6th International Conference on Applied Informatics
Eger, Hungary, January 27–31, 2004.

Learning serious knowledge while "playing"with robots

Zoltán Istenes

Department of Software Technology and Methodology,
Faculty of Informatics,
University of Eötvös Loránd
e-mail: istenes@inf.elte.hu

Abstract

This paper claims that serious knowledge can be learnt through building and programming robots.

Building and programming robots permit to confront and to deal with several simplified real world problems, give a large problem solving experience for the future and also facilitate the use and the combination of knowledge from different fields of informatics. Important skills can be acquired while having the feeling of just playing.

After the presentation of the used hardware elements and software possibilities, through three concrete projects, the line following, the game playing and the discovering robots projects, the paper summarises the educational benefits of building and programming robots.

Categories and Subject Descriptors: I.2.9 [Robotics], C.3 [Special-purpose and Application-based Systems]: Real-time and embedded systems

General Terms: Education

Key Words and Phrases: LEGO®, RCX

1. Introduction

Children are often seen playing with construction games, using small elements, building all kinds of objects, but it is a quite unusual to see adult people, serious university students still playing basically the same game [1, 2].

At the University of Eötvös Loránd, students can choose the course of "Robotics"where, using small construction and programmable elements, they have to realise different projects of building and programming robots.

From the point of view of a third party it may seem to be just a nice, unusual, colourful and may be a bit childish course, where both students and small car looking robots with flashing lights, run all around a table.

Are the students just playing and having fun? Does this course give any new and serious knowledge to the students? What kind of knowledge can they appropriate during these courses? How can this knowledge be connected to the other fields of informatics?

2. Presentation of the used material to build and program robots

2.1. Presentation of the hardware components

Most of the hardware components are used from the LEGO® Mindstorms™ Robotics Invention System 2.0 box [3, 4]. The central element of the robots is the RCX (Robotic Control X), which is a palm size box, containing a Hitachi H8/3292 16MHz microprocessor [5], 16Kbyte ROM and 32Kbyte RAM memory, an infra red connection port (IR port), an LCD display, buttons and batteries [6, 7]. Three sensors, such as light intensity, touch or rotation sensors, and three actuators, such as motors or lamps can be simply connected to one RCX.

An other important hardware component is the infra red connection tower which can be attached to a desktop computer. The communication, the controlling and the programming of the RCX is assured via this tower and the RCX IR port. The IR connection ports enable the communication amongst different RCXs too.

The RCX ROM memory contains a driver program, running when the RCX is powered up, enabling the RCX to download additional programs. The driver program can be extended by downloading a firmware to the RAM and the rest of the RAM can be used by the user programs.

The construction, the testing and the modification of the robots hardware is very simple, easy and quick with the LEGO® elements, allowing students to spend less time on the building itself and spend more time on the programming.

2.2. Presentation of the possibilities to program and control the robots

The RCX can be controlled in two ways.

In one way, the RCX receives commands via the IR port and executes them immediately. In this remote control case, a controlling device as for example a desktop computer or an other RCX, sends the commands to the RCX. These commands such as "start motor A forward"and "send sensor 2 value", are interpreted by the firmware and executed by the RCX microprocessor. The RCX is only used as a "simple input/output peripheral", the controlling program is not running on the RCX. In a concrete application, choosing this way to control the robot, the RCX sensor's values are sent to the controlling device via the IR port, then the controlling device does the necessary computation and finally sends the commands to the RCX via the IR port. In this case the IR port's communication speed is very limited, but on the other hand the controlling device's (for example a desktop computer) computational speed can be much higher than the RCX's.

In another way, the robot controlling program runs on the RCX autonomously. First the robot controlling program is created on a desktop computer and then downloaded through an infra red connection port into the RCX memory to the user program space and finally executed by the RCX processor without any further need of a desktop computer.

There exist several firmwares [8, 9] with different controlling and programming capabilities, having different program control structures, data types, communication and multitasking primitives, etc. The robot controlling program can be written in C [10, 11, 12], Java [13] or assembly-like sub-languages, or else [9], using different programming environments [14].

