Teacher Education on Robotics-Enhanced Constructivist Pedagogical Methods

Editor: Dimitris Alimisis

Published 2009 by School of Pedagogical and Technological Education (ASPETE)
ISBN 978-960-6749-49-0

Teacher Education on Robotics-Enhanced Constructivist Pedagogical Methods

Editor: Dimitris Alimisis
Published 2009 by School of Pedagogical and Technological Education (ASPETE)
ISBN 978-960-6749-49-0
http://www.terecop.eu/

Prologue

It is well known that nowadays we live in a rapidly changing world. The decade of ‘80s was marked by the transition from the “industrial society” to the “information society”, while in the decade of ‘90s we began to talk for the transition to the “knowledge society”. Nowadays we talk for the “creative society” (Resnick 2008), which requires from people to develop the ability to think and act creatively in problem solving everyday situations.

Under the effect that those changes had on education, a series of new educational tools, based mainly on the use of personal computers, have been invented last two decades to educate the student generations of the “knowledge” and “creative” society. Educational robotics is one from these new innovative tools that has attracted the interest of educational community from kindergarten to universities during last few years.

Educational robotics is introduced as a powerful, flexible teaching/learning tool stimulating learners to control the behavior of tangible models using specific programming languages (graphical or textual) and involving them actively in authentic problem-solving activities. This is the filed where the European project “Teacher Education on Robotics-Enhanced Constructivist Pedagogical Methods - TERECoP” was activated during the years 2006-2009 with the participation of 8 European educational institutions from 6 European countries (www.terecop.eu). The TERECoP Project aimed at the development of a design and implementation framework for activities advisable mainly for secondary school education related to programmable robotic constructions and based on learning methodologies inspired from constructivism and constructionism theory.

Believing that the role of teacher is crucial for the successful introduction of robotics in classrooms, the TERECoP Project focused on the training of prospective and in-service teachers in the use of robotics technologies (LegoMindstorms Education NXT) through courses implemented in each of the six participating countries, the evaluation of the training courses and the dissemination of the educational results at a European level. Finally the TERECoP project aspired to develop a community of practice between researchers, teacher trainers and teachers that can facilitate and sustain teachers’ professional development in the use of robotic tools in classrooms.

This book is a collective product coming from the cooperative and shared work done by the TERECoP partnership during the 3 years of the project. The book is addressed mainly to teachers at all school levels who are interested in introducing robotics in Science, Technology, Mathematics and Informatics classes or in the frame of interdisciplinary learning activities. In addition to that, the book is addressed to trainers of teachers offering them a constructivist model for training teachers in educational robotics.

The content of the book addresses a wide range of both theoretical and practical aspects of educational robotics.

Chapter 1 (Constructionism and robotics in education) presents and discusses some critical pedagogic aspects behind the educational use of robotics with respect to the potential of control technology to generate costructionist learning processes in the light of the constructionism theory. A review of research literature in the field of educational robotics and the evolution of Robotic technologies (from “floor Logo turtles” to Lego Mindstorms) are presented and commented.

Chapter 2 (robotics as learning object) begins with a Piagetian and Vygostkian methodological perspective for designing constructivist didactic situations to study robotics as learning object. The chapter continues with some good reasons to prefer the Lego Mindstorms NXT system among others, then offers practical knowledge for using the hardware and the software of the Lego Mindstorms NXT system with practical examples for constructing robotic vehicles and programming them to move in straight line, to turn right and left, to communicate with their environment and to acquire and log data through sensors.

Chapter 3 (robotics as learning tool) presents a constructivist methodology for designing teaching and learning activities where robotics is used as learning tool and exemplifies it with a representative example: the bus route project.

Chapter 4 (Teacher Training Course in Introducing Robotics in the Curriculum - The TERECoP Project Proposal) presents in details the TERECoP methodology for training teachers in educational robotics. The chapter includes training curriculum, detailed description of each module, training activities, tools and materials and tools for the evaluation of the training course. The chapter finishes with evaluation results, experiences and recommendations from training courses implemented in Athens and in Prague.

Chapter 5 offers some additional exemplary projects and examples of learning activities with robotics that teachers can use in their class adapting them both to the proposed constructivist training methodology and to their students’ cognitive level taking into account that some of them demand from students a sound background in mathematics.

Chapter 6 offers useful resources for teachers including books, papers (in journals and proceedings of conferences) and web resources.

Appendix 1 offers some basic technical knowledge for the LegoMindstorms NXT system (hardware and software) and **Appendix 2** a textual description language which is equivalent to the NXT-G graphical language and is used just for a convenient detailed description of some complex NXT graphical programs appeared in chapters 2 and 5.

Finally, this book aspires to bring closer researchers, academic and school staff working in the field of educational robotics and to contribute to the further development of the dialogue especially under the light of constructionism theory, not only within the TERECoP project partnership but within the broader European and international community of educational robotics. This dialogue will hopefully continue next years and the TERECoP partnership is willing to undertake or participate in new relevant initiatives in the future.

References

Resnick, M. (2008), ‘Sowing the Seeds for a More Creative Society’. | Learning & Leading with Technology December/January 2007–08, International Society for Technology in Education, 18-22.

Dimitris Alimisis, ASPETE, Greece

Patras, 2009

Chapter 1

Constructionism and robotics in education

Authors: Dimitris Alimisis, Chronis Kynigos

1.1. Introduction

Over the last few years, interest in educational utilization of robotics has increased and several attempts have been made world-wide to introduce robotics in school education from Kindergarten to high secondary school, mostly in science and technology subjects. Nowadays, robotics is considered as a flexible medium for learning, offering opportunities for design and construction against short time and small funds. The newest version of educational robotic technologies, that is the programmable bricks, enable students to control the behavior of a tangible model by means of a virtual environment and make possible new types of science experiments, in which children investigate everyday phenomena in their lives (both in and out of the classroom) (Resnick et al, 1996).

However, the successful introduction of an educational innovation in school settings is not just a matter of access to new technologies. Technology alone cannot affect students' minds and cannot act directly on learning. Appropriate educational philosophy, curriculum and learning environment are some of the important factors leading any educational innovation to success. In view of the above, before teachers and educators at all levels rush to exploit robotics in education, appropriate teaching methods need to be formulated and incorporated in the school curricula, given that most schools and teachers lack not only experience and resources, but, also, in most cases, they have to operate under a directive school curriculum that does not favor educational innovation. As Martaric points out, although robotics seems to be an excellent tool for teaching and learning and a compelling topic for students of all ages, the pedagogy of teaching robotics (we would add the pedagogy of teaching *with* robotics as well) is still in its infancy (Mataric, 2004)

1.2 Controlling and constructing robots as a constructionist environment

Construction and control were the first powerful ideas on the use of computational media for learning (Papert, 1980). With respect to digital media, this idea involved the transition from black-box software to the design of transparent (white-box) digital artifacts where users could construct and deconstruct objects and relations and have a deep structural access to the artifacts themselves (diSessa, 2000, Resnick et al, 2000). It also involved the idea of distributed control where multiple users worked with the same digital artifact either in presence or remotely from different computer screens so that they would express their ideas in collectives rather than work individually (Mor et al, 2006).

However, the existence of such media did not bring about the envisaged radical changes in learning environments based on their use (Papert, 2002). Students fell onto 'plateaus', unable to progress beyond a certain point and found that they could not construct something very interesting when starting from scratch every time. To address this problem, black-and-white-box design perspectives provided users with generic black box artifacts which they could then use as building blocks for their constructions with exploratory digital media (Kynigos, 2004).

In the use of robotics, we saw a parallel transition from black box situations of pre-programmed pre-fabricated robots, aimed for the workplace, to white box designs, where children can construct and program robots from scratch. However, there has been little or no attention given to distributed control and black-and-white-box solutions, where students can start from something complex and interesting and then move on to learning by constructing robots and programs to control them.

So, what kinds of learning can be nurtured in learning environments based on the construction, programming and control of robots? What meanings and concepts can be understood in such environments? Do they afford added value to the fostering of creative thinking?

The main learning theory, which has been perceived as useful at addressing the questions, has been that of a special kind of constructivism termed 'constructionism' by Papert and his group at the Media Lab (Kafai and Resnick, 1996). Constructivism originated from Piaget and perceives learning as the generation of meanings from individuals as they eternally strive to bring some cohesion to the ways in which they see the world (Fosnot, 1997; Brooks and Brooks, 1993). Tangible concrete experiences with the physical and social environment are used to create generalizations, discriminate invariants and construct abstractions.

Constructionism can be seen as a special case of learning in situations where we make or tinker with an object or an entity. It was seen by Papert as one of the ways in which thinking can be manifested, made public. Constructing was seen as an emergent activity, where a lot of back and forth went on, where design is part of the process of building rather than a pre-requisite and where building involves de-construction and re-construction rather than just construction (Kynigos, 1995). In coining the term, Papert wanted to convey a slightly differing perception of learning than Piaget, i.e. that humans do not necessarily strive for cohesion, but are by nature engaged in questioning their view of the world.

Constructionism was elaborated in the early eighties at a time when individualistic cognitive theories were at the forefront and was thus associated with an individualistic perception of learning. However, notions of collaborating and communicating during constructivist activity were firstly articulated as far back as the mid eighties (Rogoff and Lave, 1984) and have since become more and more pertinent as digital technologies have made it possible for more than one students to have access to the same construction at the same time (Mor et al, 2006). This has not however happened yet with mechanical technologies and robotics.

In any case, these perceptions of learning seem to fit very well with the activities of constructing robots and programs to control them. The robotics industry aims at humans using pre-programmed pre-fabricated robots to do arduous, repetitive, mundane, fast, precise, dangerous or physically impossible things from them. The ways in which the robots are made and programmed is a black box for their users. It is the same paradigm with which many technologies are constructed from hardware to software and digital tools. It is also compatible with the traditional educational paradigm of the teacher or the curriculum book revealing and explaining ready-made, ratified and, thus, unquestioned information.

In the framework of progressive and contemporary educational paradigms, construction and programming of robots have been made transparent so that individuals can engage in building and in programming robots themselves. Two main technologies have been so far designed and built for students to engage in robotics, the Lego-Mindstorms and the Pico-Crickets kits from the Media Lab at MIT (Resnick et al, 1996; Resnick et al, 2006). This white-box metaphor for construction and programming has generated a lot of creative thinking and involvement in learners, mainly in informal educational settings.

However, as in the case of digital media, there seems to be a plateau which learners reach with respect to what kind of robots they make and what they can program them to do. It quickly becomes very difficult for anyone to construct a technically robust and interesting robot and to program it to do complicated and interesting things. This was noticed some time ago, as in the case of Pico-crickets, where there was an expansion of the kinds of sensors and the kinds of constructions students could make (Martin et al, 2000) in order to enhance, for instance, the interest of female students.

An important part of learning with robots, apart from constructing and programming them, is controlling them or their environment in play. This has been rather under-exploited from an educational point of view precisely because of the white-box metaphor of starting from scratch with robotics. Controlling robots, however, can provide an avenue for black-and-white-box perspectives, where students can have distributed control of specific robots amongst others. This is seen as part of a complex learning environment likewise embedding the construction of robots and programs to control them as usually, but different in that there is also emphasis on interesting learning activity with robot control.

We consider robot control as an integral part of constructionism. We suggest that robot control can be perceived as an integral part of constructivist engagement with robotics and that given devices and setups, where control is designed to be interesting, students can learn from the kinds of feedback they get from their activities and intentions to control the robots or their environment and from the kinds of representations available to them for control.

Robotics is an integral part of control technology. The ways~~,~~ in which humans control machines, the semantics of the interfaces through which they control them and the discrimination of what is what they control in a certain machine behavior are becoming more and more pertinent for people to understand. The number and variety of automated machines that we control in our everyday lives is increasing continually and rapidly. Think of automatic doors, alarms set by motion detectors, lights put on by clapping. We interact with them all the time but have little idea of how they work. On the other hand, these are devices designed for our everyday lives, the workplace, the home, the public places, such as airports etc.

Consider devices set up for humans to learn things as they control them in order to do something interesting. For instance, the ways in which robots respond to changes in the environment and those changes to which they respond are very important concepts. Discriminating the kinds of things we can control robots to do and, by consequence, gaining insight into the way they are programmed in situations which are more complex than those in which they can be constructed by typical construction kits, has also been overlooked. The means by which we can control robots and the semantics of the devices we use to control them can operate as mechanisms through which we express our thinking, as expressive media. We do not need to wait for learners to build their own programmed robots in order to address these issues.

1.3 Robotic technologies: from floor Logo turtles to Lego Mindstorms

Research in the field of educational robotics has for years placed emphasis on the interplay between the invention of new technologies and the development of innovative ways of learning: new pedagogical ideas can lead to new technologies, and vice-versa (Martin et al. 2000). Since the late 1960's, research has been developed for robotic construction kits for children focusing on the invention of construction kits and programming tools that children will find easy to understand and control, thus becoming active participants in their learning and creators of their own technological artefacts instead of being just users of devices that others have made for them (Martin et al. 2000).

Early work, led by Seymour Papert, included the development of the Logo programming language (Papert, 1980). A popular use of Logo involved a "floor turtle," a simple mechanical robot connected to a computer by a log cord. With pens mounted in their bodies, floor turtles made drawings on paper, commanded by Logo programs. In the late 1970's, influenced by the increasing production of personal computers, the focus was shifted to screen turtles, which were found to be faster and more accurate than floor ones, while offering opportunities for children to investigate and solve more complex mathematical problems. In the 1980's, Papert's vision of computing, in which children explore ideas by constructing their own computer programs (Papert, 1980), came into being as the first microcomputers entered schools. Many of these activities involved, as a matter of fact, robotic design activities before a general-purpose robotic construction kit for children was made available.

In the mid-1980's, the LEGO/Logo technology, the first true robotic construction kit ever made available widely, appeared, combining the popular LEGO construction kit with the Logo programming language. Lego/Logo integrated two different types of design activities (Resnick & Ocko, 1991; Resnick, 1993). Children start by building machines out of LEGO pieces, using the traditional LEGO building bricks and newer pieces like gears, motors, and sensors as well. Whereas traditional construction kits enable children to construct structures and mechanisms, LEGO/Logo goes further by enabling children to construct behaviours for their artefacts connecting their LEGO constructions to a computer and writing computer programs with a version of Logo (Resnick , 1998).

LEGO/Logo might be seen like a return to the past, since it brings the turtle from the screen back to the real world. But LEGO/Logo compared with the early Logo floor turtles offer some key advantages: students can use LEGO/Logo not as ready-made mechanical turtles but they have to build their own constructions before programming them; in addition to that, children can use LEGO/Logo to build and program, not only turtles, but a wide variety of creative machines.

A serious problem encountered with the LEGO/Logo technology was the nuisance (not only in technical but also in conceptual terms) caused by the wires connecting the robot to a computer, which made it difficult for children to create autonomous and mobile robots. Programmable LEGO Bricks, which appeared in late 1980's, offered a solution to that problem since they run without wires providing in this way autonomous function to children's mechanical constructions. Children can build Programmable Bricks directly into their LEGO constructions, embedding accordingly computation directly into their constructions. Programmable LEGO Bricks expanded significantly the design and learning possibilities for children in 1990's (Martin, 1996; Resnick et al, 1996).

These first generations of robotic technologies served as the foundation for the development of the LEGO Mindstorms kits (http://www.legoeducation.com), a line of Lego sets combining programmable bricks with electric motors, sensors, Lego bricks, and Lego Technic pieces (a line of Lego interconnecting plastic rods and parts, such as gears, axles, and beams). Lego Mindstorms, named after Papert's *Mindstorms: Children, Computers, and Powerful Ideas* (Papert, 1980), are based on research and ideas from the Lifelong Kindergarten group at the MIT Media Lab (Resnick, 1998) and are already being used world-wide in both elementary and secondary education as well as in higher education.

The LEGO RCX Brick, the first retail version of Lego Mindstorms released in 1998 and marketed commercially as the Robotics Invention System (RIS), included motor outputs, sensor inputs, and an LCD screen. The educational version of the product, called *Lego Mindstorms for Schools*, came with ROBOLAB, a graphical user interface-based programming software developed at Tufts University (http://www.ceeo.tufts.edu) using the *National Instruments Lab VIEW* as an engine. The current version (Lego Mindstorms NXT) was released in 2006 and comes with servo-motors, new sensors and the NXT-G iconic programming software but can also be supported by a variety of other programming languages (such as NXC, NBC, leJOS NXJ, and RobotC).

Crickets are another robotic technology, developed in parallel with Lego Mindstorms, aimed at enabling children to learn important math, science, and engineering ideas through the creation of musical sculptures, interactive jewelry, dancing creatures, and other artistic inventions (http://www.picocricket.com/). Crickets have also been intended to engage children in new ways of learning in connection with their interests and passions, and to provide a deeper and more concrete understanding of scientific ideas and a richer sense of the interplay between science and technology (Resnick, 1998). A plurality of Cricket designs has been developed ("Display Cricket", "MIDI Cricket", "Science Cricket", "Cricket Bus system") that provide true analog-to-digital converters on the sensor inputs allowing the use of a greater variety of sensor devices, all of which can communicate with a standard cricket design.

The design of Crickets was heavily influenced by the Beyond Black Boxes (BBB) project, a science-education initiative (Resnick et al, 2000) which provided a theoretical framework and a collection of project ideas for a constructionist approach to science education. Crickets are aimed, among other goals, at enabling children (and educators) to design their own scientific instruments for investigations which they personally find meaningful. Through designing their own instruments, children are expected to gain a deeper appreciation and understanding of many scientific concepts (Martin et al, 2000)

There have also been interesting explorations with other "digital manipulatives" (Resnick, 1998), where computation is added to traditional children's toys embedding either a Cricket inside of a ball (Bitballs Project) or built-in microprocessor and LED (Digital Beads Project) or built-in electronics and infrared communication (Thinking Tags Project). All theses projects are aimed at engaging children in new ways of learning in connection with their own interests and passions: BitBalls can be used mainly in scientific investigations, Digital Beads to engage children in creating dynamic patterns and Thinking Tags to experiment with people's behaviour at social gatherings (Resnick, 1998).

1.4 Robotics in School Settings

Robotics projects and activities in school settings might be classified in two separate categories, according to the role that robotics play in the learning process:

- Robotics as *learning object*: This first category includes educational activities where robotics is being studied as a subject on its own. It includes educational activities aimed at configuring a learning environment that will actively involve learners in the solution of authentic problems focusing on Robotics-related subjects, such as robot construction, robot programming and artificial intelligence.

- Robotics as *learning tool*: In the frame of this second category, robotics is proposed as a tool for teaching and learning other school subjects at different school levels. Robotics as learning tool is usually seen as an interdisciplinary, project-based learning activity drawing mostly on Science, Maths, Informatics and Technology and offering major new benefits to education in general at all levels.

However, this classification is not always easy and clear. Even in the cases, when robotics is introduced as an autonomous learning object, it covers multiple educational aspects and serves objectives beyond those stated in the relevant curriculum extended to the development of problem-solving skills, creativity, critical thinking, collaborative skills etc. In the process of designing and programming robots, students learn important engineering, math, and computer science concepts (Druin and Hendler, 2000, Arlegui et al, 2008a). Robotics can enhance learners' research attitudes, allow learners to make assumptions, carry out experiments and develop their abstracting skills. So, learning constructed through robotics (seen as learning object) is also valuable for other cognitive areas belonging to the broader spectrum of the school subjects.

Over the last few years, several educational projects and initiatives have been developed in the field involving universities, schools or other educational and research institutions. A typical sample of them is presented shortly in the following lines just to offer a sense of the pluralism of thematic areas, educational objectives, learning approaches, topics and diverse audiences involved in past and current applications of robotics in the broader school settings.

The *Lifelong Kindergarten* group at the MIT Media Lab (Resnick, 1998 and 2008) has developed several robotics projects extending from the exploration of the fundamentals of mechanical motion (*Learning About Motion*) to a suite of tools and activities to introduce artists into robotic/electronic media (*Robotic Art Studio)* and to *Learning Engineering by Designing Robots (*for a full list of projects see http://llk.media.mit.edu/projects.php)

Fiorini et al. (2008) describe the efforts undertaken by a small community of teachers concerned with boosting science education in the school district of Verona (Italy) by promoting constructivism with the help of various configurations of robotic devices. These efforts have been going on for the last eight years, slowly gaining momentum and impact. They emphasise that the most striking difficulties have been encountered with the educational environment rather than with students themselves.

The network *Robot@Scuola* of Italian Schools works to gather into a unique national network schools from Primary to Secondary Professional and Vocational education, which use robotics in their educational processes (http://www.scuoladirobotica.it/progettieng.htm). *The Roberta-Goes-EU* project (http://www.iais.fraunhofer.de/3845.html) aspires to encourage young people, and especially girls, through Robotics to take up engineering studies, providing training courses and comprehensive teaching materials to teachers and others who wish to increase students' enthusiasm for technical professions.

The PIONEER (PIedmOnt NEt for Educational Robotics) is an Italian School-Net for the Educational use of robotics in school classes. Its goal is to promote Papert's constructionism in a cooperative environment setting up a model of minirobot programming experiences that can support the standard curricula for school years K-12 (De Michele et al, 2008).

Bers and Urrea (2000) in the framework of a research program at the MIT Media Laboratory, called Con-science, attempted to integrate learning about technology and values in a hands-on way, by involving families, as well as teachers, in the design and programming of robotic creations that represent their most cherished values.

