup@Home league in the past

1.2 The Autonomos Project

Currently, mainly two groups have succeeded in the development of an autonomous self-driving car: the Google® Group [3] of Prof. Sebastian Thrun, and Autonomos Labs® [4] with Prof. Raúl Rojas at the Free University of Berlin (Fig. 2). Based on its previous experience in outdoor environments a new challenge became two years ago: to develop an indoor autonomous vehicle, capable of bringing aid to handicapped or physically disabled people. The most robust platform for that purposes (even capable of navigating in outdoor scenarios) is an electric chair. In this field, the Otto Bock® company is famous in Germany for their experience in the field of electric wheelchairs, robustness and overall quality.

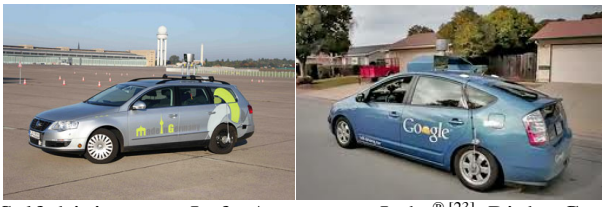

Fig. 2. Self-driving cars. Left: Autonomos Labs® [23]. Right: Google® [24]

1.3 Components of the Autonomous Electric Wheelchair Architecture

Our hardware consists mainly of three main components: a Otto Bock® Xeno [5] electric wheelchair capable of navigating at a top speed of 30 km/h, a Neuronics® Katana [6] robotic arm as grasping device, and a Robotis® Darwin-OP [7] humanoid robot as the autonomous user or "brain" in our approach (we ignore if a human person seated in the chair would be allowed in the Robocup@Home league to issue commands). The three robots together should interact to solve simple tasks such as present themselves to the jury or finding and bringing objects inside a dynamic environment as a house or apartment.

Fig. 3. Main components of the autonomous wheelchair project

In this case, each robot has a specific ability that contributes to the final accomplishment of a global task. While the wheelchair can navigate pretty fast in an indoor or outdoor environment (but only navigating), the robotic arm is only capable of manipulating objects, while the humanoid robot will be capable of both walking and grasping small objects (with a pair of grasping "hands" that we have already bought) but at very low speeds and less confidently than a robotic arm. In this form, every one of our robots would eventually need the help of the other two. This arises the need of more complex and synchronized decision plans and models (like a mixture of MDPs, synchronized through actions), than a classical sequential script of finite-state machine.

2 Module Description

In our approach, there are two main modules: the low-level perception and control module (that runs in a small Intel® Atom ASUS® notebook) and a high-level module that includes algorithms for localization, navigation, artificial vision, grasping control, global task planning and user-interaction.

2.1 Internal Sensors

The electric wheelchair has mainly two internal sensors: a) encoders for estimating the speed of the left and right rear wheels and b) encoders used for internally setting the steering angle of the front wheels, in order to be in accordance with the instantaneous center of curvature ICC [8] defined by the difference between the left and right rear wheels speed. Although the chair has two small free-rolling wheels at the front, their heading angle can be set independently. In this way, the internal chair electronics applies a differential drive model where two speed values -rotational speed and translational speed- are normally computed by the chair electronics (in the absence of control information coming from the low-level controller) every time a user operates the chair with the joystick installed on board.

By the other hand, the Katana arm has several position encoders for internal control while Darwin-OP only uses an accelerometer, apart from a factory-assembled internal magnetic encoder, incorporated in every one of their MX28 servomotors.

2.2 External Sensors

The electric wheelchair has been adapted to carry two SICK S300P [9] laser range finders, one at the front and one at the back. The front laser has a view angle of 270º while the laser mounted at the back is able to scan a maximum angle of 170º. Both laser scanners have an angular resolution of 0.5 degrees. A Microsoft® Kinect has been added to use its information in the mapping, localization and human-detection. A Minoru® 3D stereo camera and a VSTONE® Omnidirectional sensor are being

mounted for gesture recognition and object localization. Finally, we will add a microphone array to improve the speech recognition in crowded environments.

