

Robotics4.NET: Embodied Framework for Programming Robots at University of Pisa

Antonio Cisternino, Diego Colombo

Dipartimento di Informatica, L.go B. Pontecorvo, 3, I-56127, Pisa, Italy
{cisterni,colombod}@di.unipi.it

Abstract. In this paper we present ongoing research at Computer Science Department at University of Pisa. The focus of our research is on a software infrastructure suitable for programming robots sharing the environment with people. We have tested our framework against different kind of robot architectures, ranging from toys like Lego Mindstorms, RoboSapien, to hobbyist platforms as ERI, to a research platform developed at our University.

1 Introduction

Robotics is ready for targeting the mass market; toy robots, and robotic vacuum cleaners, are already available in stores. These systems differ from industrial robots: they share the same space of humans, thus their control programs should be robust and flexible enough to be accepted by people, not simply correct. One of the authors still painfully remembers when he had to remove batteries from his son's Furby when the robot was pretending to play during night.

But what is it the gap between correctness and robustness? If we go to visit an industrial plant we all agree that we have to pay attention because robotics systems make certain assumptions about their own environment. In such a context we are subject to the rules imposed by these systems: rules made to simplify the deployment of robots and to maximize the throughput of the production process. As a consequence, control software of this kind of systems can rely on several simplifying assumptions like assuming that a path is clear by construction.

Besides, if it is a robot paying a visit to us, at home, or wherever people are used to live, we are not so keen to change our behaviour in order to simplify the life of the robot. If we buy a robotic vacuum cleaner we expect that it will avoid us if we are on its way. We also expect that it adapt its behaviour to our preferences instead of forcing us to always repeat the same instructions.

Thus there is a relevant gap between the control program of robots that are employed in industries and those we expect to bring at home. In the first case we expect performance, correctness and precision, whereas in the second flexibility and robustness are more important in the second case. In our research we focus thus on software for programming robots, with particular attention to the second kind of scenario, where robots are guests in the people environment.

2 Teaching Programming Using Robots

Controlling robots is indeed fun, and it is a good way to introduce the basic notions of computer programming. Lego Mindstorms includes a programmable “brick” to be used as a brain of some robot realized with the popular construction system. We developed SharpStorms [1], a compiler for .NET that translates .NET binaries into the bytecode for the Lego virtual machine. VisualStorms integrates SharpStorms into Visual Studio, allowing students to program Lego-based robots with a standard development system.

We believe that the first wave of robots in homes will be based on toys: for instance this year the RoboSapien [2] robot has been successful worldwide. These systems are mainly remote controlled: the robot has a number of actuators, but it lacks of sensors (usually only touch sensors are available). We have developed a .NET library for controlling RoboSapien [3] using a PocketPC based PDA. Based on this package we are now able to use the PDA as an intelligent controller for RoboSapien, with an interface that allow the user to play the role of a rich sensor system for the robot: through the interface you can provide more information to the running program than the mere touch information provided by on-board sensors.

3 Lack of perception in low cost systems

In our experience it is difficult to obtain robust software for controlling robots with a small set of sensors. We have tested the ER1 robotic kit from Evolution Robotics [4]: it is a configurable set of aluminium sticks with stepped motors and infrared sensors. The system includes also a Webcam. Using the kit it is possible to build a frame for a laptop that is responsible for controlling the robot. Sensors and motor are connected through USB and a program shipped with the kit is responsible for controlling them.

We have used ER1 to produce a map of wireless signal strength in the building. The control program, based on Robotics4.NET, was completely reactive, using infrared sensors to find the least occluded direction. The system works fine, and has been able to produce the desired map, though it often makes the wrong decision because there is a wide range of materials that are invisible to the IR sensors.

Of course we can try to combine the input of IR sensors with the image from the camera, but this requires more computing power. Thus it seems that low cost devices have to cope with lack of perception; this is can be a relevant issue if robustness is valuable property of the system: people wouldn't tolerate a robotic vacuum cleaner attempting to suck a cat because it is similar to the fur it is supposed to clean.

4 What happens if your PC gets the wheels?

We think that a first important step towards having robots sharing the same environment of human people is to figure out a set of basic behaviours that allow a robot to be acceptable by humans. We have built a PC-based robot capable of moving

indoor, on a floor of a building. The robot [5] has been designed to be a research platform with a rich set of sensors and only basic movement abilities. The robot has been designed to have a height comparable with people so that it is perceived not like a toy by people interacting with it. We have designed the robot using the popular R2D2 droid from Star Wars saga as an inspiration, though our droid is slightly bigger than the original.

The first application for R2 (as we call it) will be to stroll around the Computer Science Department floor, helping visitors to find rooms. It will also play the role of bridge between the real world and the virtual world of Internet: we expect that it would rely on popular services to be able to find information at need.

It may not seem an ambitious task, though it involves several problems that should be faced by all robots in these kinds of environments: robust navigation with collision avoidance, route finding, vision (for instance face recognition), listening to the environment.

R2 sensors include:

- 24 Infrared Sensors distributed all around the body
- 6 Ultrasonic sensors on the head
- 6 Light sensors on the head
- 1 Compass
- 1 Camera (soon a stereoscopic vision system)
- 2 Microphones for stereo audio
- The state of the two PC's (both software and hardware)
- WiFi signals (802.11b)
- Blue Tooth signals
- Voltmeters (for checking batteries' state)
- Switches for sensing body posture

Actuators allow the head rotation, robot movement and shape shifting (like the droid in the movie it is able to alternate configuration between two and three wheels).