Kärnä-Lin et al (2006) note that although robotics is used worldwide in education as a learning tool, surprisingly it happens only rarely in special education. Through qualitative action research they have identified various advantages that educational robotics can bring into learning in the field of special education: the robotic technologies make it possible for students to practice and learn many necessary skills, such as collaboration, cognitive skills, self-confidence, perception, and spatial understanding.

Dias et al. (2005) presented the challenges and benefits of three higher education initiatives in Sri Lanka, Ghana, and the USA that focused on innovating and implementing robotic technologies for developing communities, examined the potential intersections of robotic technologies with education and sustainable development and the factors that contributed to the success of such educational initiatives designed specially for developing communities.

Mitnik et al. (2009) describe a robotics-based educational project and compare it with a similar computer-simulated activity. The project was aimed at developing graph construction and graph interpretation skills and at reinforcing learning of kinematics concepts. The activity was carried out by means of a set of handhelds and a robot wirelessly interconnected. Results showed that students through the robotic activity achieved a significant increase in their graph interpreting skills that proved to be almost twice as effective as compared with the computer-simulated activity. Moreover, the robotic activity proved to be highly motivating for the students and fostered collaboration among them.

The Science, Engineering, NASA Site Of Remote Sensing (SENSORS) project (Portsmore et al. 2004) was intended to help bring remote sensing and tele-robotics to upper elementary and middle school audiences. Via the web, users remotely control LEGO RCX - based rovers by submitting programs that instruct the creation to complete a or collect data.

Other research efforts have focused on the integration of Robotics in Early Childhood Education developing attractive activities and effective practice for learning with digital technologies at preschool age (Bers et al,. 2002; Pekarova, 2008), while others focus on technical and vocational school students engaging them in designing, building and programming a robotic device that allowed them to explore phenomena of mechanics like the gear-aided transmission of motion (Alimisis et al, 2005) or the gear function and mechanical advantage (Chambers and Carbonaro, 2008). Carbonaro et al. (2004) describe a project-based learning environment in which various robotic construction tasks based on LEGO Mindstorms have been undertaken by middle-school students and highlight some sample products of their work.

1.5 Educational Robotics beyond School Settings: Competitions and other Events

In addition to the activities that take place in school settings, many other robotic events run in informal education contexts, structured as competitions or exhibitions. Each year, several robotics-related associations announce challenges with certain rules, and thousands of teams of young (and older!) people compete in national and international events. The mission of the competitions is usually to engage young people in exciting mentor-based training that builds science, engineering and technology skills, inspire innovation, and foster self-confidence and communication skills. Robotics contests and the relevant project work appear as a very suitable platform to support team-based learning, which is often undervalued in the current school systems (Petrovič and Balogh 2008).

Robotics competitions are growing rapidly in size and popularity and have proven to be very motivating for young people (Sklar et al, 2003). For example, the FIRST

(**F**or **I**nspiration and **R**ecognition of **S**cience and **T**echnology) LEGO League robot challenge, open to student ages 9–14, grew from 200 student teams in the US in 1998 to more than 4,600 student teams in the US in 2006 and more than 2,800 student teams elsewhere in the world (http://www.usfirst.org). RoboCup and RoboCup Junior contest (http://www.robocup.org/) is another international event. Its goal is to foster artificial intelligence and robotics research by providing a standard problem where a wide range of technologies can be examined and integrated (Sklar et al. 2003).

Some of the many other local, national or international competitions carried out across Europe are listed below:

- *RoboParty* by the Robotics Group at University of Minho (Guimarães, Portugal) (http://www.robotica.dei.uminho.pt), where participants build robots from scratch
- *CEABOT* (http://www.robot.uji.es/research/events/ceabot08), a nationwide little humanoid robots competition by the RoboticsLab, University Carlos III de Madrid–Spain
- *RobotChallenge* (http://www.robotchallenge.at) for self-made, autonomous, and mobile robots, hosted in Vienna by the Austrian Society for Innovative Computer Science
- *Istrobot* held at the Slovak Technical University (http://www.robotika.sk) by the association Robotika.SK

A series of interactive exhibits, designed for learners to control in interesting game situations, have recently been made available at a special informal serious games centre in Athens called 'Polymechanon', which runs in informal education contexts without the constraints of the schooling system (Kynigos 2008). In Polymechanon, visitors can be directly immersed in collaborative games, where the more they understand what they control and how the robots respond to environmental change the better players they become. The concepts behind the games are robot's behaviors and aspects of the robot's environment that the human can control, the kind of control they have on these behaviors, the robot's responses to aspects of its environment and the consistent or changing roles of robots in the game at hand.

1.6 Competitions or Exhibitions?

Although competitions are motivating and beneficial in many aspects for students, exhibitions are suggested as an alternative approach that can support more collaboration and less antagonism. Exhibitions offer young people the opportunity to display their work to the public without the need to compete their schoolmates. If students are deeply involved in the design of their robotic projects, as well as in the design of the exhibition event itself, exhibitions can provide the same level of motivation and engagement, as compared with competitions (Rusk et al. 2007). Students and school community members of all ages can be invited in an exhibition to informally join and interact with each project and its creators. The open-ended nature of an exhibition format, while maintaining the motivational benefits of a public display of student projects, accommodates a wider range of abilities and offers room for a greater variety of creative expression (Turbak and Berg 2002).

1.7 The role of teachers and the TERECoP project

Athough the role of teachers in the effective introduction and use of robotics in the educational process is particularly important, only few projects have tackled the problem of teacher training in designing and implementing robotics in classroom settings. For example, Bers et al. (2002) present a methodology for teaching pre-service teachers to integrate technology in classroom following a constructionist approach. They describe experiences in which pre-service teachers design robotic projects to engage their students in exploring and learning new concepts and ways of thinking. The Student Teacher Outreach Mentorship Program (STOMP) at Tufts University (Portsmore et al. 2003) brought engineering students to educational settings as a support mechanism for teachers who were not familiar with robotics and engineering concepts, helping students with hands-on projects, resolving technical issues with equipment etc. Chambers and Carbonaro (2003) report a case study of a pilot teacher education course in robotic technology intended to design and develop a course that provides teachers with a solid understanding of robot design, construction, and programming, as well as of teaching using constructionist pedagogical strategies.

The **TERECoP** project (*Teacher Education on Robotics-Enhanced Constructivist Pedagogical Methods*, www.terecop.eu), involving 8 educational institutions from 6 European countries, is being activated in the field of teacher training in educational robotics. In the framework of the TERECoP project, a constructivist methodology meant to enable teachers to introduce robotics into their classrooms as learning tool in a constructivist context, was designed, implemented and evaluated in pilot training courses held in each of the 6 participating European countries (Alimisis et al, 2007; Alimisis, 2008; Papanikolaou et al, 2008; Arlegui et al, 2008b; Fava et al 2009).

Based on the premise that the use of robotics as learning tool requires from teachers a conceptual change from the idea of learning from technology, predominant in traditional computer-assisted instructional models, towards learning *with* technology in project-based learning environments (Carbonaro et al. 2004) and believing in the educator's axiom *"teachers teach as they are taught, not as they are told to teach"*, we designed a training methodology for future and in-service teachers aimed at engaging them in robotic activities that they could implement in a creative way with their own students. Pursuing the constructivist professional development of teachers, our course curriculum is inspired by the same constructivist spirit that we would like our trainees to foster in their school classes. Keeping line with the proposed use of robotics as a tool of constructivist learning, our course curriculum is meant to train teachers in the very way in which they are expected to educate their school students.

The idea of "learning by design" is central in our pedagogy supported by a project-based learning approach. The learning tasks of the course are organized as small or large scale robotics projects encouraging trainees to design and develop their own products. As Rusk et al (2008) point out, the way robotics is currently introduced in educational settings is unnecessarily narrow and they suggest that designing activities, focused on *themes* and not just on *challenges,* helps to engage wide and diverse audiences in robotics. In accordance with this idea, the projects proposed in our methodology focus on themes broad enough to give everyone freedom to work on a project according to their interests and are developed around open-ended problems engaging participants not only in "problem solving" but also in "problem finding" (Rusk et al, 2008).

The knowledge and the experiences gained, as well as the lessons learnt during the joint action of the TERECoP partnership lasted three years (2006-2009), are presented in the next chapters of this book, including valuable feedback from the teachers actively involved as trainees in our training courses. The authors' aspiration is to contribute to the progress of the relevant dialogue among the research community in the field and, more importantly, to convince teachers and teachers' trainers about the pedagogical potential of robotics and to provide them with training and learning methodologies, tools, examples, ideas and resources that they will, hopefully, find useful, when introducing robotics in a constructivist way in their school classes.

References

Alimisis, D. (2008). Designing robotics-enhanced constructivist training for science and technology teachers: the TERECoP Project. In the Proceedings of *World Conference on Educational Multimedia, Hypermedia and Telecommunications*, AACE, Chesapeake, VA, pp. 288-293.

Alimisis, D., Karatrantou, A. Tachos, N. (2005). Technical school students design and develop robotic gear-based constructions for the transmission of motion. In Gregorczyk G., Walat A., Kranas W., Borowiecki M.. (eds.), *Eurologo 2005, Digital Tools for Lifelong Learning, Proceedings,* Warsaw: DrukSfera, pp. 76-86

Alimisis, D., Moro, M., Arlegui, J., Pina, A., Frangou, S., Papanikolaou, K. (2007). Robotics & Constructivism in Education: the TERECoP project. In Ivan Kalas (ed.), *EuroLogo 2007, 40 Years of Influence on Education, Proceedings of the 11th European Logo Conference*, 19 - 24 August 2007, Comenius University, Bratislava, Slovakia

Arlegui, J. Menegatti, E., Moro, M., Pina, A. (2008a). Robotics, Computer Science curricula and Interdisciplinary activities, Workshop Proceedings of SIMPAR 2008, Intl. Conf. on SIMULATION, MODELING and PROGRAMMING for AUTONOMOUS ROBOTS, Venice(Italy) 2008 November,3-4, ISBN 978-88-95872-01-8, pp. 10-21

Arlegui, J., Fava, N., Menegatti, E., Monfalcon, S., Moro M., and Pina, A. (2008b). Robotics at primary and secondary education levels: technology, methodology, curriculum and science, in the Proceedings of the 3rd International Conference ISSEP INFORMATICS in SECONDARY SCHOOLS EVOLUTION and PERSPECTIVES, 1-4 July, 2008 Toruń, Poland

Bers, M., Ponte, I., Juelich, C., Viera, A. & Schenker, J. (2002). Teachers as Designers: Integrating Robotics in Early Childhood Education. *Information Technology in Childhood Education Annual*, AACE 2002 (1), 123-145.

Bers, M., Urrea, C. (2000). Technological Prayers: Parents and Children Working with Robotics and Values. In A. Druin & J. Hendler (eds.) *Robots for Kids: Exploring New Technologies for Learning Experiences*. NY: Morgan Kaufman, 194-217

Brooks, J., Brooks M. (1993). *The case for constructivist classrooms.* Alexandria, Virginia: Association for Supervision and Curriculum Development.

Carbonaro,_M., Rex, M., Chambers, J. (2004) Using LEGO Robotics in a Project-Based Learning Environment, 2004 *The Interactive Multimedia Electronic Journal of Computer-Enhanced Learning*, 6(1).

Chambers, J., Carbonaro, M. (2003). *Designing, Developing, and Implementing a Course on LEGO Robotics for Technology Teacher Education.* Journal of Technology and Teacher Education. 11 (2), pp. 209-241. Norfolk, VA: AACE.

De Michele, S., Demo, B., Siega, S. (2008). A Piedmont SchoolNet for a K-12 Mini-Robots Programming Project: Experiences in Primary Schools, Workshop Proceedings of SIMPAR 2008, Intl. Conf. on SIMULATION, MODELING and PROGRAMMING for AUTONOMOUS ROBOTS, Venice(Italy) 2008 November,3-4, ISBN 978-88-95872-01-8, pp. 90-99

Dias, M.B., Mills-Tettey, G.A, Nanayakkara, T. (2005). Robotics, Education, and Sustainable Development. *2005 IEEE International Conference on Robotics and Automation,* April, 2005, 4248 – 4253.

DiSessa, A. (2000). Changing minds, computers, learning and literacy. Cambridge, MA: MIT Press.

Druin, A., Hendler, J. (eds) (2000) Robots for kids: exploring new technologies for learning experiences. Morgan Kaufman/Academic Press, San Francisco

Fava, N., Monfalcon, S., Moro, M. Menegatti, E. Arlegui, J., Pina, A. (2009). Teacher training in the scientific field through Robotics activities: some experiences from Italy & Spain, *Proceedings of INTED2009 Conference, 9-11 March 2009, Valencia, Spain,* 747-756

Fiorini, P., Galvan, S., Giuliari, L., Pighi, L., (2008). It Takes a Village... to do Science Education, Workshop Proceedings of SIMPAR 2008, Intl. Conf. on SIMULATION, MODELING And PROGRAMMING For AUTONOMOUS ROBOTS, Venice(Italy), ISBN 978-88-95872-01-8, pp. 43-53

Fosnot, C. (1997). *Constructivism: Theory, perspectives and practice.* New York, New York: Teachers College Press.

Kafai Y., Resnick M. (Eds) (1996) *Constructionism in Practice,* Lawrence Erlbaum Associates Publishers, Mahwah, New Jersey.

Kärnä-Lin, E., Pihlainen-Bednarik, K., Sutinen, E., Virnes, M. (2006). "Can Robots Teach? Preliminary Results on Educational Robotics in Special Education," icalt, pp.319-321, Sixth IEEE International Conference on Advanced Learning Technologies (ICALT'06).

Kynigos, C. (2008), Black-and-white-box perspectives to distributed control and constructionism in learning with robotics, Workshop Proceedings of SIMPAR 2008, Intl. Conf. on SIMULATION, MODELING and PROGRAMMING for AUTONOMOUS ROBOTS, Venice(Italy) 2008 November,3-4 ISBN 978-88-95872-01-8, pp. 1-9

Kynigos, C. (1995). Programming as a Means of Expressing and Exploring Ideas in a Directive Educational System: Three Case Studies. *Computers and Exploratory Learning,* diSessa, A, Hoyles, C. and Noss, R. (eds), Springer Verlag NATO ASI Series, 399-420.

Kynigos, C. (2004). A Black-and-white-box Approach to User Empowerment with Component Computing, *Interactive Learning Environments*, Carfax Pubs, Taylor and Francis Group, Vol. 12, Nos. 1–2, 27–71.

Kynigos, C. (2007) Half-Baked Logo Microworlds as Boundary Objects in Integrated Design, *Informatics in Education,* 2007, Vol. 6, No. 2, 1–24, Institute of Mathematics and Informatics, Vilnius.

Kynigos, C., Latsi, M. (2006) Vectors in use in a 3D Juggling Game Simulation, *International Journal for Technology in Mathematics Education,* 13.1, 3-10.

Martin, F (1996) ''Kids learning engineering science using LEGO and the programmable brick.'' Presented at the annual meeting of the American Educational Research Association, New York, NY, 8–12 April 1996

Martin, F., Mikhak, B., Resnick, M., Silverman, B. and Berg, R. (2000). To Mindstorms and Beyond: Evolution of a Construction Kit for Magical Machines, Morgan Kaufmann Series in Interactive Technologies**,** Robots for kids: exploring new technologies for learning**,** Pages: 9 - 33

Mataric, M. (2004). Robotics Education for All Ages, Proceedings, AAAI Spring Symposium on Accessible, Hands-on AI and Robotics Education, Palo Alto, CA, Mar 22-24, 2004

Mitnik R., Recabarren, M., Nussbaum, M., and Soto, A. (2009). Collaborative robotic instruction: A graph teaching experience, *Computers & Education,* Article in Press, Available online 25 March 2009.

Mor, Y., Hoyles, C., Kahn, K., Noss, R. and Simpson, G. (2006) Designing to see and share structure in number sequences, *International Journal for Technology in Mathematics Education*, 13, 65-78.

Papanikolaou, K., Frangou, S., Alimisis, D. (2008). Teachers as designers of robotics-enhanced projects: the TERECoP course in Greece, Workshop Proceedings of SIMPAR 2008, Intl. Conf. on SIMULATION, MODELING and PROGRAMMING for AUTONOMOUS ROBOTS, Venice(Italy), 2008 November 3-4, ISBN 978-88-95872-01-8, pp.100-111.

Papert, S. (1980). *Mindstorms. Children, Computers and Powerful Ideas*. New York: Basic books.

Papert, S. (1992). *The Children's Machine*. N.Y.: Basic Books.

Papert, S. (2002). The Turtle's Long Slow Trip: Macro-Educological Perspectives on Microworlds. *Journal of Educational Computing Research*, 27(1), 7-28.

Pekárová, J. (2008). Using a Programmable Toy at Preschool Age: Why and How? Workshop Proceedings of SIMPAR 2008, Intl. Conf. on SIMULATION, MODELING and PROGRAMMING for AUTONOMOUS ROBOTS, Venice(Italy) 2008 November,3-4, ISBN 978-88-95872-01-8, pp. 112-121

Petrovič, P., Balogh, R. (2008). Educational Robotics Initiatives in Slovakia, Workshop Proceedings of SIMPAR 2008, Intl. Conf. on SIMULATION, MODELING and PROGRAMMING for AUTONOMOUS ROBOTS, Venice(Italy) 2008 November,3-4, ISBN 978-88-95872-01-8, pp. 122-131

Portsmore, M., Rogers, C. & Pickering, M., (2003*),* "STOMP: Student Teacher Outreach Mentorship Program", *Proceedings of the 2003 American Society for Engineering Education Annual Conference & Exposition,* American Society for Engineering Education*,* available at http://www.asee.org/acPapers

Portsmore, M., Rogers, C., Lau, P. & Danahy, E.. (2004). Remote Sensing and Tele-robotics for Elementary and Middle School Via The Internet. *Computers in Education Journal*, XIV (2), 72 – 75

Resnick, M. (1993). Behavior Construction Kits. *Communications of the ACM*, vol. 36, no. 7, pp. 64-71.

Resnick, M. (1998). Technologies for Lifelong Kindergarten *Educational Technology Research & Developmen,* vol. 46, no. 4

Resnick, M. (2006). Computer as Paintbrush: Technology, Play, and the Creative Society. In Singer, D., Golikoff, R., and Hirsh-Pasek, K. (eds.), *Play = Learning: How play motivates and enhances children's cognitive and social-emotional growth*. Oxford University Press.

Resnick, M., and Ocko, S. (1991). LEGO/Logo: Learning Through and About Design. In *Constructionism*, edited by I. Harel & S. Papert. Norwood, NJ: Ablex Publishing.

Resnick, M., Berg, R. and Eisenberg, M. (2000) Beyond Black boxes: Bringing transparency and aesthetics back to scientific investigation, *Journal of the Learning Sciences*, (9) 7-30.

Resnick, M., Bruckman, A., and Martin, F. (1996a). Pianos not stereos: Creating computational construction kits. *ACM Interactions.* September/October 1996.

Resnick, M., Martin, F. G., Sargent, R., & Silverman, B. (1996b). Programmable bricks: Toys to think with. *IBM Systems Journal,* 35(3&4), 443-452.

Rogoff, B., Lave, J. (Ed.) (1984). Everyday *Cognition: Its Development in Social Context. Cambridge*, MA: Harvard University Press.

Rusk, N. Resnick M., Berg, R., Pezalla-Granlund M. (2008) New Pathways into Robotics: Strategies for Broadening Participation, J Sci Educ Technol (2008) 17:59–69

Sarama, J., & Clements, D. (2002). Design of Microworlds in Mathematics and Science Education, *Journal of Educational Computing Research*. 27(1), 1-3.

Sklar, E. Eguchi, A.and Johnson J. (2003), RoboCupJunior:Learning with Educational Robotics, in G.A. Kaminka, P.U. Lima, and R. Rojas (Eds.): RoboCup 2002, LNAI 2752, Springer-Verlag, Berlin, Heidelberg pp. 238–253.

Turbak, F., Berg, R. (2002) Robotic design studio: exploring the big ideas of engineering in a liberal arts environment. J Sci Educ Technol 11(3):237–253

Chapter 2

Robotics as learning object

Authors: Javier Arlegui, Nello Fava, Emanuele Menegatti,
Stefano Monfalcon, Michele Moro, Alfredo Pina

2.1 Methodological Aspects for a Constructivist Teaching and Learning of Robotics

What are the main aspects to focus on when robotics activities are developed in the classroom and how? From a constructivist point of view there are two main subjects for discussion: which "learning objects" should be built and how to carry out this constructive activity, always in relation to educational robotics?

2.1.1 Which Knowledge Has to Be Built into Robotics Education?

In this section the key focus points will be on the following questions:

- What have students to learn through robotics activities?
- Which are the "learning objects" in educational robotics?

We have to conduct at least one elementary didactic transposition exercise, this identifies what to retain from the scientific, technical and social knowledge of the robots. This keeps it as the "school-teaching-robotics" knowledge which is the "didactic" knowledge.

Robots as physical systems to be programmed

A robot is defined as an "intelligent machine" implemented as an electrical and mechanical system that can be programmed to emulate human actions.

"We build machines that perceive, understand language, have common sense, learn, and act in the world…and our hypothesis is that humanoid intelligence requires humanoid interactions with the world." (Adams et al. 2000)

The traditional interaction abilities of robots were related once to the physical world, but an interaction with the social world had to be developed as well.