The Katana robotic arm does not incorporate any external sensor while the only external sensor mounted in the Darwin-OP is the off-the-shell Phillips® monocular camera.

2.3 Actuators

The wheelchair is capable of change independently the speed of its rear wheels (differential drive) and the heading of its front wheels. The Katana arm has five degrees of Freedom, while Darwin-OP has 20 servomotors, plus two servomotors that we added one at each arm, in order to be able to grasp small objects.

2.4 Low-level Control Module

A low-level module, installed on a small computer is connected directly to the wheelchair through an USB-CAN-bus converter. It receives and sends messages from and to the wheelchair internal controller, as the desired speed and steering angle. Also, that computer is capable of receiving the information coming from two laser rangefinders and the odometry. Then, it computes attraction and repulsion forces given a target location and finally controls the chair with a classical PID controller that receives the motion vector coming from the potential fields computation. This module sends information (odometry, current location, laser data and robot state) to the high-level module through standard TCP/UDP protocols. Although this module is able to compute fast-reactive control behaviors (like person-following or free-form navigation) many of the planning and navigation tasks are performed at higher levels.

2.5 High-level Control Module

This module is in charge of computing some tasks as map building, robot localization, motion planning, processing of the Microsoft® Kinect data, vision processing, object detection/manipulation, face/gesture recognition and user interaction, among others. Currently, as some of those modules are already developed for the autonomous car and they use the OROCOS Toolchain [10], we are in the process of migrating some of these modules to ROS [11], as we want to make available for the community some of these tools that we already have and, by the other hand, we would like to incorporate hardware and software already compatible with ROS in a much faster way.

2.6 SLAM and Map Building Modules

We use currently the Mobile Robot Programming Toolkit MRTP [12] as a tool for incorporating Particle Filter localization and Interactive Closest Point Slam (ICP-Slam), as well as Rao-Blackwell Particle Filters and Kinect sensor data processing for

building 3D world representations of the robot environments, useful for detecting objects or planning. We also use BOOST [13] for the serialization of C++ classes.

As many of the abovementioned Toolchains and Toolkits have many dependencies and they may consume much computing resources, we are also working in our own implementation of some those solutions, those that we observe that perform better with less resources. As an example, we have already developed a fast Monte Carlo particle filter localization MCL library in C++ that can use the laser data coming from a single laser rangefinder to compute in real time a good estimate of the robot's pose and a ICP-slam routine in ANSI C. These small libraries can be added to our low-level module without a major reduction of the overall performance. They could also fit in a small microprocessor board to avoid the need of a netbook. There are also other methods being incorporated [14] that are capable to provide a good estimation with much less resources than the MRPT libraries.

2.7 Planning Modules

In our approach there are also low-level and high-level planners. While a low level planner computes for instance the route to be followed by the wheelchair in order to go from one place to another, or the sequence of motions the robotic arm must perform for grasping an object, a high-level planner must deal with global goals (those than could involve many sequential or simultaneous actions performed by the individual robots). We have incorporated a NASA CLIPS Expert System [15] as a planner, to deal with a sequence of orders in the form of *rules* and *facts*. Whenever something has changed in the robot's environment (even when issuing an order) then comes a *fact* that modifies the world state (like a fact: *follow-human*). Individual facts can fire behavior sequences through *rules*. A rule, specifies the set of conditions (facts) that must be met to fire an action. For example, the sets of rules:

```
(deffunction followme ()
            (say "please put in front of me, and start walking")
            (assert (startx (xpos)))
            (assert (starty (ypos)))
            (assert (starta (apos)))
            (follow_human)
)
(deffunction goback ()
            (assert (return-start))
            (run)
)
(defrule send-start
            ?f <- (return-start) ?f2 <- (startx ?x)
            ?f3 <- (starty ?y)   ?f4 <- (starta ?a)
            =>
            (say "ok, followme")
            (go ?x ?y ?a)
            (retract ?f ?f2 ?f3 ?f4)
)
```

define the behavior of returning to the starting location after performing a *follow-me* command. Every *deffunction* <name> (*arguments*) => (*actions*) defines a rule that is fired when all the argument facts ($?f_1$ <- ($fact_1$), $?f_2$ <- ($fact_2$) … $?f_n$ <- ($fact_n$)) are valid (i.e. exist and have a value not *null*). In this example, the *assert* clause adds facts that consist in the current robot pose at the moment that the "*follow me*" command was issued. Once the user has issued a "*go back*" command, the fact *return-start* is asserted (is added in the fact list) and the rule *send-start* is then fired-up. This rule executes a motion to the initial start location *(go ?x ?y ?a)* and *retracts* (deletes) those facts that have fired the rule (otherwise it would "run forever").