The navigation system is structured into three layers (in a Brooks' like architecture): a first reactive layer whose goal is to avoid collisions; when no collisions are to be avoided we rely on Compass to decide the direction to follow; we use WiFi signals from different access points to triangulate an approximated position of the robot, to be used to check if a desired location has been reached.

The ability of a robot to notice people is important in order to be believable (and thus capable of establishing a relation): the on-board camera uses the OpenCV facilities for face detection and face recognition, in order to be able to spoke to people.

Lack of actuators is not a significant restriction: the robot is equipped with WiFi, thus it can control devices through an Internet Connection. The research is still ongoing, though we have former results about our navigation strategy that encourage us to pursue the taken path.

Another important point is that when a robot is autonomous people would like that it is responsible for battery charge. Again this task is not trivial because the robot should be able to sense when the power supply is low and figure out where it can

connect to the power line. R2 has been designed to be able to switch between internal, battery-based, supply and external power. There is a board responsible for governing the power switch process, ensuring that batteries are charged appropriately while the PCs running the control program are supplied by the external power line. The process of connecting the robot to the external power requires a docking station designed to simplify the docking process.

5 Robotics4.NET initiative: a software body for robots

We have started the Robotics4.NET initiative [6, 7] with the goal of developing a software framework for programming robots sharing the environment with people. The framework has been designed to not depend from the particular structure of the robot: a set of software modules (called roblets) are responsible for defining a software abstraction for the body (they are different from drivers in many ways). Figure 1 shows the architecture of the framework: the roblets define the structure of the body and communicate with the bodymap, a sort of cerebral cortex used by the “brain” software as the body. In this architecture the body is modelled as a software entity rather than a set of drivers for actuators and sensors, leading to a pluggable architecture where modules can be reused on different robots (part of the software running on R2D2 has been developed on ER1).

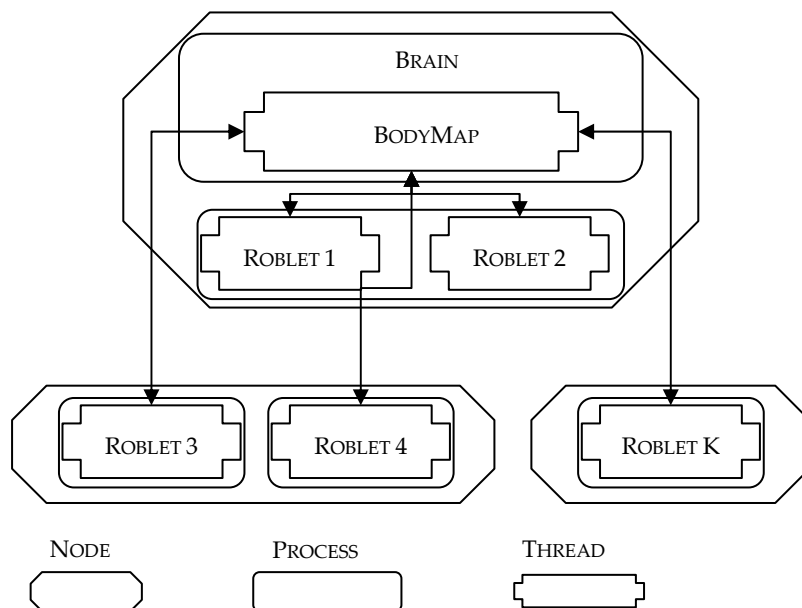


Figure 1: Architecture of Robotics4.NET

The framework includes several libraries for controlling several devices. Moreover it has been entirely developed in .NET because we rely on several facilities of the runtime for providing the communication infrastructure between roblet and bodymap.

Robotics4.NET also supports the compact framework version of the .NET runtime, available on small devices like PDAs and mobile phones. We have a demo of how to control a RoboSapien toy using a PocketPC running a Robotics4.NET based program.

We believe that the software will play a key role in the scenario of home robots: people will expect that these devices will interact and perhaps will exchange parts at need. A common software infrastructure can help this interaction allowing system developers to focus on the real aspects of the problem.

6 Conclusions

In this paper we have described the ongoing research activity at University of Pisa on robots. Our focus is on building a software infrastructure for programming robots situated in the same environment with people. We believe that in this context robustness is more important than correctness. In our experiments we have found that there is a common subset of abilities that, in our opinion, robots in this context should have. In our applications we have controlled robotic toys (that will be likely the first entering people houses), and a more complex architecture that we have realized at our University.

A first task for R2D2 will be to welcome people at our department, escorting them to desired rooms. Another application is monitoring of devices and Wireless signals in the building. We expect to use this as a case study for several relevant primitives for house robots.

Acknowledgments

This activity has been supported with grant by Microsoft Research. We wish to thank Marco Combetto for his invaluable help in our activity and for believing in our work.

Bibliography

1. Attardi, G., Cisternino, A., Colombo, D., *CIL + Metadata > Executable Program*, Journal of Object Technology, vol. 3, no. 2, Special issue: .NET: The Programmer's Perspective: ECOOP Workshop 2003, pp. 19-26, 2004. Available at http://www.jot.fm/issues/issue_2004_02/article2.
2. <http://www.robosapienonline.com/>
3. <http://www.robotics4.net/Software/RoboSapiens.aspx>
4. <http://www.evolution.com/>
5. <http://rotor.di.unipi.it/>
6. <http://www.robotics4.net/>
7. Cisternino, A., Colombo, D., Ennas, G., Picciaia, D., *Robotics4.NET: Software body for controlling robots*, to appear on IEE Proceeding Software.