"A sociable robot will be able to understand us, to communicate and interact with us, to learn from us and grow with us. It will be socially intelligent in a humanlike way. Eventually, sociable robots will assist us in our daily lives, as collaborators and companions. Because the most successful sociable robots will share our social characteristics, the effort to make sociable robots is also a means for exploring human social intelligence and even what it means to be human." (Breazeal, 2002).

In conclusion, a “school-robot” should have the following features:

1. A robot must *act in the physical environment* and perform sequences of actions to achieve some preset objectives.

2. A robot must *interact with the physical environment* and take *decisions* about the way in which to perform its tasks, based on the perceived properties of such environments.

3. A robot must *communicate with other robots* to exchange basic information that gives it a “formal knowledge” of the environment.

4. A robot must *have some knowledge to act* and obey a structured and functional *computer program*, which describes and guides its behaviour, actions, interactions and communications.

All these four characteristics should be taken into account when discussing a constructivist way of teaching and learning school-robotics.

Actions, States and Behaviour

The *acting in the physical environment* is related to three basic characteristics of a “school-robot”: the states, the actions and the behaviours.

At a given t instant, a robot is in a definite *state*, which is characterized by the values of its “state-variables”, i.e. the Kinematics variables. These values specify the “properties” of the state (the robot properties) at that time.

The *actions* are aimed at changing the state of the robot causing a progression of the states which can be best described by verbs such as “turning”, “rising” etc.

Robots´ *behaviour* can be seen as the whole sequence of states, from the initial to the final objective state, caused by one or more actions (see figure 1).

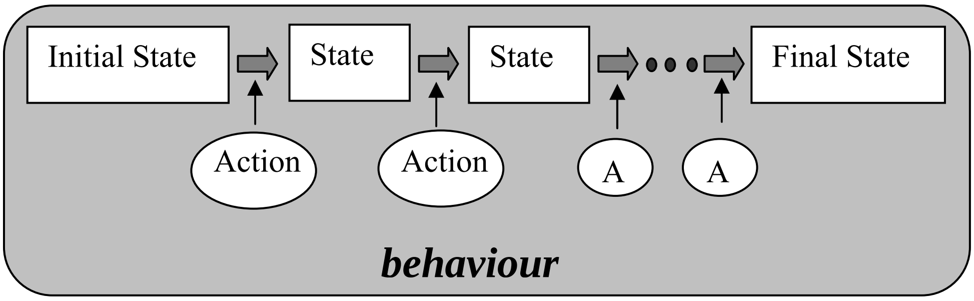


Figure 2.1.1: *Actions, States and Behaviour*

The possible actions of a robot are restricted to the physical environments and their laws. Robots do not necessarily do what students want them to do, but what they can do in interaction with the environment. Robots can "resist" the student's task-formulations, forcing them to enter a phase of exploration. This constructivist learning is based on the student's cognitive imbalance provoked by such "resistance".

Physical Environment Interaction and Conditional Formulations

In human language, as well as in formal robot language, actions are always context dependent. In pre-established contexts, a robot's task stands as an "imperative text" which is made up of a linear sequence of commanding instructions. This will be referred to as an "*action text*". For example: In LOGO language, the task to tell the robot to walk along a square is: REPEAT 4 [GO_FORWARD :length TURN_RIGHT 90]

When the context is not predefined, we must provide all the possible contextual conditions and find suitable tasks for the robot to achieve its behaviour. We have to make a hypothetical-deductive reasoning which has to be written as a "*true-false conditional text*" which controls the contextualized *action texts* (see figure 2).

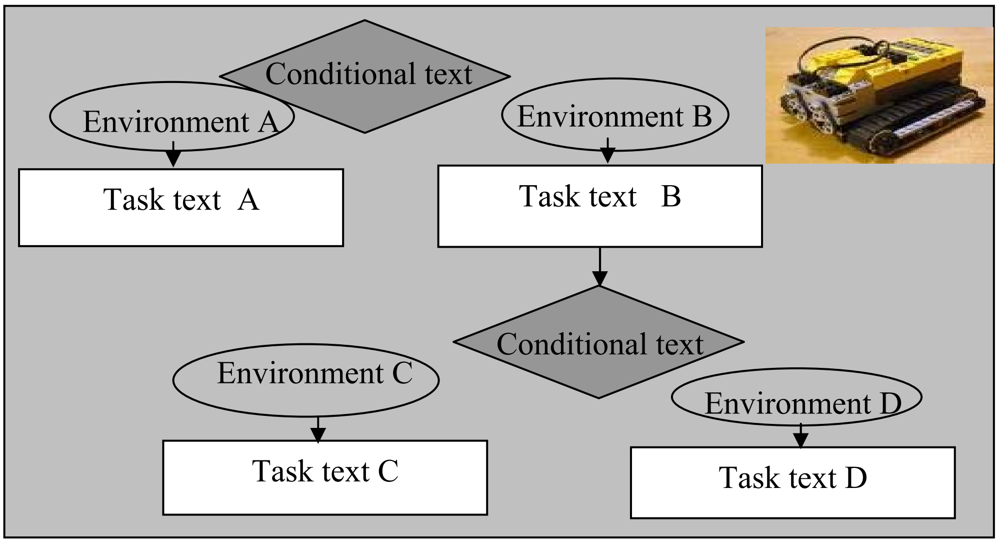


Figure 2.1.2: *a hypothetical-deductive reasoning*

Therefore, a conditional task stands as a two-level hierarchical text program, which has control instructions such as if, then, else, when, until...when the selected specific command instructions are to be executed. The conditional text constitutes a meta-text of the action text and imposes a no linear linguistic structure to the global text, with a hyper-textual or hierarchical character and a higher level of complexity. Robots must be prepared to do tasks with a flexible and context-adapted behaviour, thus robot programming must include these conditional properties as an intrinsic aspect. This will also require a new level of constructivist activity by the students, which will be shown below.

Communication amongst Robots and Levels of Communication

Although this is not a primary goal of the project, social robots are an extension of individual physical robots, creating bigger potential for teaching and learning situations in Primary and Secondary schools with the use of constructivist methods.

Any system of multiple robots is characterized by robots that operate in the same environment in a cooperative way with a view to achieving a global goal. This cooperation consists in the exchange of significative information among robots, whereupon their individual behaviour is regulated.

As Jiménez, Ovalle and Branch say (Jiménez et al. 2008), in a multiple-robot system there can be two kinds of communication.

A non intentional or indirect communication is either where information is transmitted by modifications of the environment made by the robots, or by modifications of the robots. In this kind of communication, messages are not received specifically and, therefore, there is no guarantee of reception. For example, in a "sumo" competition, a robot must try to “communicate” with the other robot, looking for where it is through distance or contact sensors.

At this level, there is no established channel; instead, there is just communication by external symbols produced by the robots. Robots must be fully capable to distinguish each other. This kind of communication is only useful in solving problems that do not require intentional coordination. However, it is very important that robots do not interfere with each other.

An intentional or direct communication is the second type. Robots use specific communication channels to communicate effectively. Messages have well-defined transmitters and receivers, which can be differentiated at two levels, depending on the complexity of the communication.

Communication based on states: the values of the robot internal state-variables constitute the important information. These values can, in turn, be obtained from the environment and can be used to “teach” other robots about them. For example: A robot adapted to locating seats of fire along its linear path can send a message to another robot telling it the position and the temperature of several fire seats, encoding this information as a list of sequential two values positions, one for the X position of the robot (in cm) and the other for the T temperature (in ºC) given by a sensor (when the temperature values are greater than 70 ºC). Within this protocol, the message [[110 80] [165 75] [240 85]] means that there are three fire seats (with 80, 75 and 85 ºC) at distances of 110, 165 and 240 cm from the origin.

Communication based on shared objectives: the information is made up of more complex texts that try to emulate the communicative roles, the typologies and textual functions of natural languages._For instance: The communication between the “She-Duck” robot and her three “Ducklings” robots could be, with a previous known protocol with acknowledge, as follows:

```
BROADCAST MESSAGE [FROM She-Duck TO [Duckling1 Duckling 2 Duckling 3]
FOLLOW_ME]
MESSAGE [FROM Duckling 1 TO She-Duck OK]
MESSAGE [FROM Duckling 2 TO She-Duck OK]
MESSAGE [FROM Duckling 3 TO She-Duck OK]
```

The present school robots (as the ones used in the TERECoP project) are able to set up intercommunication possibilities amongst themselves. There are university projects with an advanced development on this subject[i]_. The latest developments are those referring to a constructivist methodology about “social robots” in Primary and Secondary schools (Picard et al. 2004).

Knowledge, Task formulation and Programming

The students can not directly act on a robot. They act on a representational space; they build a text, and this text controls the behaviour of the robot. In a computer environment, “writing is acting”. We label the text as “tasks” that command robots´ behaviour. A task is written as an instruction (or a sequence of instructions) in a computer program. One task creates a particular kind of_behaviour (see figure 3).
In turn, the specific tasks come from “generic tasks” that command “generic behaviours” (or class-behaviours). These generic tasks are implemented as *procedures* in a computer language and become the real robot “knowledge”. Teaching and learning to build up these procedures will be the main objective of our constructivist approach.

The Nature of the Formal Objects that are built in the Programming for Robots

As we have seen above, the school exploration with robots is the formal exploration carried out by students to construct "good programs”. But, what are the formal objects that the students build when they program? And what is their nature? We are going to approach this from a didactic point of view, which is particularly interesting and well adapted to our objectives. This is the theory of "praxeological organizations» (Chevallard, 1999).

Behaviours: **T**he framework of the Teaching Anthropological Theory (TAD) postulates that human intervention (and robot intervention) in the environment is achieved through the implementation of specific actions aimed at a particular target in a given context. We call these specific actions “behaviour”. For instance, drawing a square of 20 cm per side by a pupil is a concrete behaviour.

Tasks: The behaviour can be strictly private, but often the behaviour is culturally pre-established and can be formulated and communicated. The formulation of a particular behaviour aimed at a specific goal is what we call a "task". The teacher's request "draw a square of 20 cm per side on the board" would be the task, in the above example.

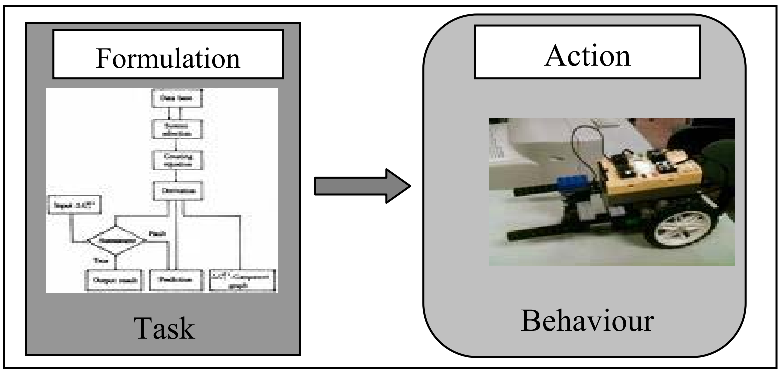


Figure 2.1. 3: *Praxeological Organizations*

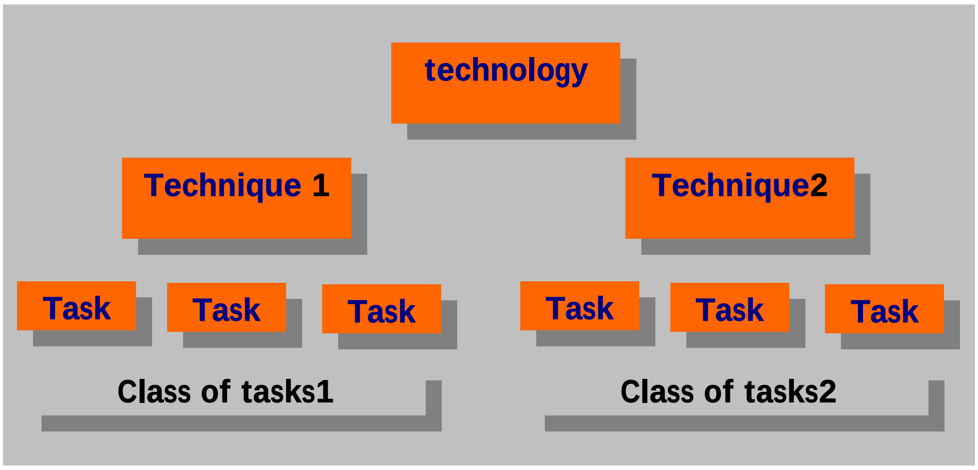


Figure 2.1.4: *Praxeological Organizations*

Classes of tasks: The social experience of human beings groups together cognitive and linguistic tasks into "classes of tasks". One class of tasks brings together tasks that are "similar" in the sense that they have similarities and differences. They have the same values for certain properties (the identifying variables) and different values for others (the discriminatory variables or state-variables). "Draw a square" is a class of tasks. The length of the square is the discriminatory variable (with different values for each square), and the number of sides and the angle are the identifying variables (always number of sides = 4, and angle = 90).

Techniques: The texts describing how to resolve a class of tasks are called texts of technical know-how, or simply “techniques”. In the above example: “Repeat four times [go forward a D distance and then turn right 90º]” is a technique.

Technologies and Theories: We can think of a meta-text on techniques, that is, a text that expresses “know-how” about a class of techniques. We refer to this text by the term “*technology*”. In the above example, once the techniques to draw squares, rectangles, diamonds… are known, we can build up the technique to draw regular polygons. In the process of making meta-texts more and more inclusive and abstract, we can formulate more academic texts. We refer to this whole text structure by the term “*theories*”. In our project, it is enough to think about the elementary levels of the theories: techniques and technologies.

Implications of the Praxeological Analysis

In the framework of the praxeological organizations, we will say that a human being and, by extension, a robot, performs a socially "intelligent" action, when it corresponds to a task that has been formulated and specified on the basis of a technique.

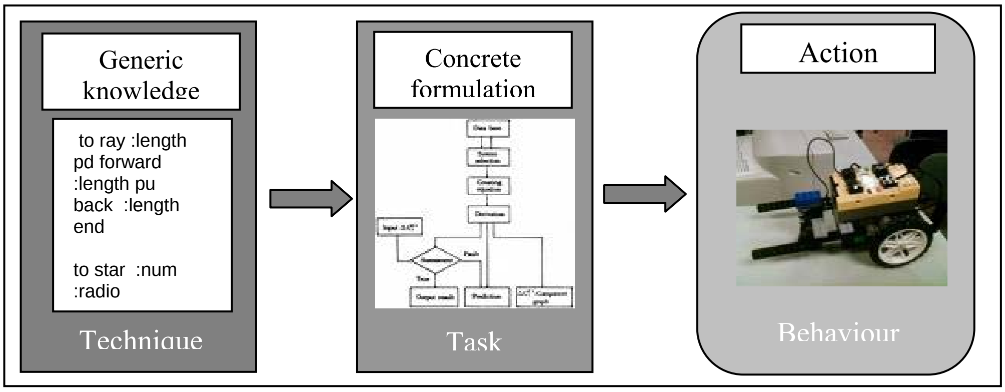
When one learns and uses a technique, the tasks associated with it are no longer "problematic” to solve, but simply “tasks” to accomplish. They give rise to the “intelligent” behaviour, as opposed to the stuttering actions corresponding to unspecified "trial and error" techniques. Learning to solve a problematic situation means moving from a problem-situation to a task-situation and needs the previous acquisition of the proper technique, that is, the building-up of a know-how text.

If we get this generic knowledge, if the technique is understood, it can be used to write the “formulated-action” adapted to the problem (the written task) that will originate, finally, the “physical-action” (the behaviour). Thus, constructivist learning in robotics is basically the pupils´ construction of robot techniques, the construction of techniques related to class-tasks (generic solving methods).

Textual Aspects of Techniques and Tasks

Techniques are written as “general propositions” and must be expressed in terms of generic class with state variables representing the formal parameters of the technique. Techniques are sentences such as *Turn “g” degrees*; *Lift up the arm “y” cm; Forward “x” meters with “v” meters per second…*

Tasks, by contrast, are “local propositions” that instance a technique for a particular case by specifying the actual parameters. Tasks are sentences like: *Turn 360 degrees; Forward “2” meters with 0.5 metres per second …*



```
 to ray :length
pd forward
:length pu
back  :length
end

to star  :num
:radio
```

Figure 2.1.5: *from techniques to behaviours*

Summarizing…

- Robots´ knowledge is implemented as a computational technique (The LEGO NXT blocks are excellent examples of techniques).
- A robot can have “knowledge for action” implemented as computational command techniques.
- A robot can also have “knowledge for decision-making” implemented as computational logical techniques.
- A task is implemented as a computational instruction.
- A task causes a robot’s behaviour. A simple task causes basic computer behaviour and a computational program with a sequence of instructions causes a more complex behaviour.
- The invocation of procedures within the new procedures allows a hierarchical -“praxeological-like” structured knowledge.
- It can be seen that the capacity and interest of a language like LEGO NXT expresses techniques and tasks which build up praxeological structures. These are well adapted to a constructivist teaching and learning process of problem-solving.

2.1.2 How to Carry out Constructivist Learning and Teaching on School Robotics?

Constructivism as a Learning Process

The expertise in commanding tasks for robots is such that they have certain behaviours with a goal in mind and can be the object of constructivist education on the teacher's side and learning on the student's side. It is necessary to select and adapt to our objective the most pertinent characteristics of the theories of Piaget and Vygotsky, known as cognitive reconstruction theories and assuming constructivist teaching and learning.

Remembering Piaget…

The Piaget theory is a theory of the *dynamic construction* of knowledge. Piaget bases this construction on the process of "improving" adaptation that is formulated as the tendency to a growing equilibrium (balance) between the processes of assimilation and accommodation gets under way.

Assimilation:
*The process by which people interpret the information coming from the environment, depending on their available conceptual structure***s**
scheschemes available"

Accommodation:
The process of modification of conceptual structures by people, when they try to assimilate new characteristics of the environment

The *assimilation* suggests that "we see" all things not as they are, but as "we are", according to our available patterns of understanding. We only incorporate from reality those inclusive elements that can be recognized by our previous *schemes*. If only assimilation was there, much of our knowledge would be unreal fantasy and would lead to constant mistakes.

The *accommodation* explains the tendency of our schemes of assimilation to adapt themselves to reality, becoming more "consistent" (or balanced) with it. If these schemes are insufficient to assimilate a given situation, they will probably require future modification to interpret additional characteristics of the situation. The accommodation supposes not only a modification of the previous schemes based on the assimilated information, but also a new assimilation or reinterpretation of the previous knowledge based on the new constructed schemes. It is what we call "reconstruction" and is the most important effect of the constructivist process.

Teaching and Learning as a Process of Successive Re-equilibration

When students first start contact with new knowledge, they are generally imbalanced. They apply their previous cognitive schemes and usually assimilate only part of the aspects of the object. To develop a progressive adaptation, the student has to do a double work: a *direct empirical interaction* with the object and *a linguistic interaction* with a teacher (in reference to the object). This will facilitate the student to have a progressive adaptation both to the understanding of the actions of this object and to the understanding of the terms of the language with which we describe these actions.

This will take students to a first *re-equilibration stage*, but a further interaction with the object and / or a new problematic question of the teacher regarding the object will lead students to a new situation of *disequilibrium* that must be overcome through the same procedure to reach a new state of equilibrium (see figure 6).

The teacher's role is to trigger successive "controlled" imbalances to the students through "controlled" questions, being careful in order not to introduce too many new features in each question, and then helping them to reach a new re-equilibration stage with "demonstrations" in the Vygotskian way, showing "well realized" actions on the object, and "well formulated" linguistic expressions referring to the object.

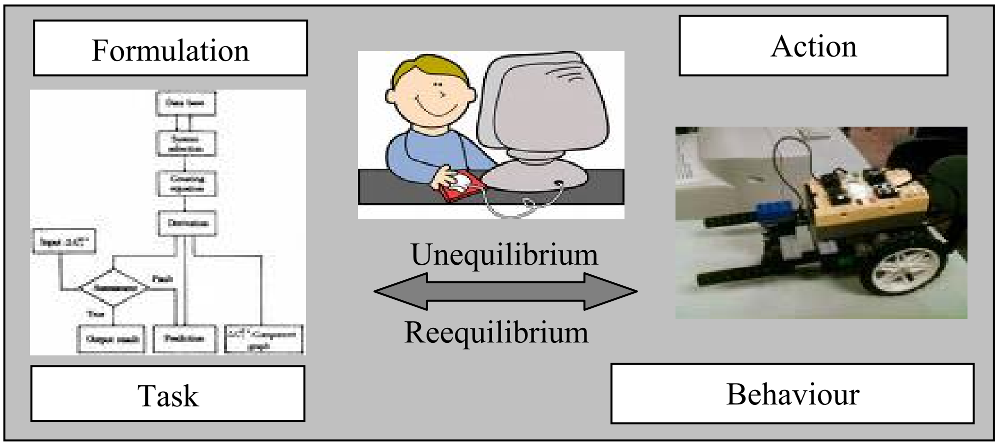


Figure 2.1.6: *the interplay between disequilibrium and equilibrium*

The role of the student is essentially to be intellectually "active" in the process, striving to identify new "inclusors" (new cues of reference) in the previous schemes and trying to give significance to the "demonstrations" of the teacher.

We must emphasize that when the students have to deal with "linguistic objects", for instance, to construct adequate programming "techniques", the individual constructivism of the student does not fit. On the contrary, it is necessary for a student to engage with "guided" constructivism and act for a "double" reequilibration: the student interacting with the object for a "semantic" feedback while talking with the teacher for a "syntactic" feedback.