2.8 Human-Robot Interface

According to [16, 17], we convert all issued commands in a Conceptual Dependency Primitive CDP. In this way, a command like "*robot, please follow me*" is converted to the primitive:

atrans (actor: *robot*, object: *robot*, from: *robot*, to: *human*)

where *atrans* means "to change the location of something" and needs some parameters: an *actor* that performs the change of location, the *object* that will be transferred, the *place* from the object will be taken and the final *destination* for the object being transferred. In this basic example, it is clear that the word *robot* can have multiple meanings as it can represent an actor, an object or a place. If we use (*assert* <fact>) in CLIPS

```
(assert (atrans (robot, robot, robot, human)))
```

and create a rule that executes a (*go* x_{human}, y_{human}) whenever an *atrans* fact exists and does not retract the *atrans* fact unless a *stop* fact exist, if every time the pose of the human is updated according with the robot sensors, then that simple set of rules will make the robot to follow a human continuously until receiving a "*stop*" order. In a similar way, every spoken command can be coded in an *n*-tuple of a fact (the action) followed by *symbols* (the possible values for actors, objects, places and so on) that can be treated individually with a sequence of actions.

2.9 Gesture recognition and Object Recognition/Manipulation

In the field of human recognition and gesture recognition, we are currently adapting our work in [18] to include it in our OROCOS Platform. As the wheelchair is equipped with a Kinect, then there would not be any trouble in incorporating this kind of recognition to our high-level planning module. Also, we are adapting our previous work in [19] to use it as an object recognition module and working in the control of the Katana arm [20] for grasping objects.

2.10 Future Work

Although it is possible to build in CLIPS an action planner as the one explained in the previous section, it has two inconveniences: 1) the facts and actions must be deterministic and 2) a fact can *exist* or not *exist* (true or false), for example a glass can have water or not. In the case of complex behaviors, for example, if we have a fact that represent that the object *shoes* is at the bedroom, when the robot arrives to the bedroom and that object is not present, then a bigger sub-problem starts: the robot must first try to find the object in the rest of rooms of the house by itself, but that implies a policy. Probabilities would be useful for representing, for instance, that there is almost null probability that the shoes could be inside the fridge, or in the washing machine. Of course the robot could just return with the user and tell him that the object is not present in the room and wait for the next place to find, but that wouldn't be an "intelligent behavior". We wouldn't like to buy an expensive service robot that will always ask what to do if anything goes wrong.

There are another kind of approaches that we have tried in the past and can be really useful when dealing with uncertainty, like Markov Decision Processes MDPs. Currently, based on [21] in the field of using MDPs to model robot soccer playing scenarios with the possibility of synchronizing individual MDPs to achieve a global task (receiving a *reward* or *penalty*), our team is also analyzing accomplishing a global task through the optimization of a joint of individual MDPs (one for every robot). In this way more "fluid" or "adaptive" plans could be used to solve a task involving many actors than just following a fixed sequential plan that could fail if anything results as unexpected. We are experimenting with reinforced learning to adapt those models to real-life scenarios, where the pre-calculated probabilities could differ substantially from the actual values. That is what usually happens. Remember the phrase: "Anything that can go wrong will go wrong".