The flexible hardware and software background facilitate the construction of a wide variety of programmable robots.

3. Examples of programming and using robots

At the University of Eötvös Loránd, at the courses of "Robotics", students work on their projects in teams of three, for one or two weeks, then they present it during the courses to the other students and to the teacher. Then the project is fully discussed by the class. The first common project is generally the simple "line following"project, then the teams can choose or find out a more complex project.

3.1. Presentation of the "line following"project

The objective of the "line following"project is to build and to program a mobile robot that follows a line as fast as possible.

Different lines like a thin "guiding line"or a large one like a road, being continuous or with gaps, straight or with sharp turns and junctions, all need different kinds of robots and programs. A mobile robot using 1, 2 or 3 light intensity sensors can detect the line, since it has a different colour than the background. Several line following algorithms can be found and implemented.

In the easiest version the robot follows a large, continuous and straight line.

This can be complicated by changing the parameters of the line, making it thin, with gaps, with sharp turns or junctions, all these changes require more complex control algorithms.

In a more complicated version, faster the robot moves, more the real time constraints have to be considered. For example, a fast moving robot may not have enough time to read the sensors, process the data and control the robot to stay on the line.

The robot controlling program also has to deal with situations where the line is lost, so it has to be robust, fault tolerant and "error recovering". One recovering solution can be to stop the robot and drive it back to the position where the line was last seen. In an other solution, the robot doesn't stop immediately when the line is lost, but tries to continue to go towards the "estimated line direction", the program memorises on which side the line was last seen and tries to drive the robot towards that direction, "hoping"that the line will be found. If the line is not found in a certain amount of time, it switches to an "active search"mode where the robot tries to find the line by checking several directions or going in bigger and bigger circles, etc.

The robot controlling program can learn from previous situations and adapt the robot to the environment, for example by turning sharply in the sharp turns or by accelerating in the long straight lines, or by adapting the robot to the changing lighting conditions, etc.

The project can be extended, for example by putting an obstacle on the line, the robot has to go around this obstacle and find the line on the other side.

3.2. Educational benefits of the "line following"project

In the line following project students learn from an already known language (C, Java) to use a very similar, but still another different programming language (NQC, BricxCC, LegOS, BrickOS^{TM}[10, 11, 12, 8], LeJOS [13]).

They learn how to cope with the limited data types of the firmware, how to represent the robot controlling, by designing state machines and by converting these state machines into a program. They also discover how to deal with real time constraints, how to design robust fault tolerant control algorithms, and how to use error "recovery"techniques.

Simple reinforced learning, adaptation techniques and even fuzzy control methods can be experimented.

3.3. Presentation of the simple game playing projects

The objective of the different "game playing"projects is to make the RCXs to communicate with each-other via their infra red communication ports.

Different games such as number guessing, tik-tak-toe or battleship, etc, all need different kinds of communication methods. The original firmware communication possibilities are very simple, the RCX can send or receive only one byte via the IR connection, without any acknowledgment of the reception. When several RCXs communicate, there is no indication from which RCX the message came and no indication about which RCX received the message.

In the battleship game the "server"RCX draws the battleship table, communicates with the two player "client"RCXs by asking them their "shots", and by sending them back their "hits"or their "misses".

In the number guessing game three RCXs with an identical program play with each-other. The RCXs elect one "game master"amongst themselves, which thinks about a number that the two other RCXs have to guess and find out. In the next round another game master is elected.

The game logic itself had to be realised as well.

3.4. Educational benefits of the simple game playing projects

In the simple game playing project, students learn how to design and implement synchronisation mechanisms, communication protocols, client server architectures and distributed algorithms.

They also learn how to realise the game logics by using knowledge of artificial intelligence.

3.5. Presentation of the experiments with "discovering robots"

The objective of the "discovering robots"project is to make the mobile robot go to a target object, using an overhead camera's images.