Levels of Complexity of the Re-equilibration

Piaget elaborated on several models of the equilibrium process (Pozo 1989). In the latest of them, he says that the equilibrium between assimilation and accommodation takes place at three levels of increasing complexity:

- The first level is *equilibrium with the facts*. The initial schemes must reach a balance with the new objects to assimilate.

- The second level *equilibrium with the schemes* is a balance between old and new schemes that have to be reached in order to be assimilated and accommodated to each other.

- The third level *equilibrium with the hierarchic structure of schemes,* where new re-equilibration of the schemes hierarchical structure should be achieved.

Class Problems for a Constructivist Teaching and Learning Process in Robotics

The robotic education should proceed with the *formulation of successive problems* grouped as class-problems:

a. *Problems on the same class of behaviours (robot actions),* which must give origin to the same class of programming tasks, from which "the best" technical programming procedure (a "technique") must arise. For instance, to instruct the robot to walk along a square of 20, 30, 40, … cm per side is a class of tasks.

b. *Problems on a new class of behaviours*, close to the previous one, but slightly different, from which a new technical procedure, different from the previous one, (a new "technique") must arise. The comparison and contrast between those two techniques should create a "technological" piece of knowledge. A contrast between the technical procedures to build up squares, rectangles, diamonds,… could be lead to the construction of a more general procedure to move a robot along a generic parallelogram (having the length, the width and the two angles as parameters), which is a technological procedure.

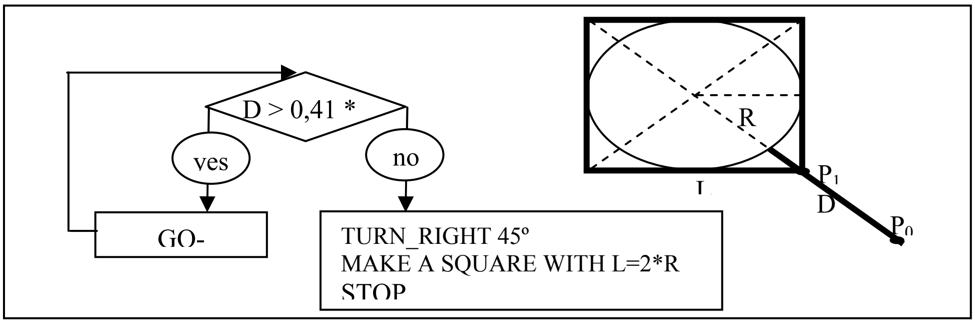


Figure 2.1.7: *a conditional procedure*

c. *Problems on a conditional (context adapted) behaviour* for a class of contexts, from which a conditional procedure should arise to control the conditional tasks to be done, building the "knowledge for decision-making" mentioned above. For example, making a robot move along a square circumscribed on a circumference of radius R, starting the robot initially at an external point P_0, is a conditional problem. To solve it we have to write a conditional procedure (see figure 7): While the robot is approaching the circumference from P_0 (measuring its distance D with an infrared sensor) we have to decide if it has already reached the P_1 point to start then the square route.

2.1.3 Designing Didactic Situations in a "Constructivist" Robotic Course in the Classroom

The Curricular Dimension of Constructivism

For strict consistency, a course that focuses on constructivist learning should lead a constructivist methodology. This methodology, as we shall see, should be focused on learning based on problems and projects. What does a "constructivist course" mean? A "constructivist" model usually emphasizes the *learning* process and thus the role of the learner. Learning must be active, meaningful, through inquiry, and the students are seen as the builders of their own autonomous learning.

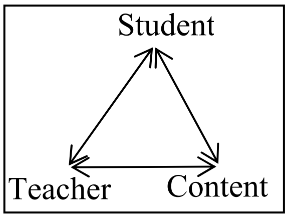


***Figure 2.1.8**: student's relationship with teacher and content*

Often, little is said of the *teaching* process and the teacher's role, which is regarded as that of a mere facilitator of the above process ("…It's enough to provide the students with rich environments" in order to ensure their constructivist activity). Also, little is said about the role of content (curricular subjects), as if "anything goes" if multiple learning activities can be designed around it.

This expression of constructivism is primarily a *psychological proposal* on the methodology of the student learning. This is especially true in "natural" contexts (pupils learn from the environment without a teacher), more than in school contexts. It is *not a curriculum proposal* because it focuses on the student autonomous work and neglects the student's relationship with the teacher and the content (see figure 8), relationships that inevitably exist in school education. Constructivism in school is not only a matter of the student (or the social group of students) but a matter of the students in relation to the teacher and the content. These relations are to be made explicit in a curricular design (deciding what and how to teach, and how to learn).

The Importance of Formulating the "Problem":The problem is at the core of the constructivist activity

One must question the role of teachers in this type of course. As we deepen in the root of the child constructivist process, we see that Piaget identified it with the process of "unbalancing" and "rebalancing", associated with any new adaptation. The child constructs rebalanced new schemes with an environment that previously had changed and had become problematic. The construction process is the gradual process of "*re-equilibration*".

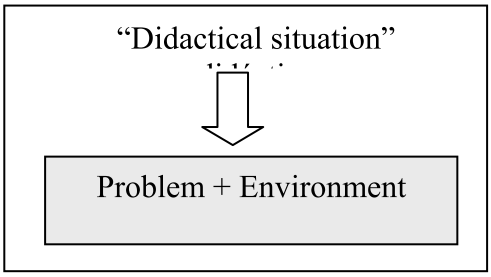


Figure 2.1. 9: *a specific exploratory interaction triggered by the problem*

In science and technology in school, how is the "environment" presented to students? It is a "pack" of two elements: a physical environment and a problematic question concerning it. It is not merely an environment that allows the student to make random interactions, but an environment that is offered for (see figure 9). This pair (problem and environment) is known as "didactical situation" and is the teachers´ duty to design and present it to pupils. Sometimes we refer to it simply as the "P problem."

Hence, the importance of this problem, which will guide the student exploratory-constructive activity, should be carefully planned by the teacher taking into account the following aspects:

- Any problem formulation should refer to aspects already known by the students and must contain one or more elements of novelty that cause an imbalance (disequilibrium) in the students. Any new problem should start below the last problem knowledge level so that the "cognitive distance" between problems (A, B, C ...) is small.

- Although in many cases it is difficult to predict the quantity of novelty the problem has, it should be just enough to allow a resolution by independent exploration on the part of the student (enquiring work). In most cases it should allow a resolution by working in Vygotsky's "zone of proximal development" with the help of the teacher, for example by developing activities undertaken jointly with the teacher, in which the teacher progressively transfers responsibilities to students helping them to solve the problem. In most cases teachers should play an active part in solving the problem.

In the diagram of figure 2.1.10:

- The student is initially at level A, equilibrated with a particular environment located at an A level of understanding in front of a given phenomenon. For example, a student can understand and skilfully use the "move" block (in Lego Mindstorms NXT-G programming language) to have the robot following different linear routes with different values of power and time.

- At the moment t_1, the teacher suggests the problem (P1), which requires the student to understand the environment from a new point of view (level B), in which the student does not fit. At this moment we say that the student is in a new didactic situation where he is imbalanced. For example, the teacher can ask the student how to reformulate the primitive "move" block to make a "my_move" block with only two parameters (power and time). This is a problematic task for the student that has no idea how to generate this new user block…

- From t_1 to t_2 the student initiates an active process of re-equilibration (assimilation and accommodation) that progresses to a new level of understanding, where the problem P1 leaves gradually the status of a "problem" and is becoming a "task". For example, the student may start to look for help on the NXT-G environment, reading about "my blocks and how to create them or starts to imitate the way the teacher acts when he shows him how to generate a "my_move" block…

- At the moment t_2, the student recognizes and solves the problem (P1) in a consistent manner. We say that he has achieved a re-equilibration with the environment at level B or simply that he has learned to solve the problem P. For example, the student finally generalizes the technique of how to generate a user block from pieces of previous blocks. Gradually the student becomes skilful in this technique using it in a class of programming situations. How to generate user blocks is no longer a problem for him, but a task to accomplish.

The Class-problems and the Generic Techniques

It should be remembered that, although for simplicity reasons we have talked about "problems" in the previous paragraph, we should always speak of "classes of problems." A "unit of learning" is built only when students give solutions to a set of problems of the same class. We say that we have found a *technique* (technical knowledge) to solve such class of problems.

It is, therefore, the teacher's responsibility to ask questions that are "class problems" of some generality, the answer to which requires the resolution of several "punctual problems" (P1, P2, P3 ...) of the same class. When the problem concerns an "artificial" environment linguistically controlled, such as a robot, writing new "techniques" for the robot requires a dual activity on the part of the student:

- An initial deductive enquiry activity explores (from the existing procedures) the instructions I1, I2...to solve the punctual problems P1, P2, (from the same class).
- A final inductive enquiry activity showing findings from I1, I2…and a general formulation to solve the class-problem P (integrating P1, P2…).

Building a technical knowledge to solve a class-problem involves for the students a process of successive approximations. These contribute to the improvement of the global adaptation from level A to level B, which is achieved when students construct a generic technique that allows them to solve any problem associated with the class problem.

Learning Objects and Tools for Learning

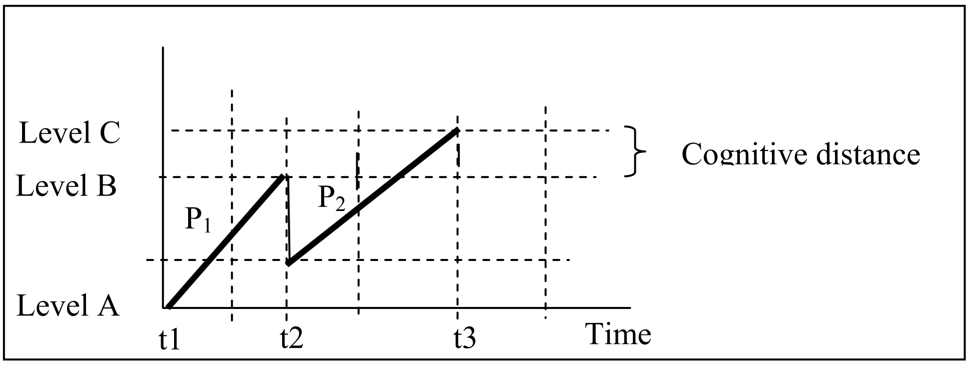


Figure 2.1.10: *an active process of re-equilibration to a new level of understanding*

Any object has potentially a double status: it may be taken as a "knowledge object", when we are studying and learning about it, and can be taken as a "knowledge tool", when we use the object (making it part of the interactive environment) to learn about something else. In the case of the robotics-based learning, the dual role the robot plays in the learning process seems clear:

- *The robot as a learning object* (building knowledge about the robot): The robot, as a physical and programmed object, should be the focus point of knowledge. Therefore, in a training course we will find class-problems related to the characteristics of robots. They will be sequenced gradually in topics depending on the increasing complexity of the robots movements and functions and their diverse sensors.

- *The robot as a learning tool* (building knowledge with robots): On the other hand, robots can be used to gain knowledge of certain characteristics of the physical environment.

The stages in a constructivist learning process

From the above we can afford to separate the process of building a constructive course in two components:

The task of the teacher: The teacher is responsible for

- the formulation and proposition of the problems
- the "Vygotskian" help to the students' learning
- the institutionalisation of knowledge emerging from the students

The task of the students:

- the "enquiry effort" to build up a piece of knowledge from a "didactical situation"
- the search for meaning with the building of "learning skills" ("know-how" learning),

Thus, a basic minimum teaching unit in this constructivist process should be able to show the following steps:

-*Formulation of a class-problem by the teacher*, referring to a class-behaviour of the robot.

Constructing particular solutions by the students, writing different tasks to accomplish different behaviours.

Generalizing the solutions: building up a procedure.

Empirical validation: getting evidence of good "behaviour" of the procedure.

Using the procedure: to solve new problems of the same class.

Looking for the limits of the procedure and adapting it to new-type problems.

An Example

a. Formulation of a Class-problem by the Teacher

Modern automatic trains (without drivers) stop after precise distances between stations... We could build a robot that acts like an “intelligent” train. To do so:

How can we make a robot travel distances D pre-established...?

b. Constructing Particular Solutions

task	distance AB (cm)	degrees
	0	0
1	50	1033
2	100	2070
3	150	3106
4	200	4138

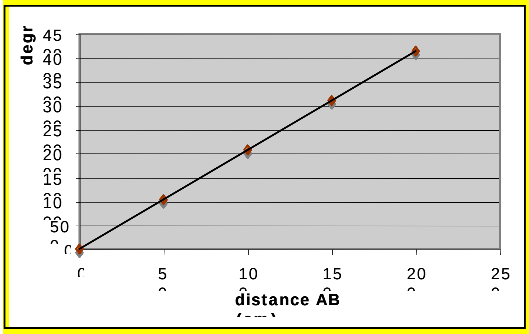


Figure 2.1.11: table and diagram of four tasks drawing linear routes with different distances

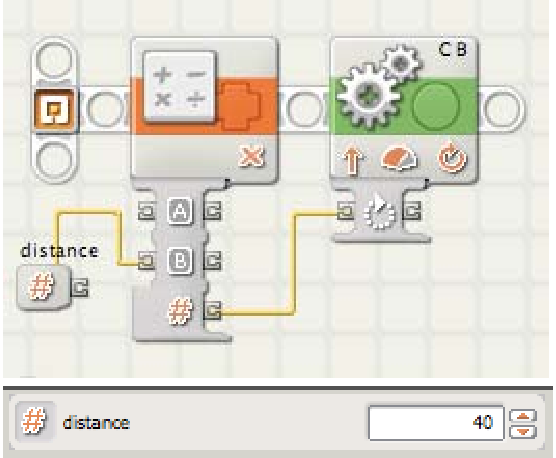
A group of pupils (Primary school) tentatively propose to use the primitive "motor" instruction, implemented in NXT-G, and initially decide to change the "degrees" (leaving the "power" with a fixed value).

They prepare a series of four tasks (of the same type), drawing four linear routes on the ground with different distances each: distance 1 = 50 cm; distance 2 = 100 cm; distance 3 = 150 cm; distance 4 = 200 cm. Students begin planning, by trial and error, various trips for each robot with different values of the "degrees" parameter until they get a trip with the desired distance.

They obtain finally the data shown in the figure 11 and draw their graphic representation. From the four “local” solutions the students can generalize that: *"the robot requires a value of 2070 degrees per hundred centimetres it progresses”* or, proportionally: *"the robot requires a value of 20.7 degrees per one centimetre it progresses”* and as a result: *degrees num. = distance in cm. x 20.7*

c. Generalization of Solutions: Development of a Procedure

With this "knowledge" students can now program the robot in a general way. They write the procedure called "*move_a _distance*", which takes as parameter the value of the variable "*distance*."



distance 40

Figure 2.1.12: *Development of a Procedure*

d. Empirical Validation

The students program a *move_a _distance* instruction with the value *40* as the distance parameter. They check the behaviour of the robot and measure the distance it travels and confirm that it travels 40 cm with high precision. They program different *move_a _distance* instructions with different distance values, confirming the good robot behaviour in any case.

e. Using the Procedure

From now, the students can use this procedure "as a tool" to make the robot move over specified distances, integrating it in other problems and projects.

f. Generalization of the Procedure to new Class-problems

What happens if we change the wheels of the robot? What happens if we simultaneously change the value of "Power" on the engine? These are examples of new class-problems that lead, by a similar sequence of steps, to construct a "second level" (based on two parameters) more general procedures.

References

Adams, B., Cynthia Breazeal, Rodney A. Brooks, Brian Scassellati (2000), "Humanoid Robots: A New Kind of Tool", IEEE Intelligent Systems and Their Applications: Special Issue on Humanoid Robotics, Vol. 15, No. 4, pp. 25—31.

Breazeal, C.L. (2002), Designing Sociable Robots (Intelligent Robotics and Autonomous Agents), Ed. Massachussets Institute of Technology. Boston.

Chevallard, Y. (1999), Teachers training in the Didactic antropological theory analysis (El análisis de las prácticas docentes en la teoría antropológica de lo didáctico), Recherches en Didactique des Mathématiques, Vol 19, nº 2, pp. 221-266.

Jiménez Builes, J. A. Ovalle Carranza, D.A. Branch Bedoya, J. W. (2008), Comunicación en sistemas de múltiples robots desde la metodología MAD-Smart, REVISTA INGENIERÍA E INVESTIGACIÓN, vol. 28, num. 2, pp 59-65.

Picard, R. W., Papert, S., Bender, W., Blumberg, B., Breazeal, C., Cavallo, D., Machover, T., Resnick, M., Roy, D., and Strohecker, C. (2004). Affective Learning — A Manifesto. BT Technology Journal 22, 4, pp 253-269.

Pozo, J.I., Cognitive learning theories, Ed. Morata, Madrid, 1989.

2.2 Some reasons to use Lego Mindstorms NXT

2.2.1 Introduction

The choice of an appropriate platform whereupon to develop educational robotics activities is very important and should be carefully planned by the instructors. In the TERECop project, different robotics platforms have been considered. What was sought was a general platform not tailored on specific activities or educational disciplines. The following requirements for the robotics platform mentioned above have been identified:

- It should be programmable at different complexity levels and should support different programming paradigms;

- It should be exploitable at many levels of complexity in order to be usable at different educational levels (i.e. at different ages);

- It should have simple, but significant possibilities of expansion. This could be achieved by plugging-in additional sensors or by interfacing that robotics platform with other devices to allow remote processing or remote control;

The ultimate choice was the LEGO MINDSTORMS NXT kit (in short NXT) (website: mindstorms.lego.com/). NXT fulfills the above requirements and has several other advantages. In our experience the most important advantages are:

- the start-up time for working with NXT is very short;
- the assembly of the robot is very intuitive and no electrical wiring is necessary. No workshop tools are needed, not even a screwdriver or a solder.
- NXT is very familiar to students. Almost all of them played with LEGO bricks. This motivates them. It reminds them of their toys rather than of their assignments.

NXT complies very well with the constructionist learning approach. First of all, a robot is a “*public entity*” in the Papert's sense (Harel and Papert 1991). NXT comes straight from Papert's experience and works. NXT is the last evolution of the work Papert started with Logo and continued with Dacta. NXT is modular and incremental. It leads to a bottom-up oriented development. Starting from the basic bricks, which define the fundamental standard for all the other elements of the Lego kits, you can build more and more complex architectures by combining simpler, already built parts.

NXT complies very well also with the TERECop philosophy of approaching initially the robot as a “learning object” to be investigated in order to understand how it works and how to control it, but then to use (or better to “**exploit**”) the robot as a “learning tool” whereby to study curricular disciplines. From that point of view, one of the advantages of NXT is that the students achieve very quickly the first step in which the robot is seen as a “learning object”. With NXT it is easy to learn how to build and how to program a robot. Thus, the students can move very fast to the second step in which the robot is seen as a “learning tool” to be accordingly utilized.

The modularity of NXT makes it very flexible and expandable as well. From a hardware point of view, beside the sensor and the motors provided by LEGO, several third part sensors and actuators are available whereby to enlarge experimenting possibilities. For instance, companies like Hitechnic (http://www.hitechnic.com/). and Mindsensors (http://www.mindsensors.com/) produce several sensors that can be directly read and logged by the NXT, while other companies, such as Vernier, (www.vernier.com/nxt/) produce NXT adaptors to connect their sensors for the most disparate scientific experiments.

NXT can be controlled and programmed via different programming languages and different programming environments. One can use the graphical programming environment NXT-G, developed by LEGO and National Instruments (www.ni.com/) or the C-like NBC and NXC or the Java-based LeJOS-NXJ. Next Byte Codes (NBC) (http://bricxcc.sourceforge.net/nbc/) is a simple open-source language with an assembly language syntax that can be used to program the NXT brick. Not eXactly C (NXC) (http://bricxcc.sourceforge.net/nxc/).is a high level open-source language, similar to C, built on top of the NBC compiler. It can also be used to program the NXT brick. NXC is basically *Not Quite C* **(**NQC) for the NXT. leJOS NXJ is a high level open-source language based on Java that uses custom firmware developed by the leJOS team (http://lejos.sourceforge.net/)

Moreover, one has the possibility to use several operating systems and/or platforms (URBI, Universal Real-time Behavior Interface (http://www.gostai.com/) for Windows, Mac OS X, Linux or NXT-Symbian running on Symbian 6.0 Java-enabled mobile phones (http://sourceforge.net/projects/nxt-symbian/).

In conclusion, the use of the NXT in the TERECop project was chosen because NXT is a good tradeoff between complexity and expansion possibilities. Last, but not least, cost was taken into consideration as well: the kit's cost allows the students (and the teachers, as well) to buy their own personal kits to continue experimenting at home.

2.2.2 Advantages of Lego Mindstorms (from teacher perspective)

From a teacher perspective, there are other advantages in using NXT in the class. NXT is a widely used platform and there are several resources on the web, ranging from discussion groups to several collections of lab activities at different educational levels. Being adopted worldwide, the teacher is not alone, but part of a large community, which provides technical support and many ready-to-use examples in the web (even if most of them do not have an appropriate methodological and/or didactical background).

Teachers willing to promote open-source philosophy among their students will find extremely valuable the LEGO choice of going open-source with the NXT firmware. Most of the software projects to fully exploit NXT capabilities are open-source projects and this allows teaching students how complex projects and innovative solutions can be built exploiting the open-source philosophy.

