LEARNING MOBILE ROBOTICS USING LEGO MINDSTORMS

Paulo Leitão, José Gonçalves, José Barbosa

Polytechnic Institute of Bragança, Department of Electrical Engineering, Quinta Santa Apolónia, Apartado 1134,P-5301-857 Bragança, Portugal
{pleitao, goncalves,jbarbosa}@ipb.pt

Abstract: The learning of some areas of knowledge requires the application of new learning methods in order to motivate and involve the students in the learning process. This paper describes an experience of the introduction of the flexible, modular and educational LEGO Mindstorms platform at high-degree studies to support the learning of mobile robotics topics.

Keywords: Robotics, Robot Programming, Computer Control, Education

1. INTRODUCTION

Statistical studies related to scholar success present some preoccupying indicators in the learning of some areas of knowledge, namely physics and mathematics. This observation deserves a deeply reflection about the current learning methods and the way to motivate and involve the students in the learning process.

The authors think that the current trend in learning programs, even at high-degree studies and research activities, is applying the concepts of 'learning by doing' and 'learning by enjoying', involving a strong effort in the integration of multidisciplinary skills and teams. In this context, the LEGO Mindstorms (Mindstorms, 2005) platform appears as a simple, flexible, attractive and educational way to achieve the referred challenge in several domains of knowledge.

LEGO Mindstorms has its roots in the pioneering work of Logo, a computer language and a philosophy of education, begun by Dr. Seymour Papert in the 1960s at the Artificial Intelligence Laboratory and continued at the MIT (Massachusetts Institute of Technology) Media Laboratory. Papert was perhaps the first to suggest that children should program computers, an absurdly radical idea at the time. This notion led to Papert's theory of learning called "constructionism" which suggests that: i) learning is an active process of building ideas which is unique and personal for each learner, and ii) this journey can be greatly facilitated when the learner builds things-in-the-world which become social and shared objects of reflection. Papert's Logo programming language, and later LEGO Mindstorms, inherit from this set of ideas.

The LEGO Mindstorms platform uses the basic concepts of LEGO to build mechanical models. Due to the intrinsic features exhibited by the LEGO Mindstorms platform, such as reusability, modularity, flexibility, and cost-effectiveness, the authors argue that the introduction of that platform in some Engineering curricula is useful, for example to improve the learning of several areas of knowledge, such as mobile robotics, computer programming, artificial intelligence, distributed systems and electronics.

This paper describes an example of the introduction of LEGO Mindstorms at high-degree studies. This platform were used at the Polytechnic Insti-

tute of Bragança in the Automation and Robotic class of the Informatics Engineering course, supporting the students to build autonomous mobile robots using a flexible platform.

The paper is organized as follows: initially, Section 2 introduces some concepts related to mobile robotics and innovative ways to learn it. Section 3 describes the LEGO Mindstorms platform and Section 4 makes an overview of how to elaborate robotic programs for the LEGO Mindstorms platform. Section 5 describes the experience toke place at Polytechnic Institute of Bragança with the introduction of LEGO Mindstorms to learn mobile robotics. At the last, Section 6 rounds up the paper with conclusions and presents the future work.

2. HOW TO LEARN MOBILE ROBOTICS

Mobile autonomous robotics is an interdisciplinary subject, combining and integrating different areas of knowledge, such as mathematics, physics, mechanics, electronics, control, computer programming, artificial vision and artificial intelligence. Indeed, mobile robotics make appeal to the integration of multi-disciplinary skills and teams.

The word *robot* is derived from a satiric theater play, called R.U.R. (Rossums Universal Robots), written by Karel Capek in 1921, who used it to designate *labour force*. In spite of several definitions, it is possible to say that a robot is an automatic machine that executes operations normally executed by humans. The word *Robotics* was introduced by Isaac Asimov in 1942 to designate the science that handle with the robots.

As definition, a mobile autonomous robot possesses the capability of mobility and executes its own decisions using the feedback from the environment.

In spite of being an emergent and interesting study field, the autonomous mobile robotics topic encompasses a significative set of complex tasks, such as the perception, localization, navigation, and path planning (Dudek and Jenkin, 2000).

The sensorial perception aims to answer to questions like where am I, and what is the environment where I am placed, gathering information to determine the location of the vehicle and the distance to obstacles. This component requires the implementation of a sensorial system using for example ultrasonic sensors, infrared sensors and artificial vision systems.