3 Conclusions and Discussion

In this paper we have briefly described our motivation to take autonomous vehicles as the self-driving car into a maybe more challenging world: a house or an office. Inspired by the idea of developing an automated vehicle that could become not only a help for handicapped or aged people, but a real extension of their bodies if they could with it overcome some of their limitations in motion or speech (if we succeed in using another transducers as a helmet that can be used for driving a car with some EEG signals [22]). If we use all these technologies to transform complex behaviors as walking or preparing the food into something much more simpler like a small primitive and symbols, then something almost impossible for some people like "please follow me" would turn into a small *primitive* or even just into a simple *thought*. With the acquired experience of Prof. Raul Rojas in AI, the experience of Adalberto Llarena in the Robocup Small-Size, At home and Humanoid Kidsize leagues, plus the experience in self-driving cars of the rest of our teammates, we expect to have a good participation in Robocup Netherlands 2013.

References

1. http://www.robocup.org
2. http://www.ai.rug.nl/robocupathome/, 2006
3. http://en.wikipedia.org/wiki/Google_driverless_car
4. http://autonomos.inf.fu-berlin.de/
5. http://www.ottobock.com/cps/rde/xchg/ob_com_en/hs.xsl/24065.html
6. http://www.ros.org/wiki/katana_driver
7. http://www.robotis.com/xe/darwin_en
8. Latombe, Robot Motion Planning. Massachusetts, USA: Kluwer Academic Publishers, 1991
9. http://www.sick.com/group/EN/home/products/product_portfolio/optoelectronic_protective_devices/Pages/safetylaserscanners.aspx
10. http://www.orocos.org
11. http://www.ros.org
12. http://www.mrpt.org
13. http://www.boost.org
14. Adalberto Llarena, Jesus Savage, Angel Kuri, Boris Escalante, Odometry-Based Viterbi Localization with Artificial Neural Networks and Laser Range Finders for Mobile Robots", Journal of Intelligent and Robotic Systems. (2011 DOI: 10.1007/s10846-011-9627-8
15. CLIPS Reference Manual Version 6.0. Technical Report Number JSC-25012. Software Technology Branch, Lyndon B. Johnson Space Center, Houston, TX, 1994
16. Jesús Savage, Adalberto LLarena, Gerardo Carrera, Sergio Cuellar, David Esparza, Yukihiro Minami, Ulises Peñuelas; ViRbot: A System for the Operation of Mobile Robots. RoboCup 2007 Symposium, Atlanta, EU, 2007
17. Schank, Roger: 1972, 'Conceptual Dependency: A Theory of Natural Language Understanding', Cognitive Psychology 3, 552-631.
18. 2012 Simon Lang, Marco Block, Raúl Rojas: "Sign Language Recognition Using Kinect", in: 11th International Conference on Artificial Intelligence and Soft Computing (ICAISC 2012), Zakopane/Poland, published by Springer in the Lecture Notes in Artificial Intelligence series, Part I, LNCS 7267, pp. 394-402, 2012.
19. 2012 Markus Lindner, Marco Block, Raul Rojas: "Object Recognition Using Summed Features Classifier" in 11th International Conference on Artificial Intelligence and Soft Computing (ICAISC 2012), Zakopane/Poland, Zakopane/Poland, published by Springer in the Lecture Notes in Artificial Intelligence series, Part I, LNCS 7267, pp. 543-550.
20. Heinrich Engelmeyer: Autonomes Greifen eines Objekts mit einem Roboterarm. http://www.inf.fu-berlin.de/inst/ag-ki/rojas_home/documents/Betreute_Arbeiten/Diplomarbeit_Engelmeyer.pdf
21. Llarena, A., Rosenblueth, D.A.: Model checking applied to humanoid robotic soccer. In: Proc. Federation of International Robot-Soccer Association (FIRA 2012), pp. 256–269 (2012). Lecture Notes in Computer Science No. 7429
22. 2013 Daniel Göhring, David Latotzky, Miao Wang, Raúl Rojas: Semi-autonomous Car Control Using Brain Computer Interfaces, Springer Verlag, Intelligent Autonomous Systems 12, Advances in Intelligent Systems and Computing, 2013, Volume 194, Part 1, 393-408, DOI: 10.1007/978-3-642-33932-5_37
23. http://inhabitat.com/self-driving-dial-a-cars-could-reduce-traffic-by-up-to-80/
24. http://www.techzost.com/2012/06/googles-self-driving-car-meets-you.html