The Mindstorms NXT-G graphical interface is built on National Instruments LabView (www.ni.com/labview/). National Instruments and LEGO are collaborating since 1998 with the development of ROBOLAB, the programming software created for the original RCX LEGO MINDSTORMS. LabVIEW substituted the text-based programming approach for a graphical programming environment. It is used for automated measurement and control systems in many industrial test facilities. LabVIEW is used in several didactical laboratories (especially in technical schools) because of the ease with which it is used and the wide diffusion in the industries. There are many similarities between LEGO MINDSTORMS NXT-G software and LabVIEW. NXT-G retain all of the core of the LabVIEW graphical programming elements, while optimizing the user interface for novice computer users.

There is also a LabVIEW Toolkit for LEGO MINDSTORMS which is used to create and download files to operate and control NXT. Additionally, that toolkit makes it possible to create native blocks for MINDSTORMS NXT software. This allows a step-by-step migration from programming a NXT robot to programming an industrial test facility in LabView. The students will start programming their robots using NXT-G, then they can learn how to program NXT robots in LabVIEW, and, in the end, they will smoothly learn how to program and control real industrial measurement instruments and actuators with the full LabVIEW potentialities.

Similarly, one can use NXT robots to teach MATLAB and Simulink (www.mathworks.com/) programming. MATLAB is a high-level programming language for numerical computing, data acquisition and analysis. It can be used to control LEGO NXT robots over a Bluetooth serial port (serial port communication is part of the base functionality of MATLAB). Simulink is a MATLAB-based environment for modeling and simulating dynamic systems. Using Simulink, a user can design control algorithms, automatically generate C code for those algorithms and download the compiled code onto the LEGO NXT. Several examples of use of NXT programming to introduce MATLAB scientific programming capabilities have been developed, one example being the work developed by Prof. T. Aach at the RWTH Aachen University, which can be found at http://www.mindstorms.rwth-aachen.de/.

However, there is a major disadvantage with NXT: being so general and easy to use, it is somehow limited in its expansion capabilities. Indeed, once the students work with it for a year or more, they can easily reach the hardware limits of the NXT computer brick in terms of both, computational power and hardware expansion capabilities. At this stage, the student is mature to move on to more powerful, more flexible robotics kits. However, when moving to more advanced kits, the teacher should not loose the focus on the didactical and methodological aspects. In order to simplify that transition, examples of TERECop laboratory activities are provided, as implemented on small humanoid robots (see Chapter 5).

References

Official LEGO Mindstorms website: http://mindstorms.lego.com/

HiTechnic website: http://www.hitechnic.com/

Mindsensors website: http://www.mindsensors.com/

Vernier website for NXT sensors:, http://www.vernier.com/nxt/

Next Byte Codes (NBC) homepage: http://bricxcc.sourceforge.net/nbc/

Not eXactly C (NXC) homepage: http://bricxcc.sourceforge.net/nxc/

Not Quite C **(**NQC) homepage: http://bricxcc.sourceforge.net/nqc/

leJOS homepage: http://lejos.sourceforge.net/

NXT-Symbian homepage: http://sourceforge.net/projects/nxt-symbian/

URBI, Universal Real-time Behavior Interface webpage: http://www.gostai.com/

MATLAB and Simulink webpage: www.mathworks.com/

Harel, I. and Papert S. (1991). *Constructionism.* Norwood, New Jersey: Ablex Publishing Corporation.

2.3 Straight-line robots

2.3.1 Introduction

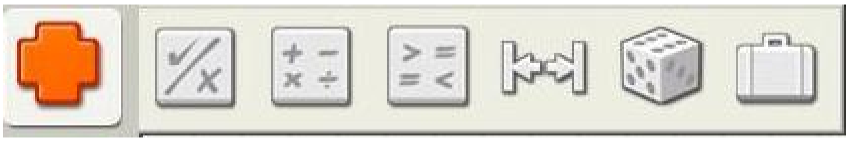
In this section, we will introduce the reader into the "designing-building-programming robots" loop process necessary to address the problem of "Robotics as learning object". We need to acquire all the skills necessary for designing, constructing and programming robots. From a methodological point of view we suggest that the proposed activities in the study of "Robotics as learning object" should be based on the constructivist educational methodology explained in other sections of this book. The format is, therefore, a progressive sequence of problems that can synthesize the constructivist path towards the "robot designing-building-programming" process. In this section, we start with the simplest possible robot i.e. a robot moving along a straight line.

The main objectives (competences or skills to be acquired) behind the problems presented here are:

- To design and build "good" straight line robots
- To know how the actuators (motors) work
- To know how to write, download and execute a NXT-G program
- To know how to create and manage variables and blocks (necessary for abstractions from NXT-G in order to construct primitives oriented to the user and/or to the problem)

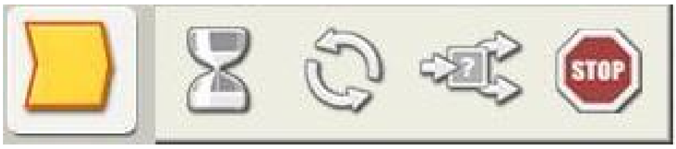
2.3.2 Necessary features from NXT-language

Using *arithmetic & logic operators*: Data group blocks



This group of icons allows the user to make any arithmetic or logic operation. The use of variables (explained later on) is also available here.

Using *Control Structures:* The Flow group is represented by three main different icons.

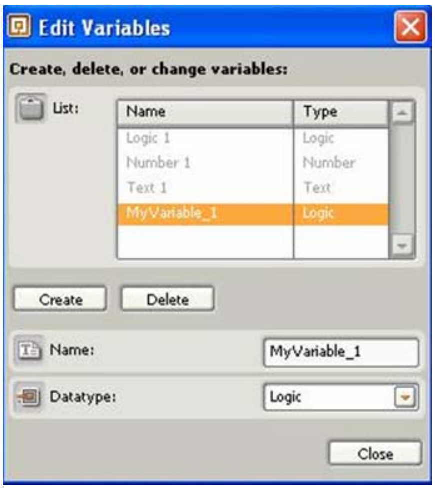


Wait: Waiting for an amount of time or waiting for a specific sensor event

Loop: Loop control structure (inclusive of either time or sensor events)

Switch: This block allows the use of *either If* statements or Case statements.

Creating and using *Variables:* To use a variable with LEGO Mindstorms, we need first to create it; to do so, we define the type, we write the name and then we click the "create" button.



Edit Variables

Create, delete, or change variables:

List:

Name	Type
Logic 1	Logic
Number 1	Number
Text 1	Text
MyVariable_1	Logic

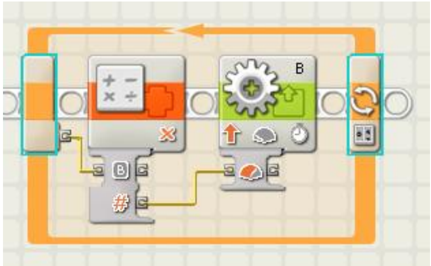
Create Delete

Name: MyVariable_1

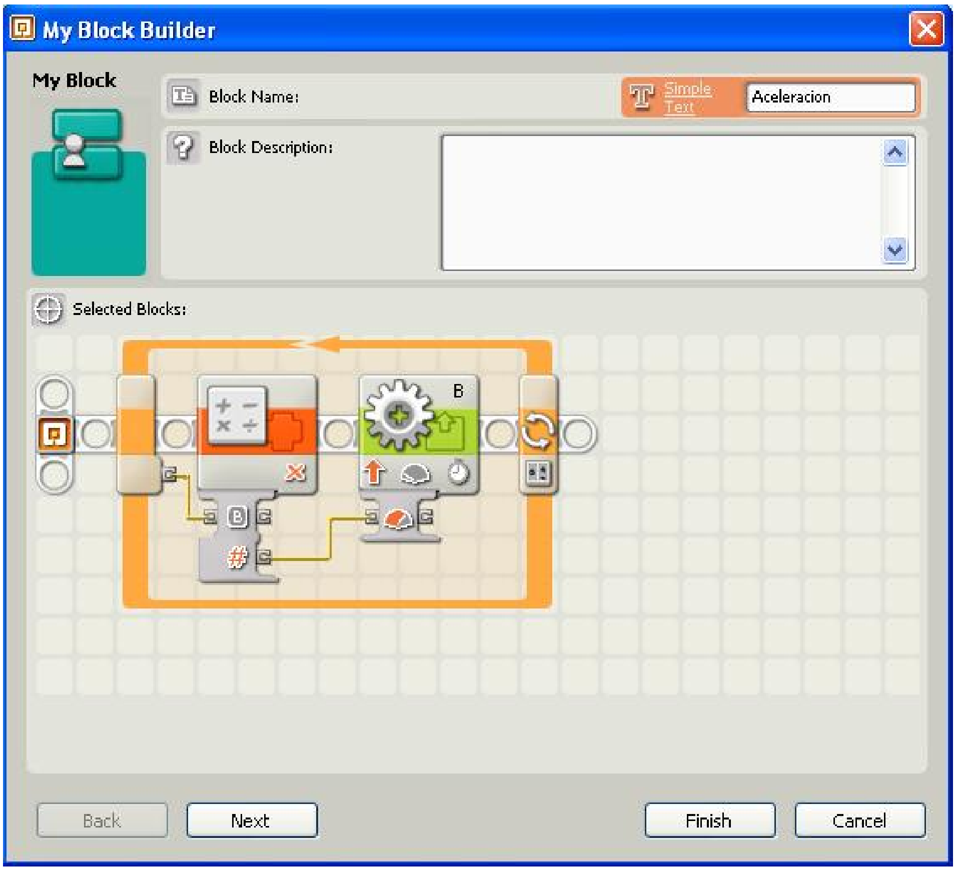
Datatype: Logic

Close

Creating and using *User Blocks:* User blocks are for the user (robot programmer) the means whereby to define procedures; from a didactical point of view, this can, also, be seen as a tool for abstraction on the part of the user (as we will explain it later on). Let us see how we can create a user block with an example. We have written a program that tells to our robot to go forward with a constant acceleration. The following NXT-G program has a loop where a variable is increasing at every step that produces a constant acceleration:



If we wish to transform this program into a new User Block: We select the entire program, and then we click "*New My Block*": we follow the instructions giving a name and assigning an icon to the new block. These "created blocks" are, in fact, procedures and can be defined both without and with parameters or arguments. Here is another example where a new block is defined with one input parameter and one output value (it is a block to convert numerical values into text values for displaying purposes):



My Block Builder

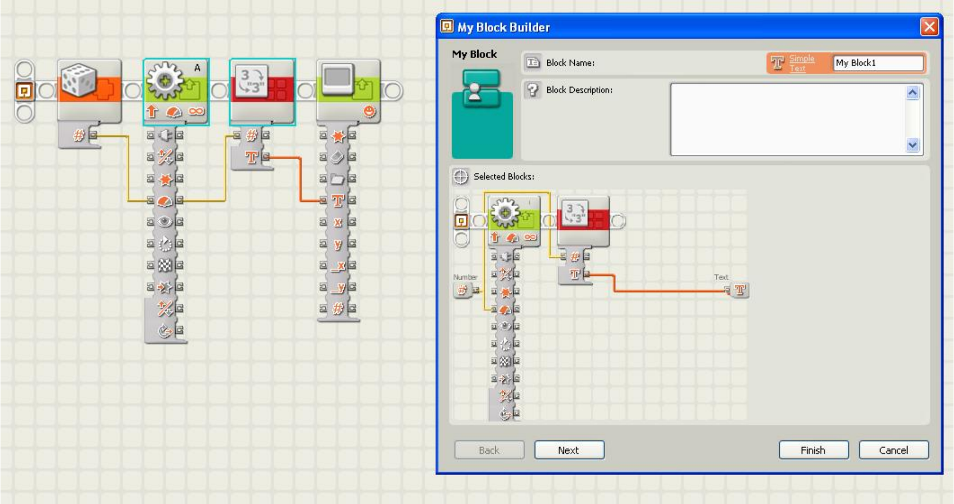
My Block

Block Name: Simple Text Aceleracion

Block Description:

Selected Blocks:

Back Next Finish Cancel



My Block Builder

My Block

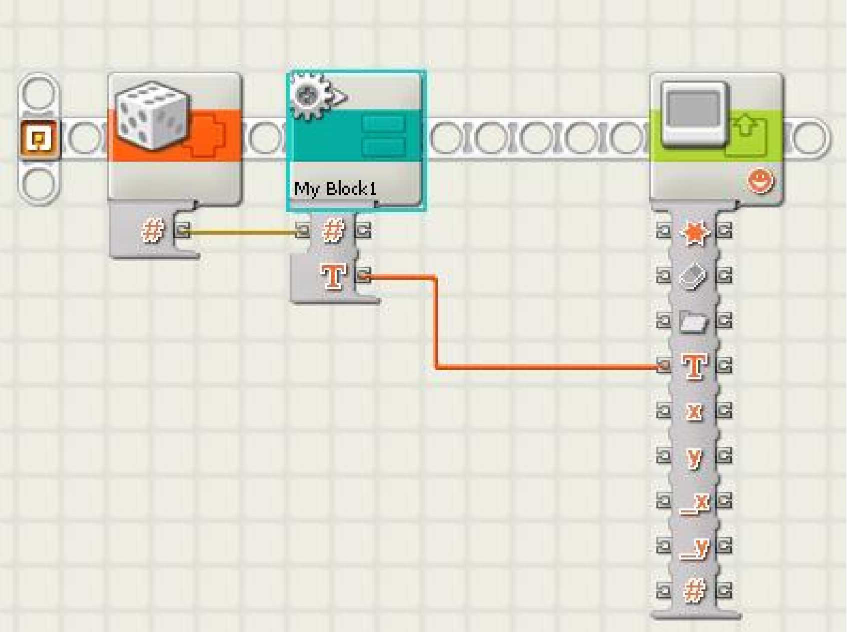
Block Name: Simple Text My Block1

Block Description:

Selected Blocks:

Back Next Finish Cancel

The new block can be used within a NXT-G program.



The problem 0 will focus on designing and building robots. The problems 1 to 6 will focus on the use of the *Move* blocks; the main interest is in understanding the relation between the 2 main parameters, i.e. power and duration (degrees/rotations or time) of this block and the speed (v), time (t) and displacement (x) of the robot (for example in cm/s and in cm):

2.3.3 Problem 0: How to assemble "good" straight line robots

a. Objectives/ Aims: Pupils should be able to

- Assemble simple straight line robots (with one or two Motors) using the educational Lego Mindstorms kits.
- "Copy" some of the models shown as examples and designed with LEGO DIGITAL DESIGNER
- Compare and asses the different robots built in the classroom by others from other projects. Criteria such as stiffness, stability, simplicity and aesthetics should be applied.

b. Previous knowledge required:

- Basic techniques/skills to assemble LEGO Mindstorms pieces:
    - Girders / beams assembly in parallel with pegs/ dowels
    - Girders/ beams cross assembly

  - Axles /axis of symmetry, friction rings, wheels
- Basic mechanical rules about:
  - Rigidity/ stiffness
  - Stability
  - Friction and adherence
- Simplicity and aesthetics criteria
- Basic user knowledge of LEGO DIGITAL DESIGNER software (http://ldd.lego.com)

Some basic skills are required to assemble the pieces and a previous recognition of the different kinds of pieces is, likewise, required. An initial step could be to take a robot, dismantle it and classify its different pieces.

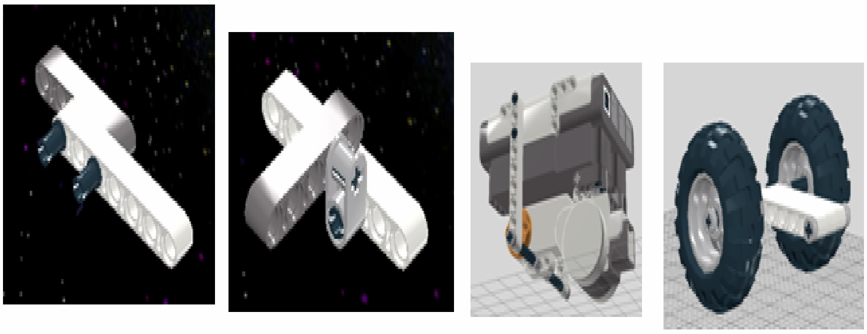
c. The teacher formulates a problem: *Are students able to build a simple straight line robot after observing the examples of the LEGO DIGITAL DESIGNER (LDD) application?* Conditions: a simple, stiff stable robot with one or two motors.

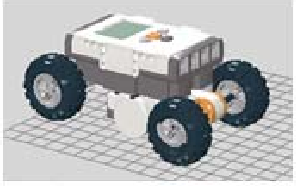
d. Students will build their own robots with the teacher's help. Most students will have their first contact with Lego Mindstorms kit, so the teacher will have to teach the basic assembly skills:

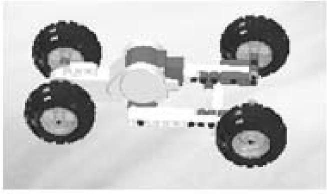
- Longitudinal assembly with pegs / dowels
- Girders cross assembly
- Axles assembly with rings and wheels

Next, students will build a complete robot, while teachers should stimulate them to build their personal robot, to obtain a certain variety and, then, analyse the different criteria of rigidity, stability etc.

Students can build robots as shown in the next images.





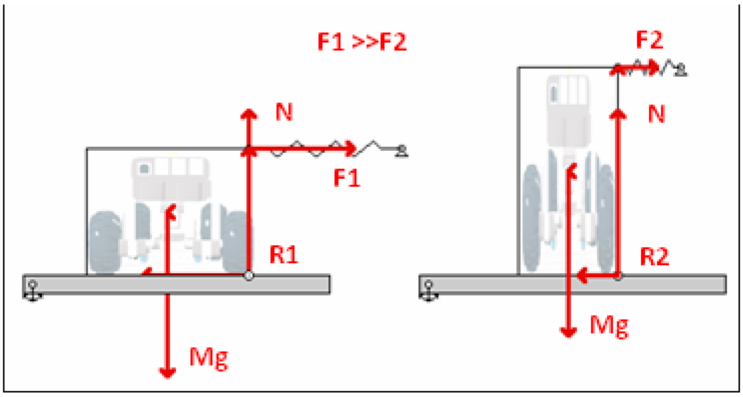


e. generalisations of local solution

Teachers will propose to students to test the quality of the robots, taking into account different aspects. The activity consists of testing the robots and discussing in groups what characteristics an ideal robot should have.

Some examples follow:

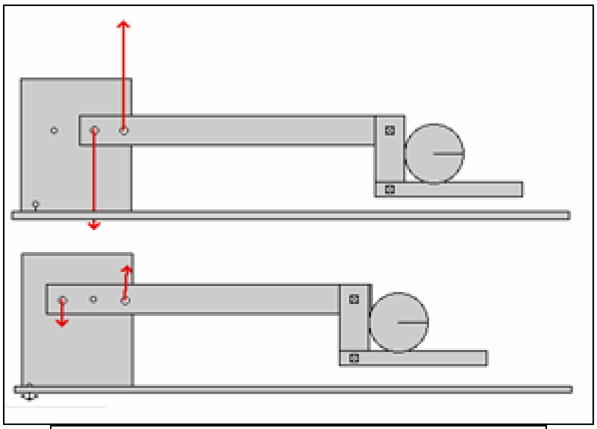
Stability: Every car will be pulled with a dynamometer from the height it corresponds to the top of it and, then, the force necessary to knock it over will be observed. The next image represents forces F1 and F2.

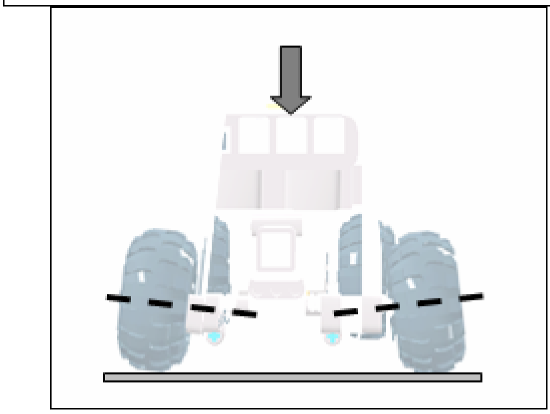


Both robots are of the same weight but of different dimensions (robot 1 is short and wide and robot 2 is high and narrow). It can be observed that both robots are of the same weigh, but robot 1 needs more force than robot 2 to be knocked down.

Rigidity / stiffness: the higher the tension of the pieces (axles/axes and pegs/ dowels), the less the rigidity of the robot. This is due to external forces like gravity and weights to be lifted by the robots.

The following image shows the different tensions in dowels/pegs assembling an arm, when they are exerted in contiguous or alternate holes. In the first case the tension is higher.





It is enough for rigidity testing to apply an external force on the robot (pushing down with a finger, for instance) and to observe if the structure gives way or gets loose.

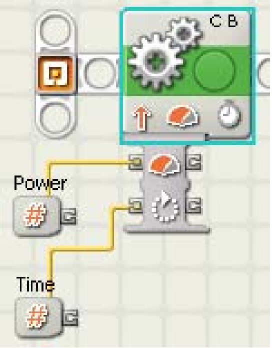
2.3.4 Problem 1: How to teach a simple robot to do a forward movement for a given interval t of time with a given Power P?