The navigation component aims to determine the path between a initial position to a target position, avoiding the occurrence of collisions. This component comprises three main sub-tasks: mapping and modelling the world, path planning and selection, and guidance. In certain situations, additional tasks are required, such as the cooperation with other autonomous entities, for example in soccer robotic applications.

The locomotion of the mobile robot is possible by using DC motors. Normally, gear boxes and encoders are coupled to the motors, allowing respectively to increase the torque and to determine the positioning of the robot.

The learning of mobile robotics requires the usage of new methodologies that apply the concepts of 'learning by doing' and 'learning by enjoying', allowing to motivate and involve the students in the learning process.

In this context, the LEGO Mindstorms platform appears as a simple, flexible, attractive, educational and suitable tool to increase the learning of mobile robotics topics. The main reasons to sustain this argue are that it is a tool that everybody is familiarized. The LEGO parts allows connectivity, eliminating the need of using screws or glue making the construction of mechanical models much more clean and easier. It is also an ecological tool because although the plastic is not easy to recycle, the parts are never made unusable, so they never become garbage, being used over the time.

At high-level studies, the difficulty to build quickly mechanic structures for the robot retracts the time and motivation to learn robotics topics. The possibility to build quickly robots with different configurations provided by the LEGO platform offers a good opportunity to learn this topic by doing and enjoying.

3. LEGO MINDSTORMS PLATFORM

LEGO, a well-known world trademark, is a set of plastic parts for the construction of mechanical models. The LEGO Mindstorms Robotic Invention System (RIS) platform uses the basic concepts of LEGO to build mechanical models, used by millions of children around the world, powered with technologies introduced by MIT Media Laboratory, allowing to learn by designing, inventing and experimenting computer-controlled systems, such as mobile robotics. The flexibility and modularity associated to the LEGO Mindstorms platform allows to create quickly different robot configurations, presenting a motivation for the people that is giving their first steps in the world of robotics (and specially in the world of mobile robotics), and assuming an important role in rapid prototyping (Reshko *et al.*, 2000).

The RCX is the programmable LEGO brick, acting as the brain behind Mindstorms, that trans-

forms mechanical models into robots and controls their actions. The RCX extends the child's construction kit, enabling students to build not only structures and mechanisms, but also behaviors. Figure 1 illustrates the RCX brick and the main components connected to it.

Fig. 1. Components of LEGO Mindstorms

The RCX has three inputs (numbered using numbers from 1 to 3) and three outputs (numbered using letters from A to C). This means that the RCX brick can gather information from the environment, via three sensors, and can power three motors. The common sensors used by this platform are the infrared sensors and the touch sensors. However, it is possible to build additional sensors to connect the RCX brick, such as ultrasonic and temperature sensors, exploring the potentialities of LEGO Minsdstorms (*RCX Internals*, 2005).

The platform provides two 9 volt DC motors equipped with a gear box. The gear box allows reducing the angular speed of the wheel and increasing the available torque.

One of the RCX Brick limitations is having only 3 input and 3 output ports. Using the infrared communication, two RCX bricks can communicate with each other, minimizing this limitation expanding ports to the double.

Being a world wide used tool, a set of significant information are available to experiment and develop robotic applications using the LEGO Mindstorms. In the web, the site (*Mindstorms in Education*, 2005) aims to act as a resource center for parents, educators and researchers who would like to use Mindstorms for educational purposes. Some books are also available, such as (Bagnall, 2002), (Laverde *et al.*, 2002) and (Ferrari *et al.*, 2002).

4. HOW TO BUILD A ROBOT PROGRAM

The programming of the RCX brick can be executed using the RCX Code, within the RIS programming environment, which is a graphical programming tool supplied by LEGO, as illustrated in the Figure 2.

Fig. 2. RIS Programming Environment

The RIS graphical programming language that consists in blocks that are arranged to program the brick. It comprises a set of *Big Blocks* that act like macros, i.e., contains several sub-blocks, each one performing one specific task. For example, there are already pre-defined blocks to move the robot forward for some amount of time, to *Turn left* and to *Turn right*.