Two programs have to be written, one running on the desktop computer and another one running on the robot RCX.

The program running on the desktop computer captures the overhead camera images, recognises the robot, the target object and the obstacles on the camera images, then it uses artificial intelligence techniques to find a path to the target object and finally sends the guiding information to the robot.

The program running on the robot RCX receives and executes the guiding instructions from the desktop computer via the infra red communication port. The communication between the desktop computer and the RCX is not continuous, the robot tries to discover its environment and only asks for "guiding"from the desktop computer when it runs into an obstacle.

3.6. Educational benefits of the experiments with "discovering robots"

In the "discovering robots"project students learn how to use knowledge from three different fields of informatics such as computer graphics, artificial intelligence and communication.

They learn how to use image processing, pattern recognition algorithms, how to map the recognised objects into an internal representation, how to find a path to the target object avoiding the obstacles, how to send this path as guiding instructions and finally how to communicate with the robot when the communication is not reliable.

4. Conclusion, educational benefits

Building and programming a robot can be very easy and simple in the beginning, giving a lot of self confidence, success and joy, motivating the students to go further on and to realise gradually, even more and more complex and difficult projects.

Robotics can be seen as a reduced test environment of the real world, where problems arise faster since the hardware and software limits are reached sooner. This allows the students to confront problems as they might find, later on, in the real world, but in a smaller and easier way, facilitating them to find out and apply solutions, and giving them a large experience in solving problems in the future.

Building and programming a robot require knowledge from several different fields of informatics, such as programming languages and techniques, communication and networks, artificial intelligence and agents, pattern recognition, etc. Students at their different courses learn about several kinds of methods, which they can after tangibly and easily try, use and test on programming the robot RCX. The results are immediate and evident just such as problems or failures.

Students enjoy this kind of thinking, programming and learning. They work in a team, they have to use their creativity, both in planning a project and realising it, and at the end they present a tangible, working result to the others.

Finally student use and learn serious knowledge while having the feeling of "just playing with robots".

References

[1] Klassner, F., and Anderson, S., *Lego MindStorms: Not Just for K-12 Anymore*, IEEE Robotics and Automation Magazine, in Press (January 2003).
`http://www.csc.vill.edu/~klassner/pubs/klassner-mindstorms-IEEE-RA02.pdf`

[2] Randall D. Beer, Hillel J. Chiel, and Richard F. Drushel. *Using autonomous robotics to teach science and engineering.* Communications of the ACM, 42(6):85–92, 1999.
`http://citeseer.nj.nec.com/beer99using.html`

[3] *Lego Mindstorms Home*
`http://mindstorms.lego.com`

[4] Rob Salgado: *About the Lego Mindstorms The Robotics Invention System*
`http://www.pulsar.org/archive/stormwatch/AboutLegoMindstorms.html`

[5] *H8/3292*
`http://www.renesas.com/eng/products/mpumcu/8bit/h8300/3297/`

[6] Kekoa Proudfoot: *RCX Internals*
`http://graphics.stanford.edu/~kekoa/rcx/`

[7] Russell Nelson: LEGO® Mindstorm™ Internals
`http://www.crynwr.com/lego-robotics/`

[8] *BrickOS™*
`http://brickos.sourceforge.net/`

[9] Ralph Hempel: *pbForth*
`http://www.hempeldesigngroup.com/lego/pbFORTH`

[10] *Not Quite C*
`http://http://bricxcc.sourceforge.net/nqc/`

[11] *Bricx Command Center*
`http://http://bricxcc.sourceforge.net/`

[12] Markus L. Noga: *LegOS*
`http://www.noga.de/legOS/`

[13] *LeJOS*
`http://lejos.sourceforge.net/`

[14] LEGO® Mindstorm™ Resources
`http://mindstorms.lego.com/eng/community/resources/default.asp`

Postal address

Zoltán Istenes
Eötvös Loránd Tudományegyetem,
Informatikai Kar,
1117 Budapest,
Pázmány Péter sétány 1/c.
Hungary