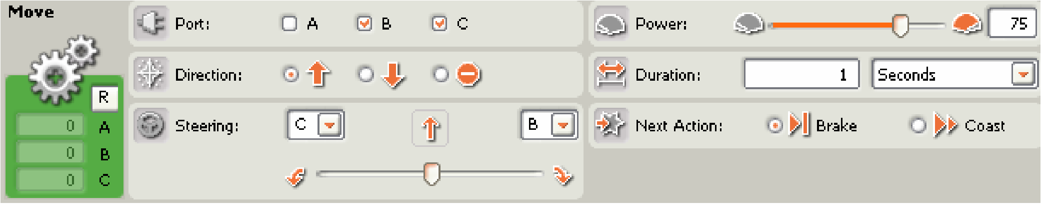
a. Aims: The pupils should learn how to write and use a procedure (my block) that allows the robot to make a lineal displacement for a given time t and with a given power P.

b. Previous knowledge required:

- To know how to build a simple robot.
- To know the basic features of NXT-G: to write a program and to download it to the robot.

c. Constructing the procedure *Move_PT (power, time):*





0	A
0	B
0	C

Port: □ A ☑ B ☑ C

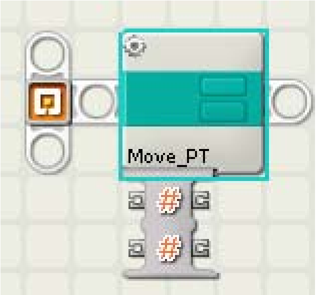
Direction: ● ↑ ○ ↓ ○

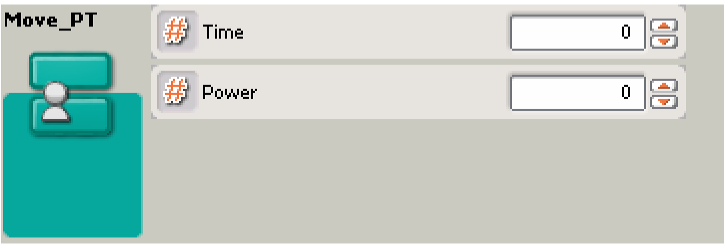
Steering: C B

Power: 75

Duration: 1 Seconds

Next Action: ● Brake ○ Coast





Time 0

Power 0

We construct one user block with 2 parameters: time and power. This means that we have one block with only 2 parameters, easier to use than the normal “Motor block”. With the new block we only have to define the power and the time (in seconds).

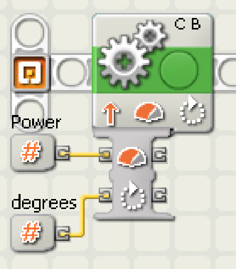
2.3.5 Problem 2: How to teach a simple robot to make a forward movement with a given power for a given angular distance φ (of the wheels)

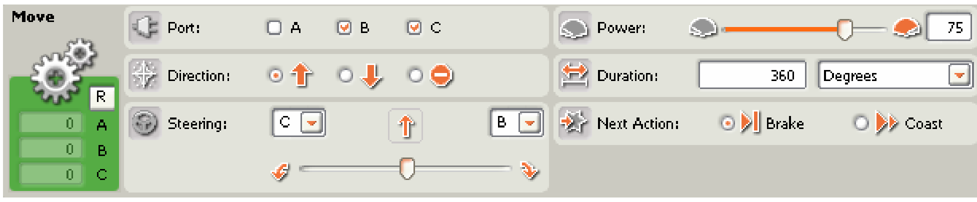
a. Aims: The pupils should be able to write and use a procedure to make a lineal displacement of the robot for a given value of *φ* degrees (of the wheels) of the Motor and a given Power P.

b. Previous knowledge required:

- Problem 1
- To know how to build a simple robot.
- To know the basic features of NXT-G: to write a program, to download it to the robot.

c. The procedure *Move_PD (power, degrees)*:




Move

R

0 A

0 B

0 C

Port: ☐ A ☑ B ☑ C

Direction: ● ↑ ○ ↓ ○ ⊖

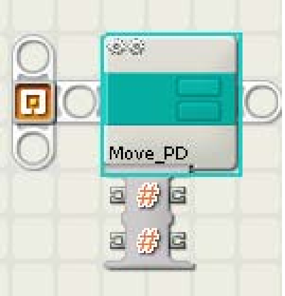
Steering: C B

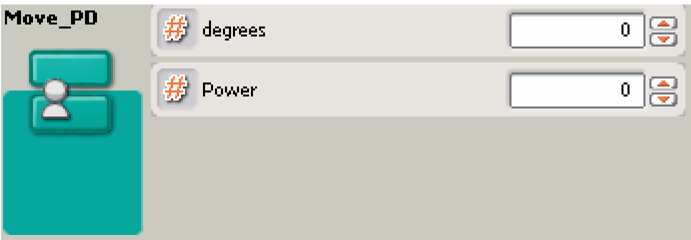
Power: 75

Duration: 360 Degrees

Next Action: ● Brake ○ Coast

We construct one user block with 2 parameters: angle and power. This means that we have one block with only 2 parameters, easier to use than the normal “Motor block”. With the new block we have only to define the power and the number of degrees (it means how long we rotate the wheels/motors of the robot); the result is to move forward (or backward) a given distance depending on the given angle.




Move_PD

degrees 0

Power 0

2.3.6 Problem 3: How to teach a simple robot to make a forward movement movement with a given power for a given distance X?

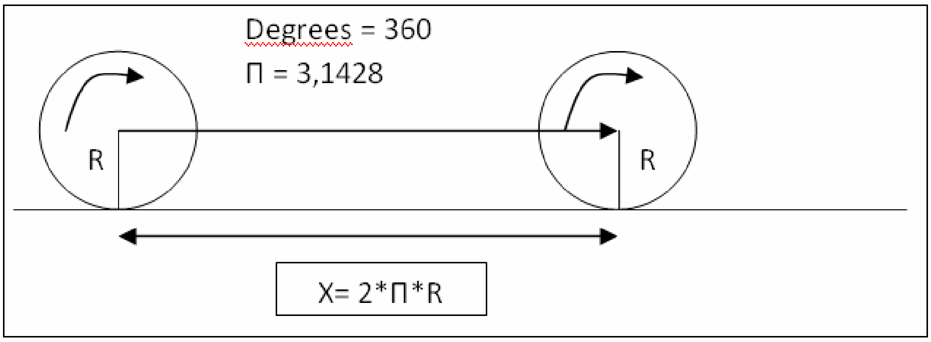
a. *Aims:* The pupils should

- learn how to create a procedure to convert a value X of distance in cm into the equivalent parameter “degrees” of the Motor block (angular distance)

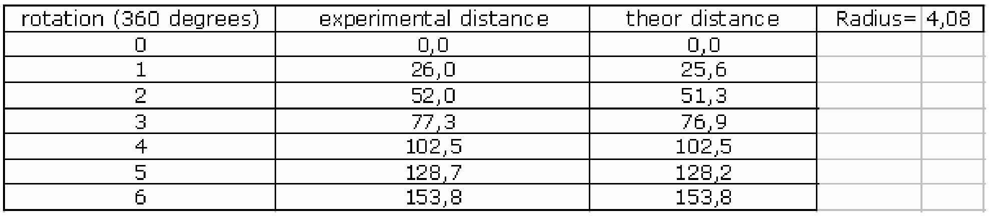
- be able to create a user block *Move_PX (power,x)***,** where the distance is given in cm.

b. To do so, we could use the already constructed block *Move_PD(power, degrees)*. We need to make the following transformation: OPERATION (x)→degrees, transforming angular distance into linear distance.

We should know that there is a relation between the radius R of the wheels/motors, the angular distance (rotation in degrees of the wheels/motor) and the linear distance (X). In the following figure we see that for 360 degrees of rotation, the linear distance X is given by the expression: 2*Π*R.



This means that we have the following relation: Angle/x = 360/2*Π*R and then we can deduce that Angle = (360*X)/(2*Π*R). The relation between angle and x (angular and linear distance, what we need to find) is: angle = (360/(2*Π*R)) X or angle = K*X where K = 360/2*Π*R = 14 (for R=4,08). Then we can make the following explorations:

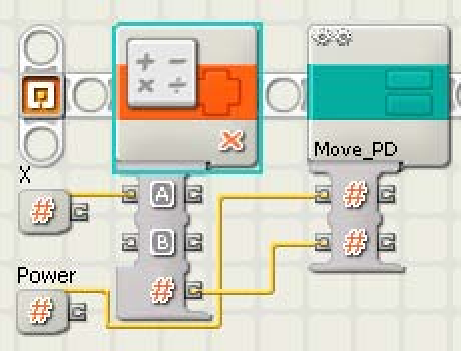


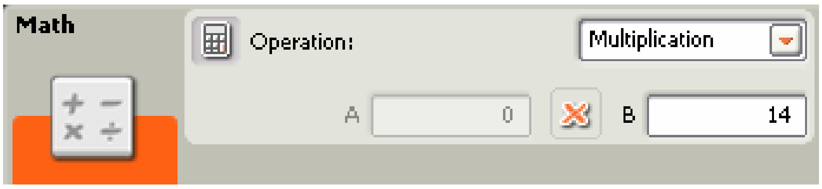
rotation (360 degrees)	experimental distance	theor distance	Radius=	4,08
0	0,0	0,0		
1	26,0	25,6		
2	52,0	51,3		
3	77,3	76,9		
4	102,5	102,5		
5	128,7	128,2		
6	153,8	153,8		

c. Previous Knowledge required:

- Problems 1, 2 and 3
- Experimental knowledge: X = 2*π*R* degrees/360
- Programming previous knowledge

d. The procedure *Move_PX (power, X)*

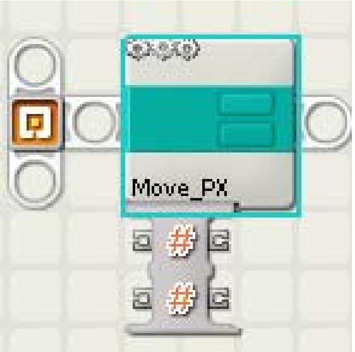


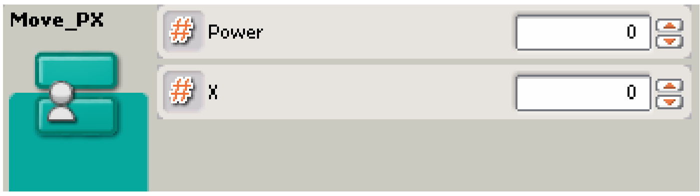


Math

Operation: Multiplication

A 0 B 14





Move_PX

Power 0

X 0

2.3.7 *Problem 4: How to teach a simple robot to make a forward movement with a given speed V for a given interval t of time?*

a. Aims: The pupils should learn

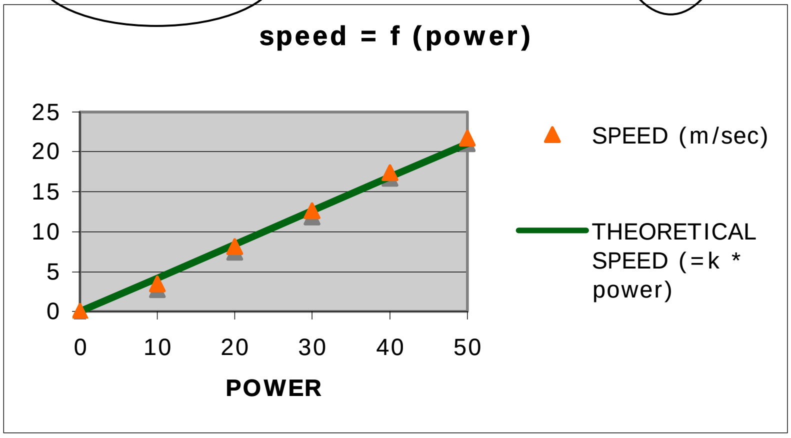
- To create a procedure to convert a value V, cm/s, into the equivalent of Power P (as it is defined in the Motor block).
- To create a block "*Move_VT (V, time)"*, where the velocity is given in cm/sec and time in seconds.

We can use the already constructed block *Move_PT (power, time)* and for that we need to make the following transformation: OPERATION V → power (transforming velocity into power values)

b. We do several experiments where we measure the distance travelled for 10 seconds with different power values. Doing so, we obtain the experimental speed. We observe in the following data table and graph that there is a relation (K) between Power and Speed:

V = 0,42*Power and Power = V/0,42 or Power = 2,38 * V

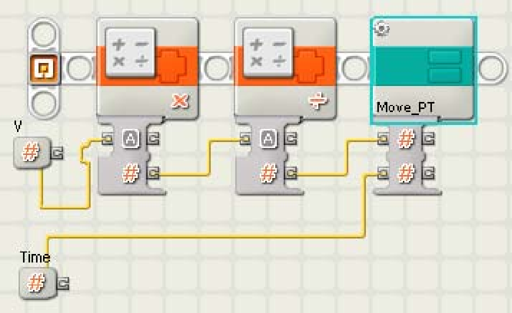
POWER (%)	DISTANCE (cm)	TIME (S)	SPEED (cm/sec)	THEORETICAL SPEED (=k * Power)
0	0	10	0	0
10	35	10	3,5	4,2
20	82	10	8,2	8,4
30	126	10	12,6	12,6
40	174	10	17,4	16,8
50	217	10	21,7	21
k=	**0,42**			

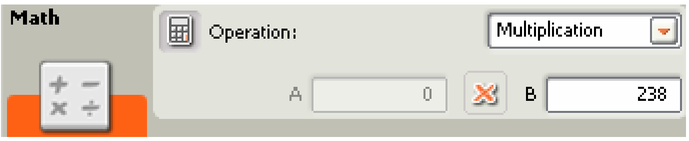


c. Previous Knowledge required:

- Problems 1 and 2
- Experimental knowledge: V = 0,42 * Power
- Procedure *Move_PT(power, time)*

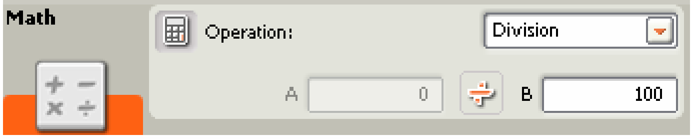
d. So the procedure *“Move_VT (v, time)”* is the following:




Math

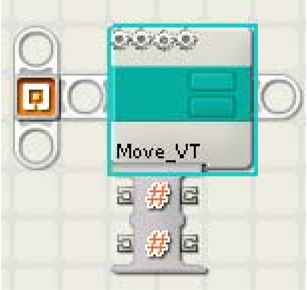
Operation: Multiplication

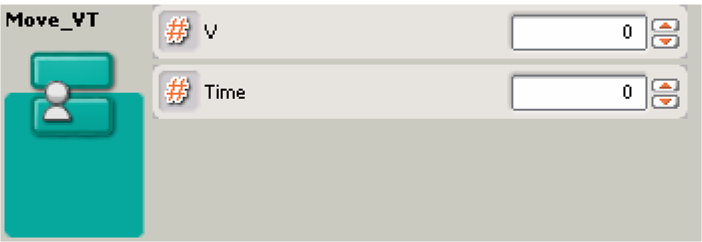
A 0 × B 238


Math

Operation: Division

A 0 ÷ B 100



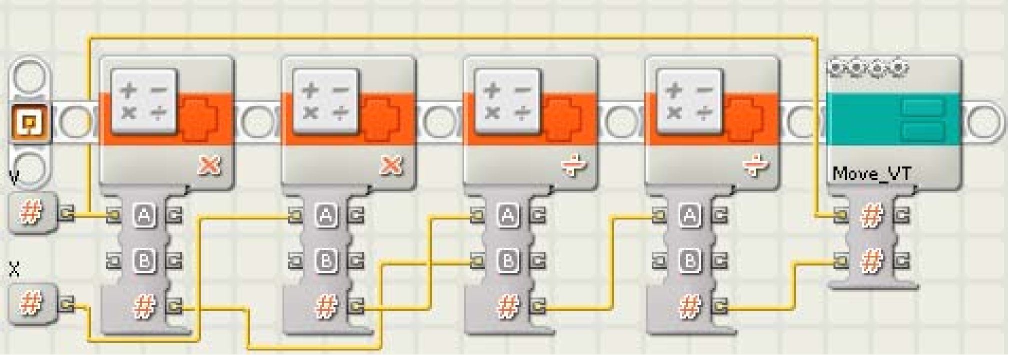

Move_VT

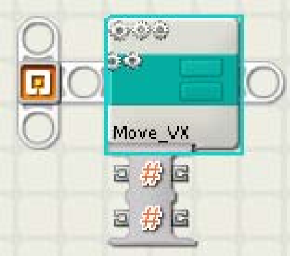
V 0

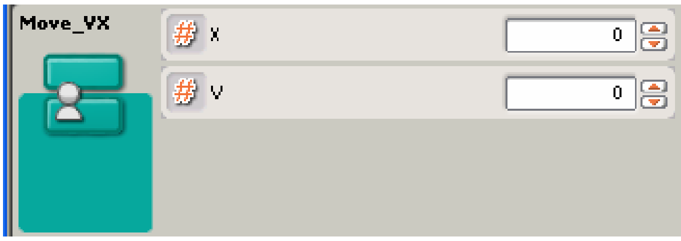
Time 0

2.3.8 Problems 5 and 6: How to teach a simple robot to make a forward movement with a given speed for a given distance X? How to teach a simple robot to make a forward movement for a given distance X during a given time t?

After all we have already done it is straightforward to "construct" the procedures *Move_VX (v, x)* and *Move_XT (x, time)* using the known relation V= X/T and the already developed procedure *Move_VT (v, time)*.

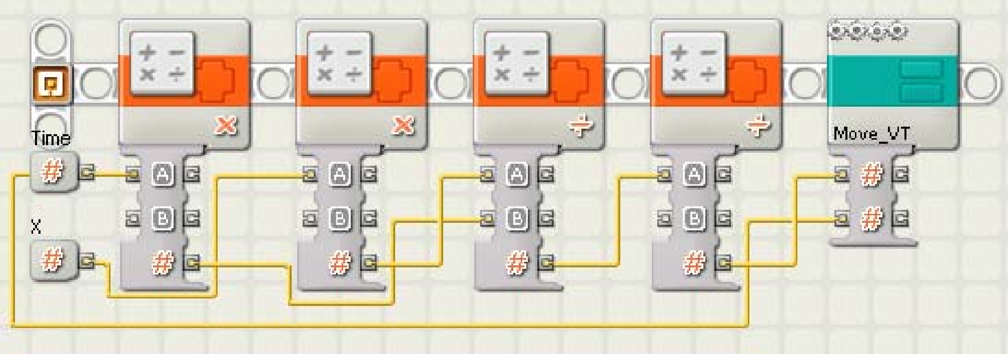


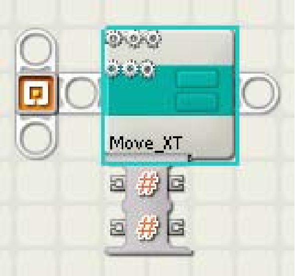


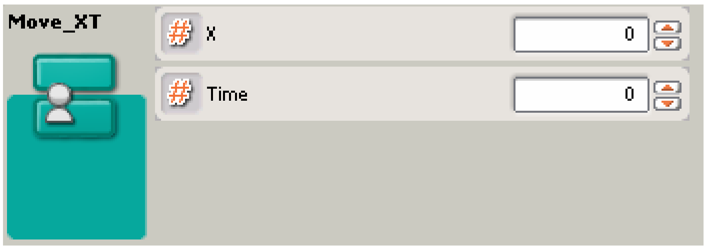

Move_VX

X 0

V 0







Move_XT

X 0

Time 0

We have now completed all the possible combinations of V, X and T to address the motion for a “straight line robot” constructing different procedures or blocks that realise these possibilities and solve all the sequence of relevant problems.

After solving these problems, we can discuss the following further questions (mostly to introduce the use of loops with the blocks):

- How to make a “long” forward displacement as a sequence of short forward displacements?
- How to make different successive short displacements for a given time and continuously longer every time?

2.4 Sensorized robots

2.4.1 Introduction

The main objectives (competences or skills to be acquired) behind the problems presented in this section are:

- The students have to know the features of the Lego Mindstorms NXT sensors (measured magnitudes, calibration, programming blocks, etc)
- The students should be able to design and build “good” robots, where the sensors are used and placed in the best possible places
- The students should be able to combine appropriately different programming techniques (Loops, If’s, variables, user blocks, etc….) with the use of one or more sensors.
- The students should be able to decide which sensors to use and how to program them in order to convert their robot into a reactive robot (as adaptive and autonomous as possible)

2.4.2 NXT-G necessary programming Features

To work on the proposed problems, we will use the following NXT-G features: Loops , if, variables, Myblocks, Miscellaneous blocks (arithmetical, logic operations, time etc). But first of all, let us see the different possibilities of the sensor blocks and the programming options.