It also comprises the so called *Small Blocks* that act like functions that can be used to control some features of the brick, like to control the power of the motors, the sound and the infrared communications. The RIS graphical language also allows to build new *Big Blocks*, i.e., it is possible to group one set of actions that will be used more than once on the robot program. The RIS programming language also contains some *Wait* and *Repeat* blocks, and a set of blocks to interact with the different available sensors.

As an example, the RIS programming environment is used to develop a robot program that follows a black line marked in the floor. The program, illustrated in the Figure 2, comprises three independent columns that addresses the three possible situations, that are, i) both sensors on the black line, ii) the left sensor outside the black line, and iii) the right sensor outside the black line.

The column on the left addresses the situation when both sensors are on the black line. This is made with a *Repeat Forever* cycle that makes the robot move forward. The other two columns are identical and have a *Repeat While* cycle that make the robot turn to the opposite side of the sensor that is outside of the black line.

As an example, if the left sensor (in this case the sensor connected to the RCX input 1), is outside the black line, the robot turns right while that sensor is outside the black line. This situation is shown in the Figure 3 that illustrates the set of blocks used to perform this action.


```
if(state==2){
  while(left_sensor<=lum){
    Motors.SpeedMotorA=3;
    Motors.SpeedMotorC=-3;
    left_sensor=(int)Sensor.S1.readRawValue();
  }
  state=1;
}
```

Fig. 3. Detail of the Blocks Related to one Sensor

Summarizing, the RIS programming environment allows the easy development of robot programs, essential for the kids handling. In spite of its advantages, mainly referred to the simplicity and intuitiveness, RCX Code presents some drawbacks that constitutes a barrier to the development of complex tasks. Several programming languages were developed addressing this challenge, such as the Not Quite C (NQC) and LeJOS (Ferrari *et al.*, 2002).

Both LeJOS and NQC are high-level programming languages that allows the student of university level to make some elaborated control programs although it doesn't use all the potential of the both languages. For example, the LeJOS programming language do not have Garbage Collector, which is positive because the processor is not going to waste time with a process that is called cyclically, without the intervention of the programmer.

As an example, the graphical program portion developed using the RIS programming environment can be coded using the Java programming language, as illustrated in the right side of the Figure 3.

The robot programs, developed using one of above programming languages, are downloaded to the RCX brick via infrared communication, using the infrared tower supplied by the LEGO Mindstorms platform.

5. EXPERIENCE IN THE INFORMATICS ENGINEERING COURSE

The introduction of LEGO Mindstorms at high-degree studies was exemplified at Polytechnic Institute of Bragança, by using it in the Automation and Robotic class of the Informatics Engineering course supporting the students to build autonomous mobile robots using a flexible platform.

5.1 Contextualization

The class was constituted by 10 students that were divided into 4 groups, each one having one LEGO Mindstorm Robotic Invention System 2.0 box.

The challenge addressed by the students was to build an autonomous mobile robot that be able to follow a black line drawn in the floor (see Figure 4.a), and also to find a green circle within a house (see Figure 4.b).

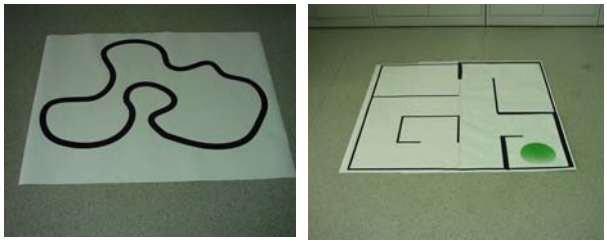

Fig. 4. Challenges to be Executed: a) Follow a Black Line and b) Find a Green Circle.

Figure 5 illustrates the several autonomous mobile robots developed by the students during the class, each one equipped with 2 motors and 2 infrared sensors.

Fig. 5. Robots built using LEGO Mindstorms

This shows an imaginative and creative aspect in building and designing systems, which has, for an engineer, so much importance as the technical aspects.

5.2 Robot Programming

In order to extract the maximum of potentialities from the RCX brick, all groups used the LeJOS programming language, mainly because this programming language has a significatively

faster API implementation than the alternative NQC programming language. For that purpose it was necessary to install the Java sdk1.4.2 and the LeJOS platforms in the PC that will support the development of the robot programs.