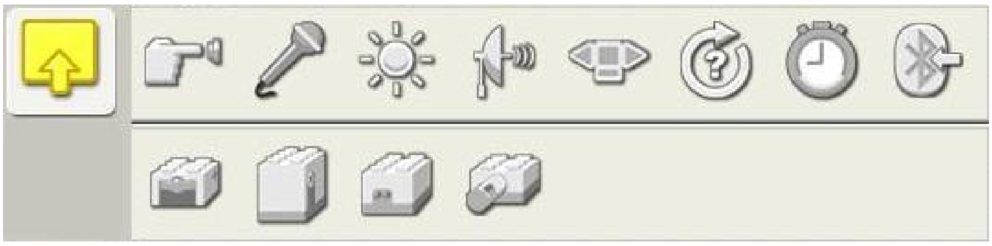


***Figure 2.4.1**: The Sensor group of blocks*

The sensor group (fig 2.4.1) has several blocks (one for each sensor). These blocks allow the user (the programmer) to measure the sensor value and, for example, to write it into a variable (fig 2.4.2):

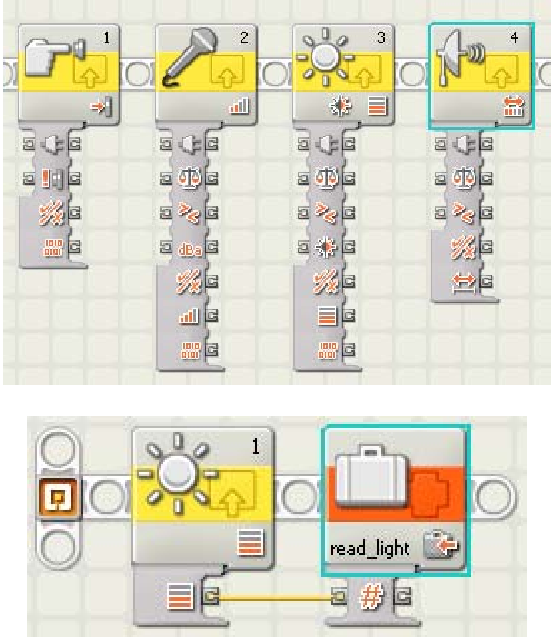
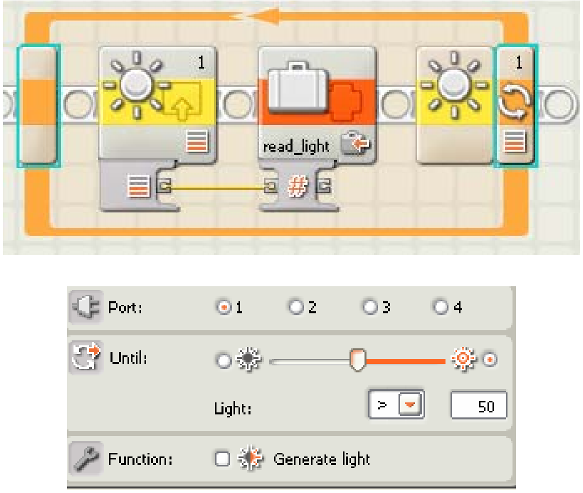


Figure 2.4.2: *Reading and Storing sensor values: Reading the sensor value of port 1 and storing it into the variable read_light.*

We can use also the sensors within a loop, as it is shown in the next figures:



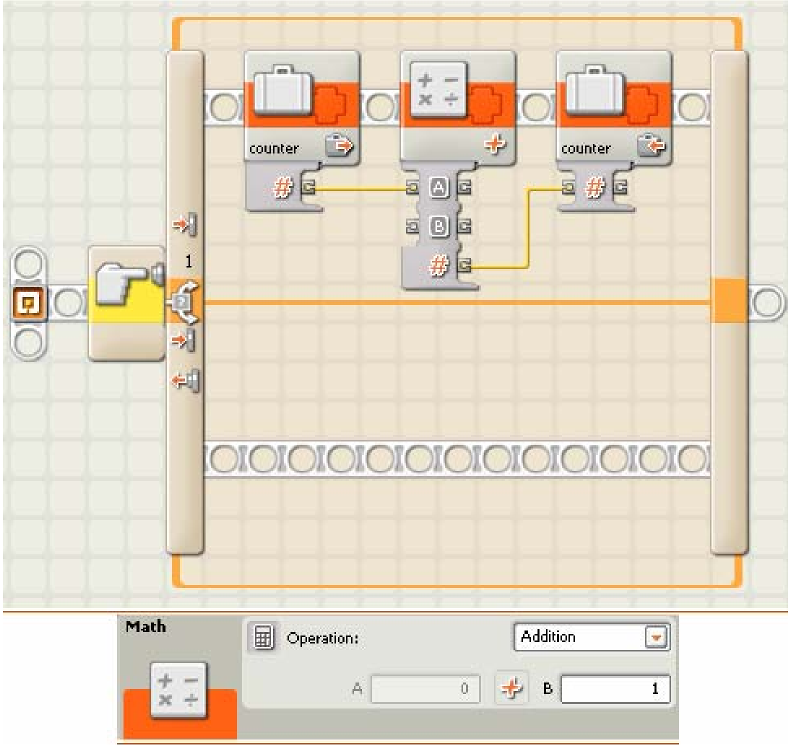
Port: ● 1 ○ 2 ○ 3 ○ 4

Until: ○ ●

Light: > 50

Function: □ Generate light

Figure 2.4.3: *Reading the sensor value of port 1 and storing it into the variable "read_light" until the read value of the light sensor (intensity) is higher than 50*



Operation: Addition

A 0

B 1

Figure 2.4.4: *Counting the number of times that the sensor touch is pressed*

2.4.3 The Tunnel problem

These problems aim at learning to work with the light sensor and to write a normal program (normally an infinite loop). The main motivation for the student is to propose to him/her to try to construct a mechanism for cars that make them possible to detect whether they need to switch lights on and whether they must switch lights off.

We propose the following sequence of steps:

a. To use the light sensor as a “photometer”, just to get to know how the sensor works.

b. To design a program which detects when the light is needed or not (depending on the light sensor values received) and switch the light on or off correspondingly.

For the first problem, we are going to use the *Try Me* option of the NXT brick with one light sensor directly connected with it. Then, we need to use one robot with 2 motors and the already acquired competencies about using the motors for straight-line motion and steering purposes. We will use a bulb actuator, conditional statements (if) and loops. We could formulate our problem as follows: “How can we construct an automatic device that could automatically switch on-off the light of our robot?"

The students should:

a. Try to understand how to interpret the light sensor values (intensity, reflected light, ambient light, etc)

b. Do that with the Try Me process, but also with a NXT-G program that can write the results on the display.

c. Understand the following combinations of blocks:

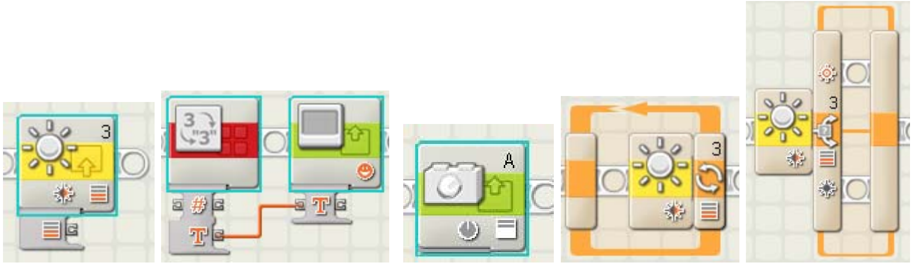


Figure 2.4.5*: NXT-G features needed to solve the problem*

The conclusion should be that it is possible to solve the problem for a given context as it is shown in the following figure, where the car/robot has to cross a tunnel and the goal is just to cross it turning the light on or off whenever needed. In order to generalize the use of such "security devices for cars/robots" we should integrate this "gadget" with others related to mobility (straight-line and turning motion) or to integrate other sensors in order to avoid obstacles and to be able to follow a path (surrounded by walls, for example).

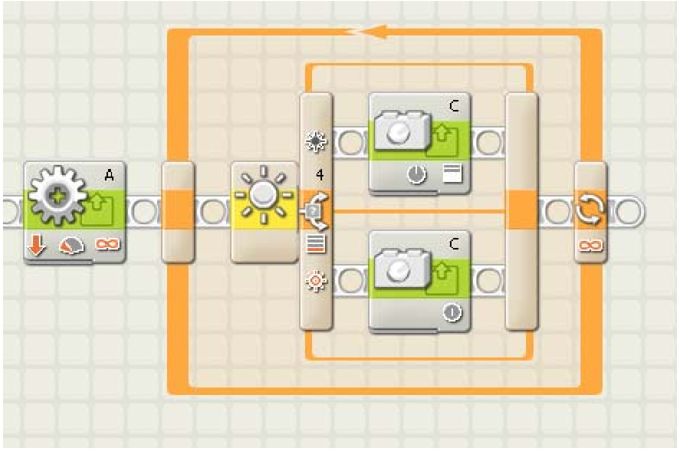


Figure 2.4.6: *a solution to the Tunnel Problem*

2.4.4 Counting black Lines on white background with a Light Sensor

The main objective is to use the light sensor to distinguish between black and white lines; this is an important feature in order to follow lines, or to keep the robot within a ring (a white one limited for example by a black line). In this case, we have to use a light sensor able to "read" light floor values and to react correspondingly.

We will use a mobile robot, for example the robot used for the tunnel problem. We need to place the new light sensor in the right position in order to read reflected light values from the floor. In fact the problem can be formulated as it follows: "How to count black stripes onto a white floor?"

We could undertake, for example, the following explorations: Calibrating our light sensor, counting one line, counting 2 lines, counting n lines, counting lines until some final condition is reached. We have to deal with the entire program setting in order to use the sensor, but, also, we need to store the "counting". We still need, as well, a kind of loop to "include" all our counting readings.

In the next figure we see one example of one program that counts n black stripes. This version of the program counts 3 black stripes (the external loop verifies this end condition). The robots go forward until he detects (with the light sensor) a black stripe. At this point it produces a sound and increments the value of a variable which is counting the number of black stripes and is directly linked with the loop.

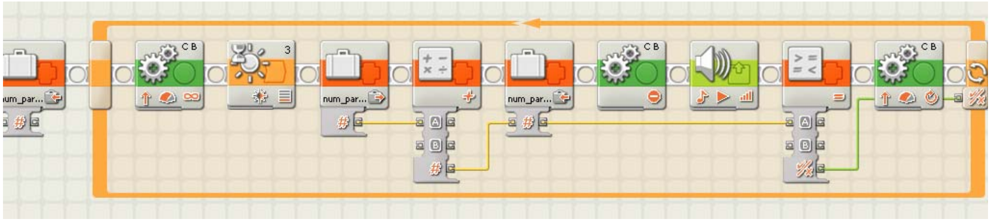


Figure 2.4.7: *a program counting a finite number of black stripes*

We can generalize this kind of programs: either finding an ending condition (for counting) or, maybe, making a "*MyBlock*" that counts a stripe every time it encounters one.

In any case, our robot is able to "read" black lines, which means it is able to detect if it goes out of or into a part of a circuit or similar, while, at the same time, this kind of behaviour (detection of black stripes) could be used to construct some kind of "language" (black on white) to tell the robot to do something special.

2.4.5 Detecting obstacles "My robot does not hit any obstacle"

In order to enable our robot to detect any obstacle before hitting it, we have to use either the ultra sonic sensor and/or the touch sensor. In the following examples, we suggest to use the commands "loop" and "switch", even if the "wait" command might also be used.

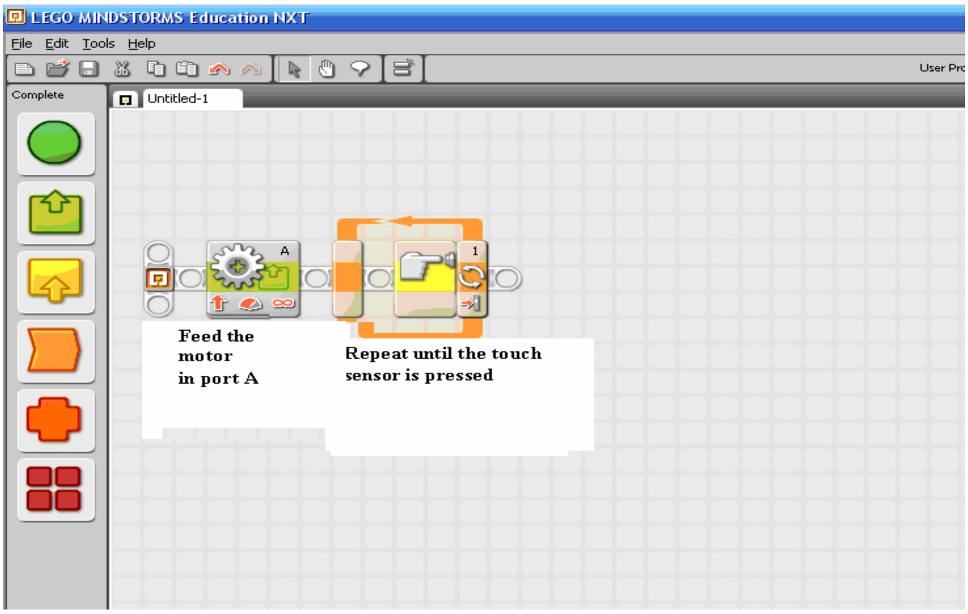
To solve this problem, students could use a touch sensor placed in the front part of the robot, which, when it is pressed, makes the robot stop. As you can see in the figure *2.6.a*, the program could be realized with two instructions:

- Feed the motor connected with port A
- Go on until the sensor connected with port 1 is pressed.
- After the last operation, stop the execution of the program

As you can observe in the figure 2.4.8 the command "move" is not posed inside the loop. We adopted this solution because, the command "move", if is posed in the loop, causes a discontinuity in feeding the motor every time the loop starts and the motor is turned on again. If the command "move" is outside the loop, the motor is turned on only once and then it is powered by the same value of power.

Motor (Port=A, Dir=FD, Pwr=75, PwrCtrl=OFF, Dur=FOREVER)

LP1:Loop(Ctrl=SENSOR, Sensor=TOUCH, Dis=OFF, Port=1,Act=PRESS) [LP1]



Feed the motor in port A

Repeat until the touch sensor is pressed

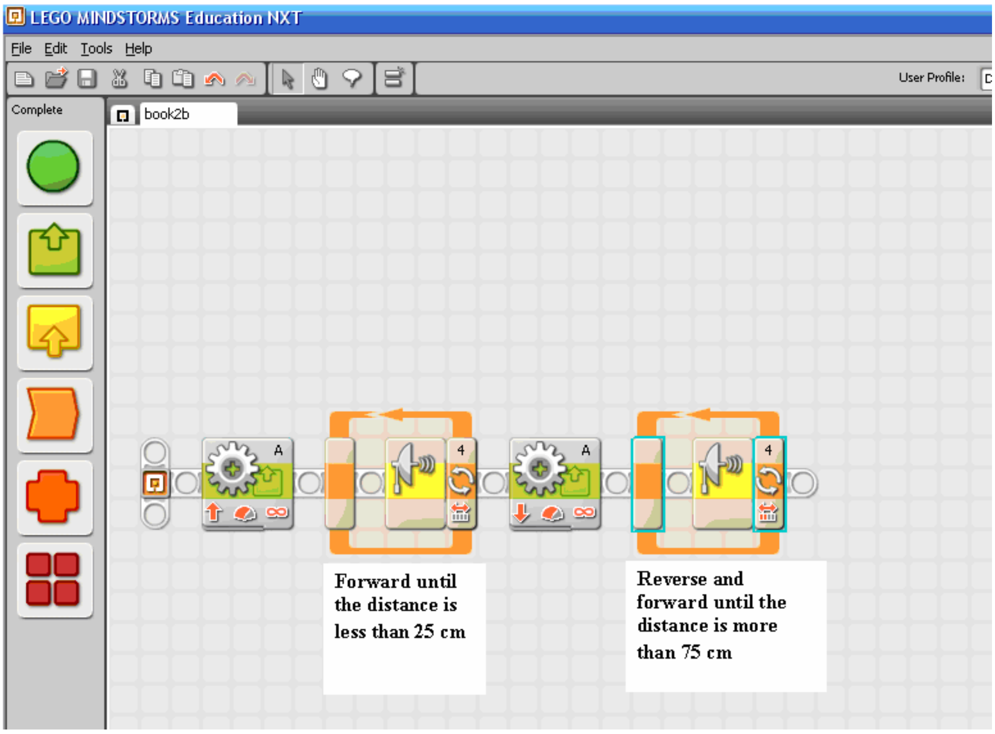
Figure 2.4.8: *"My robot does not hit any obstacle" program*

nt value is
old

The problem could be solved also by using an ultra sonic sensor and defining a threshold. The robot feeds the motor until the distance value is less than the threshold and then the execution of the program stops. The use of the light sensor is also interesting. In fact, this sensor can measure the light reflected by a surface. So students can use the sensor to "see" the light reflected by an obstacle.

The next step is to add another sensor to the robot, thus enabling it to reverse the sense of motion when it encounters an obstacle. In this way we can solve the following problem: "*a robot moving between two fixed points*". To solve this problem, students can use different kinds of sensors attached to different points of the robotic trolley. They can choose, for example, an ultra sonic sensor for the front side and a touch sensor for the backside of the robot, thus enabling it to move forward until the distance is lower than a fixed threshold, to reverse the motion and to go back until the touch sensor is pressed.

After solving the problem with the two sensors, the students should think about a solution of the same problem using only one distance sensor or a light sensor. The distance sensor could be placed in the front part of the robot, thus enabling it to move forward until the distance is lower than a fixed threshold, and then reverse the motor and go back until the distance exceeds another fixed value.



Forward until the distance is less than 25 cm

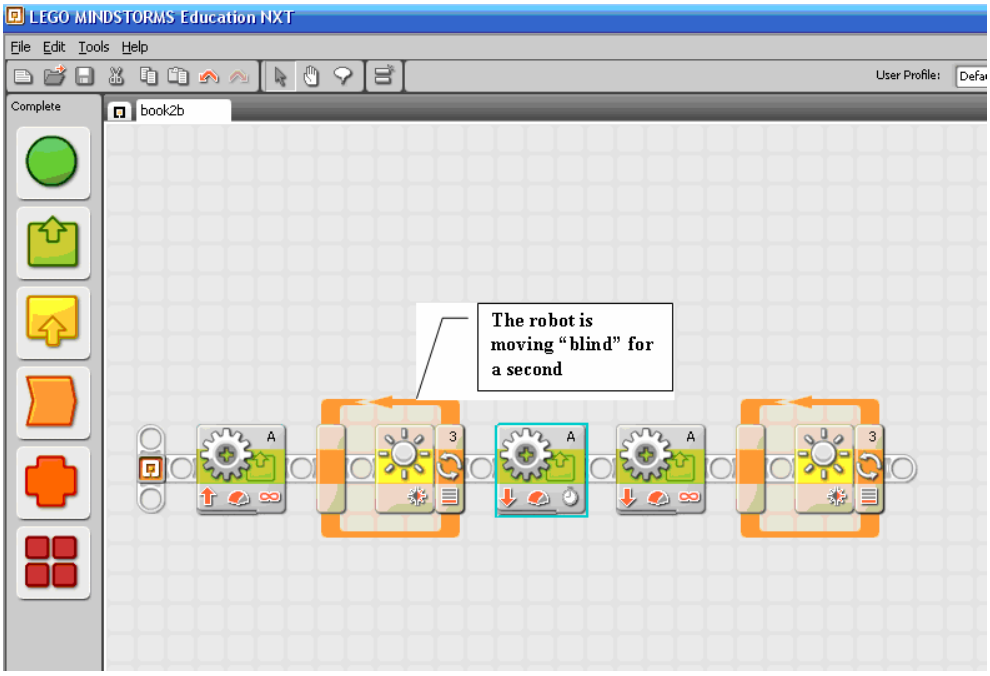
Reverse and forward until the distance is more than 75 cm

Figure 2.6.9 *Using only one distance sensor*

In figure 2.6.10 you can see a program for a robot moving on a white surface between two black lines. Note that, when the motor is reversed, the robot has to proceed for a brief step without interrogating the light sensor; in fact, because of its inertia, the trolley doesn't stop immediately, when it detects the change of light. So during the backward motion, it has to keep moving until it has moved away from the black line and then to start again detecting the light reflected by the surface behind it. Without the "blind" phase, the robot would detect immediately the same black line and remain on it. This is an example of another way to solve the same problem of the robot moving between two lines.

Another example of sensorized robot is the one which can change direction when it meets an obstacle. We can attach a distance or a touch sensor in the frontal part and program the robot to power the two motors at the same time on the same verse until the distance is lower than a fixed threshold or the touch sensor is pressed. Then the motors are reversed backwards for a short time, they are reversed forwards again with two different values of power for a second short time so that the robot moves avoiding the obstacle and finally it starts again the straight motion.

An interesting evolution of this problem is to program the robot to do "slalom" between some obstacles. It has to alternate turning right with turning left. The simplest solution is to add to the program an equal sequence of instructions and just change the turning direction in the last one.