In order to prepare the RCX brick to accommodate Java programs it was also necessary to transfer the LeJOS firmware to the RCX brick, using the LEGO infrared tower connected to the USB port, through the following command:

```
c:\lejos > lejosfirmdl
```

Different approaches to the first addressed problem (i.e. to follow a black line) were presented by the several groups. An example is illustrated in the flowchart of the Figure 6.

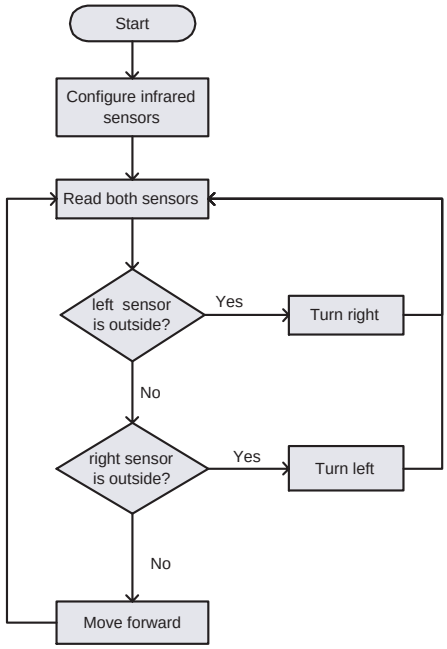

Fig. 6. Flowchart for the Navigation Algorithm (Following a Black Line)

Initially, the algorithm control comprises the configuration and activation of the sensors. Such configuration, for the sensor connected to the port 1, is obtained using the following Java code:

```
Sensor.S1.setTypeAndMode (3, 0x00);
Sensor.S1.activate();
```

After the sensors configuration, the algorithm enters in an infinite while cycle. The first action done in each cycle is to read the value of both infrared sensors. Having the knowledge of those values it is possible to decide the next move that the robot should execute. The code used to read the sensors is:

```
left=(int)Sensor.S1.readRawValue();
right=(int)Sensor.S3.readRawValue();
```

Then, it is necessary to take some actions: if one of the sensors is outside the black line the robot should turn to the other side and do this until that sensor is above the black line. This is done using the following Java code:

```
if(state==2){
  while(left<=lum){
    Motors.SpeedMotorA=3;
    Motors.SpeedMotorC=-3;
    left=(int)Sensor.S1.readRawValue();
  }
  state=1;
}
```

In the previous portion of code, the motor connected to the port A is moving forward and the motor connected to the port C is moving backwards. This code is used if the left sensor is outside the black line, allowing the robot to turn right. A similar portion of code can be used to implement the behaviour for the right sensor. The *lum* variable contains the value used to differentiate the white colour from the black colour.

If none of the sensors is outside the black line the robot must to move forward, i.e., both motors must be powered to go in the same direction and with the same power. This is done using the code below:

```
Motors.SpeedMotorA=3;
Motors.SpeedMotorC=3;
```

In the same way, different approaches were presented for the second problem (i.e. to find a green circle). An example is illustrated in the flowchart of the Figure 7.

In this case, besides the white and black colours it's also needed to find a target that has a different colour. If both sensors are in the ground, the robot must move forward. If one of the sensors sees a wall, the robot turns to the other side. The other possible case is if one of the sensors finds the target. The sensor that finds the target makes the robot to turn to that side to make both sensors stand above the target. When both sensors are above the target the robot beeps to indicate the target.

5.3 Results

The performance of developed robots and associated navigation systems were evaluated using different configurations for the black line tracks and for the house layouts. In the Figure 8 it is illustrated one of the developed robot following a black line in one of the evaluated configurations.

All the developed robots fulfilled the addressed tasks. However, due to the several black line configurations, some of them complex in order to evaluate the efficiency of developed control

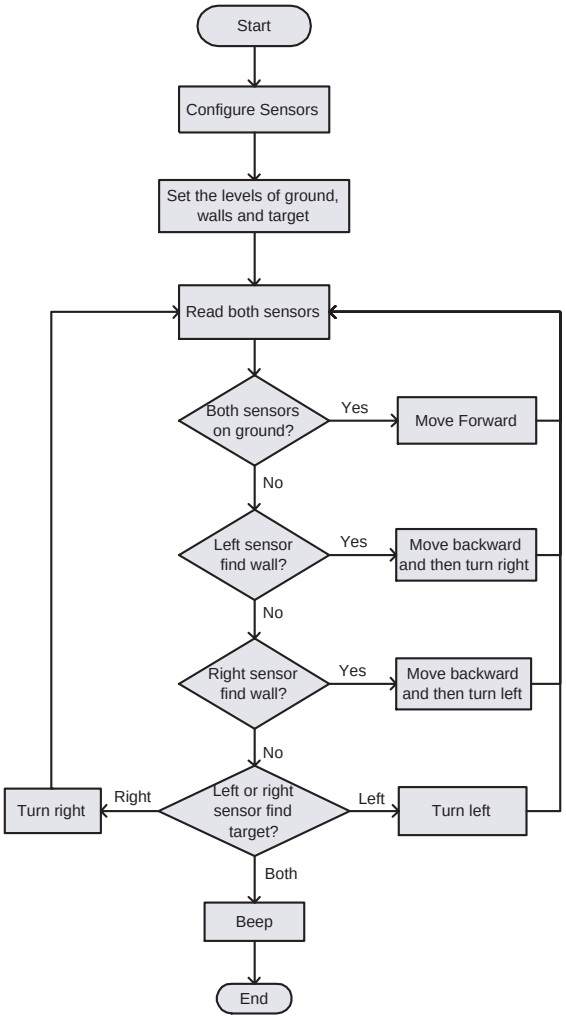

Fig. 7. Flowchart for the Navigation Algorithm (Find a Green Circle)

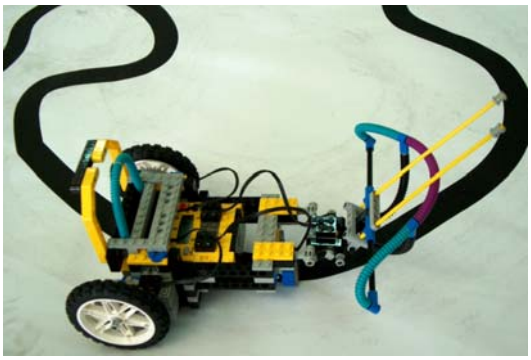

Fig. 8. A LEGO Mindstorms Robot Following a Black Line

algorithms, it was observed that each developed robot presented different performance according the configuration.

At the end of the class, an enquire was performed to the students in order to evaluate the results of the introduction of the LEGO Mindstorms. The main conclusion extracted from this questionnaire is that the introduction of LEGO Mindstorms platform was a remarkable experience, and an important factor of motivation to learn robotics and computer programming issues.

This enforces the idea that the LEGO Mindstorms is easy enough for kids, but powerful enough to support learning in high-degree schools and studies.

6. CONCLUSIONS AND FUTURE WORK

This paper describes an experience using the LEGO Mindstorms platform to support the learning of mobile robotics topics. Due to the possibility to build quickly robots with different configurations, modularity and connectivity features exhibited by this platform, the experience allowed to put in practice the concepts of "learning by doing" and "learning by enjoying".

During the class, each group built its own autonomous mobile robot and developed control algorithms to allow the robot to perform two different tasks: follow a black line and find a green circle. At the end of the class, the students considered that the introduction of LEGO Mindstorms platform was a remarkable experience and an important factor of motivation to learn mobile robotics topics.

In the future, we intend to expand the application of this platform to support the learning of other areas of knowledge, such as electronics, computation, pneumatics and artificial intelligence.

REFERENCES

Bagnall, B. (2002). *Core LEGO Mindstorms Programming: Unleash the Power of the Java Platform*. Syngress Publishing, Inc.

Dudek, G. and M. Jenkin (2000). *Computational Principles of Mobile Robotics*. Cambridge University Press.

Ferrari, M., G. Ferrari and R. Hempel (2002). *Building Robots with LEGO Mindstorms, The Ultimate Tool for Mindstorms Maniacs*. Syngress Publishing, Inc.

Laverde, D., G. Ferrari and J. Stuber (2002). *Programming LEGO Mindstorms with Java*. Syngress Publishing, Inc.

Mindstorms (2005). http://mindstorms.lego.com/eng/default.asp.

Mindstorms in Education (2005). http://viztel.com/mie/.

RCX Internals (2005). http://graphics.stanford.edu/∼kekoa/rcx/♯Intro.

Reshko, G., M. Mason and R. Nourbakhsh (2000). Rapid Prototyping of Small Robots. Technical report. Carnegie Mellon University.