The robot is moving "blind" for a second

Figure 2.6.10 *A program for a robot moving between two black lines*

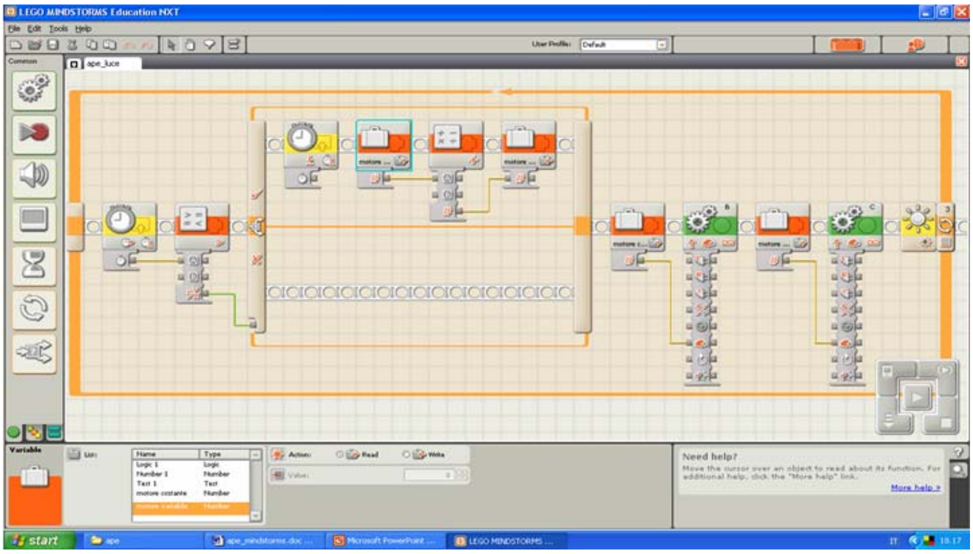
2.4.6 The Bee Problem

A particular example of complex turning robot is the “Bee robot”. The goal is to create a robot based on a strategy used by animals looking for food. The example was developed by the educational section of the Town Museum of Rovereto (Italy). Some researchers at the museum are interested in the behavior of insects. So, they projected an educational activity to study the bees simulating them by robots. The adopted strategy consists of tracking circular paths increasing periodically the radius until a flower is found.

In our activity the flowers are black spots fixed on white floor. We propose that students build the simplest robot they can imagine. So, the final model may be a robot with two-motorized wheels and a half sphere as a third touch with the floor. The most important problem is to translate the strategy in a program for robots. We can divide the problem in sub-problems:

- The robot has to move in circular motion
- The robot has to increase the radius of the circle
- The increasing of the radius must be a periodical action
- The robot has to be able to recognize a flower

We can solve the first problem feeding the motors in the same verse with different values of power. We can increase the radius increasing the power of the slower motor.



Name	Type
Logic 1	Logic
Number 1	Number
Text 1	Text
motore costante	Number
motore variabile	Number

Need help?
Move the cursor over an object to read about its function. For additional help, click the "More help" link.
More help »

***Figure 2.6.11** A solution for the bee- problem*

The most intuitive method to make this change periodical is to wait until a period is over and then to increase the value of the power for a constant quantity each period. So we have to save the actual value of the power on a variable which is increased every period. In order to “look for “the flowers, the robot can use a reflected light sensor pointed to the floor.

In the program shown in figure *2.6.11*, the two starting values for power are 80 and 40 percent. The robot moves, every 5 seconds the lower power is increased by 5 units and the process is repeated until the light sensor measures a value under the threshold of 40% of the saturation.

We can observe that the problem of tracking circles with regularly increased power could be an interesting topic for maths activities; in fact, it might be useful to study methods to approximate the spiral by circular arches of increasing radius. Consequentially, this kind of robot-enhanced activities might be useful to study general methods whereby to approximate mathematical curves.

2.5 Turning robots

2.5.1 Introduction

Linear robots are the simplest constructions that permit the analysis of basic rules of motion, fundamental control commands and how to integrate sensors to make the robot able to react to stimuli coming from the environment. In spite of their simplicity, they offer a didactic platform to investigate many fundamental concepts and relations, such as space, time, rotation, linear and angular velocity, acceleration, direct and inverse proportionality and others. But robots exploit all their flexibility when moving on a 2-D plane: pupils can design interesting strategies for avoiding obstacles and reaching target positions, use effectively all the information coming from the environment through sensors, reproduce behaviours more similar to those of the every day life. Therefore, the following step of development regards turning robots. With this term we speak about robots able both to move on a straight line and to perform nonlinear trajectories.

The first idea could be to build robots similar to cars or similar to live beings. Unfortunately these types of robots are excessively complex. Cars require critical mechanical subsystems to turn the steering wheels (steering-gear) and to reduce the lateral friction of the non-steering wheels (differential-gear): both such subsystems are very complex and do not add any interesting educational aspects, apart from their mechanical properties. Robots that emulate natural behaviours require several degrees of freedom, that is, several joints and motors, a structure hard enough to be made stable and very complex motions. If this last type of robots is of interest, it is advisable to use already built robots or kits specifically designed for it ('animaloids' or 'humanoids').

With Lego Mindstorms NXT, a turning robot can be more easily built by connecting one motor to each one of a couple of independent wheels and adding one or two free wheels to the robot to obtain a sufficient stability and to permit simultaneous application of different powers to the two motors. This produces different angular speeds for the two drive-wheels and, therefore, the robot can follow nonlinear trajectories with a limited friction (similarly to trolleys used in supermarkets). A free wheel can be also substituted by a sphere free to rotate in any direction within its seat (think of an old computer mouse).

2.5.2 General remarks and theory

Like in the case of linear robots, we are interested in the general potentialities of the class of robots that we call 'turning robots'. As mentioned previously, we consider the most significant aspects of the robots of this class to have two independently motorized wheels and the possibility to turn. Several constructions could exhibit these properties (fig. 2.5.1). In the following text, we will focus attention only on the motorized wheels.

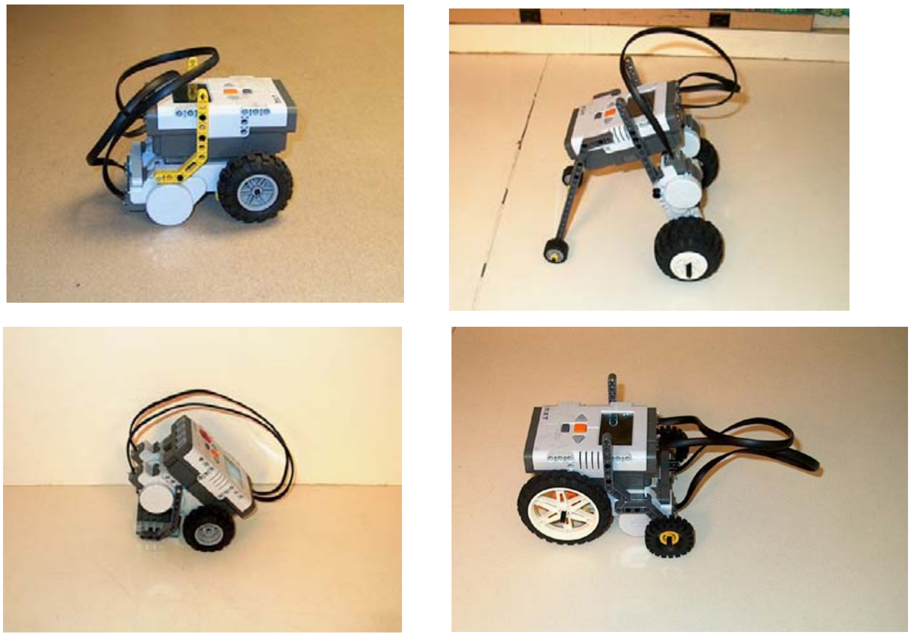


***Fig. 2.5.1** – Turning robots*

Considering positive a clockwise rotation of the wheels and supposing that this rotation makes the robot move forward, if wheel$_1$ has an angular speed ω_1 and wheel$_2$ has an angular speed ω_2, with $\omega_2 \geq \omega_1$ measured in rad/s, it can be easily verified that the robot turns right and the two wheels draw two concentric circular trajectories (fig. 2.5.2). Let r be the common radius of the wheels, d the (fixed) distance between the two wheels, in a short motion of Δt time units the two wheels track an arc of circle long respectively $\Delta A_1 = \omega_1\, r\, \Delta t$ and $\Delta A_2 = \omega_2\, r\, \Delta t$. Let R be the distance from wheel$_1$ and the hypothetic centre of the two arcs of circle, and $\Delta\Theta$ the angle corresponding to the two arcs, it results in:

$$\Delta A_1 = \omega_1 \text{ r } \Delta t = \text{r } \Delta\theta_1 = \text{R } \Delta\Theta \qquad (2.5.1)$$

$$\Delta A_2 = \omega_2 \text{ r } \Delta t = \text{r } \Delta\theta_2 = (\text{R+d})\ \Delta\Theta \qquad (2.5.2)$$

and therefore:

$$\Delta\Theta = \omega_1 \text{ r } \Delta t / \text{R} = \omega_2 \text{ r } \Delta t / (\text{R+d}) \qquad (2.5.3)$$

$$\Omega = \Delta\Theta / \Delta t = \text{(the ‘angular speed’ of the robot)} =$$

$$= \omega_1\, r / R = \omega_2\, r / (R+d) \qquad (2.5.4)$$

$$\omega_1 / R = \omega_2 / (R+d) \qquad (2.5.5)$$

$$R = d\, \omega_1 / (\omega_2 - \omega_1) \qquad (2.5.6)$$

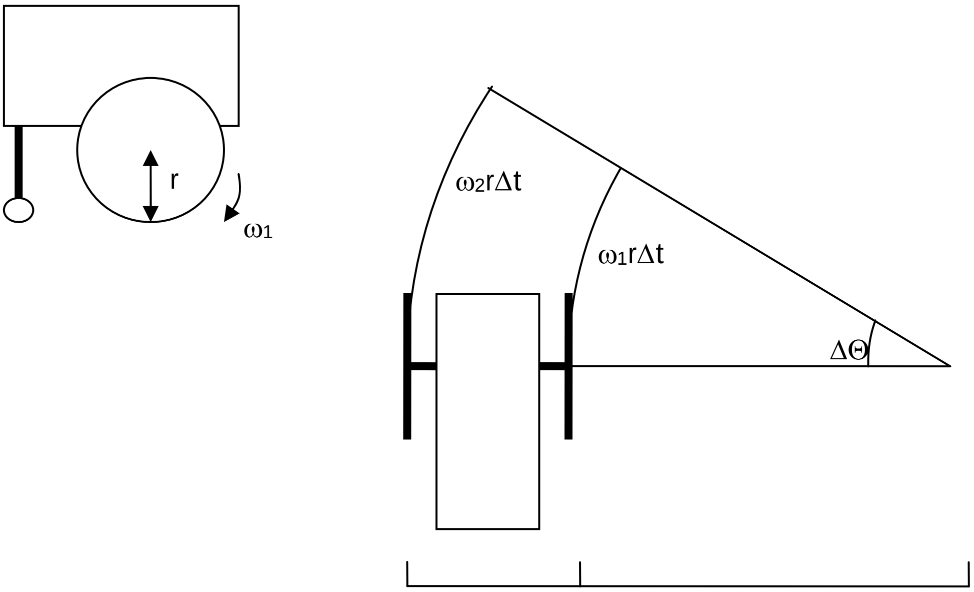


Fig. 2.5.2 *– Turning motion*

Therefore, the radius R does not depend on the radius r of the wheels, but only on the angular speeds and the inter-wheel distance. When $\omega_2 \to \omega_1$, then $R \to\infty$, the motion is straight and the equation (2.5.1) becomes simply $\Delta A = \omega\ r\ \Delta t = r\ \Delta\theta$. When ω_1=0 then R=0 and the robot pivots around wheel$_1$ that is not moving. When $\omega_2 = - \omega_1$ ($\omega_2 > 0$), it results in:

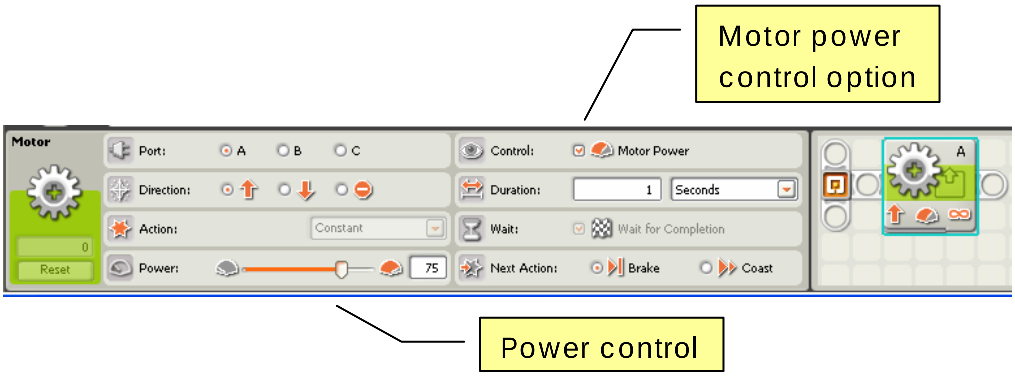
$$R = - d /2 \qquad (2.5.7)$$

and the robot pivots around the middle point between the two wheels, regardless of the value of the applied power. The angular speed of and the angle performed by the robot (ignoring the sign) are in this case:

$$\Omega = 2\, |\omega_1|\, r / d \qquad (2.5.8)$$

$$\Theta = |\theta_1|\, r / R = 2\, |\theta_1|\, r / d \qquad (2.5.9)$$

Now the problem is to provide suitable commands to impose the two ω_1 and ω_2 angular speeds required by the desirable trajectory. First of all, we recall that, for a straight motion with one motor, the NXT-G Motor command actually provides a speed control, when the ‘Motor power control’ option is active (fig. 2.5.3).



Motor power control option

Motor

0

Reset

Port: ● A ○ B ○ C

Direction: ● ○ ○

Action: Constant

Power: 75

Control: ☑ Motor Power

Duration: 1 Seconds

Wait: ☑ Wait for Completion

Next Action: ● Brake ○ Coast

Power control

Fig. 2.5.3 *– The Motor block*

Empirically, it is easily possible to determine the relation between speed and power. We did this with a program that samples many times the rotation sensor integrated in the motor applying different powers with the 7.5 V rechargeable battery (for a description of this technique see section 2.5.7.1). Supposing that a negligible load is applied to the motor, this relation appears linear (fig. 2.5.4).

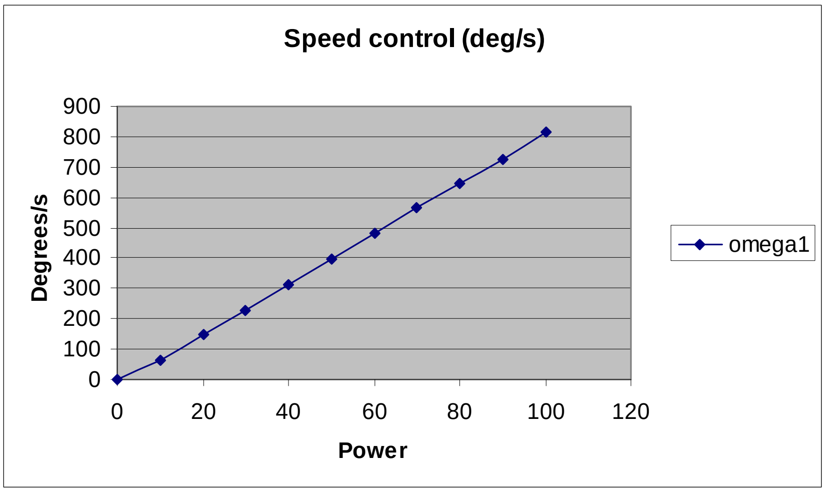


Fig. 2.5.4 *– Angular speed versus Power*

Let $\omega = k_{\omega P}\, P$ be the relation with P the applied power in percentage, as usually indicated: the constant of proportionality $k_{\omega P}$ has been determined as about 8.15 degrees/s, in the following formula:

$\omega = k_{\omega P}\, P = 8.15\, P$ deg/s

Given that 1 deg = $2\pi/360$ rad and 1 deg/s = 1/6 rpm,

it results in $\omega = 0.1422\, P$ rad/s = $1.36\, P$ rpm (2.5.10)

From (2.5.6) we obtain also:

$R = d\, \omega_1 / (\omega_2 - \omega_1) = d\, P_1 / (P_2 - P_1)$ (2.5.11)

With a 9V battery $k_{\omega P} > 9.5$ degrees/s: thus, the actual constant depends heavily on the battery charge: it is advisable to replicate the analysis with the suggested method so that the teacher (and the students) can evaluate this constant with their own robots. With a significant load, the linearity in the relation speed-power is limited to the lower powers, whereas for higher powers and high load the control saturates and from one point to the maximum of power the speed remains almost constant.

With two motors, the first possibility is to provide separate commands to them with two successive Motor blocks, the first one necessarily not waiting for completion, in order to leave a very short (and negligible) time separation between the two commands.

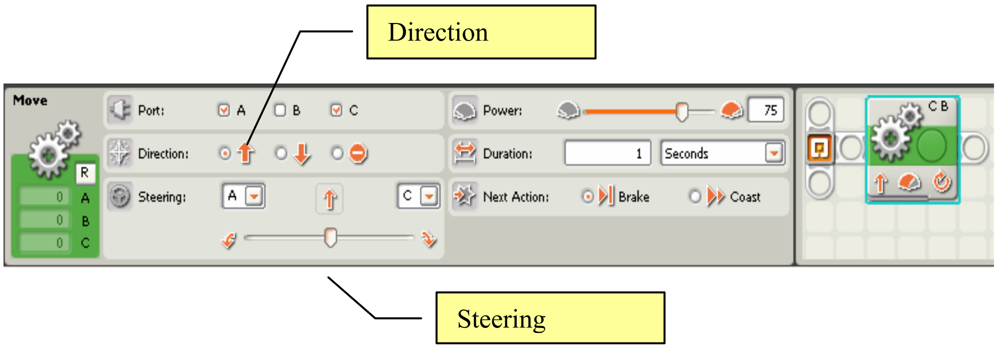
Just to have a quantitative idea of the turning motion, taking from equation (2.5.10) 0.1422*P rad/s as the valid formula giving the angular speed with a rechargeable battery, mounting the standard intermediate size wheels of the NXT kit which have a diameter of $2r$=56 mm, and supposing to want to turn right with a radius R=100 mm and a distance d=70 mm, every second the robot performs:

- An internal arc of radius 100 mm and length of $\theta_1 = 28\, \omega_1$ mm (angular speed in rad/s) = $3.98{*}P_1$ mm
- An external arc of radius 170 mm and length of $\theta_2 = 28\, \omega_2$ mm (angular speed in rad/s) = $3.98{*}P_2$ mm
- It must hold: $P_1 / (P_2 - P_1) = R/d = 10/7 \quad \Rightarrow \quad P_2 = 1.7\, P_1$

Setting for example $P_1 = 30$ and $P_2 = 1.7\, P_1 = 51$, in 3 seconds the robot draws two arcs with:

$A_1 = 3.98{*}30{*}3 = 358.2$ mm $\qquad A_2 = 3.98{*}51{*}3 = 608.9$ mm

There is a second possibility integrated in the NXT-G Move block (fig. 2.5.5): this block permits to control concurrently two motors with a further parameter called ‘steering’ that you can set through a specific slider or, as usual, connecting an input data wire. This parameter controls the ratio between the angular speeds of the two motors, but unfortunately such relation is not well documented and it must be analyzed empirically. Let us assume that, when the Direction parameter in the Move block is set to straight forward, i.e. up arrow, in case of a straight motion the angular speed of the motors is positive. It is known that the steering parameter can vary from -100 and +100: 0 corresponds to a straight motion ($\omega_1 = \omega_2 > 0$), negative values are set to steer towards left motor ($\omega_1 > \omega_2$), positive values to steer towards right motor ($\omega_1 < \omega_2$). With the extreme values the robot pivots around the middle point ($\omega_2 = -\omega_1$, $\omega_1 > 0$ with steering=-100, $\omega_1 < 0$ with steering=+100).



Move

0 A
0 B
0 C

Port: ☑ A □ B ☑ C

Direction: ● ○ ○

Steering: A C

Power: 75

Duration: 1 Seconds

Next Action: ● Brake ○ Coast

Fig. 2.5.5 – The Move block

To study the intermediate values, we initially prepared an observation experiment with a robot provided with a pen to draw on a paper sheet during the motion (fig. 2.5.6). Then we measured the radius of the drawn circles with different steering values, obtaining the results in fig. 2.5.7.

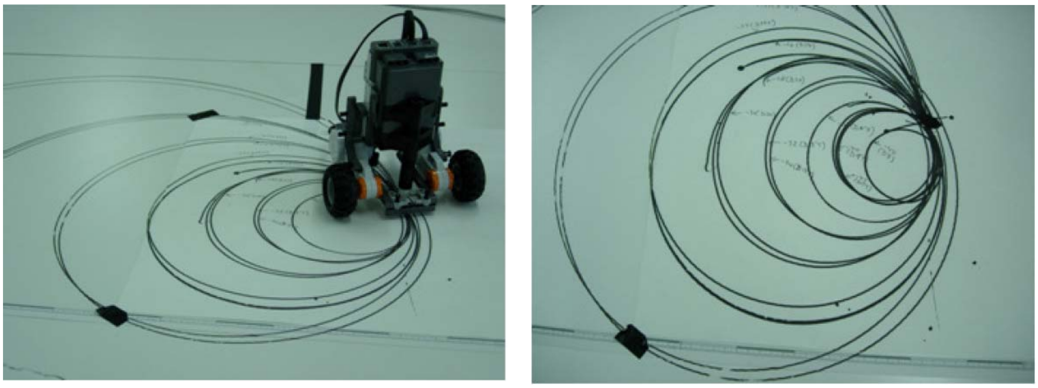


***Fig. 2.5.6** – The steering parameter*

The imprecision of this first approach counseled us to prepare an experiment similar to the previous one, used to measure the speed-power relation, to evaluate the average angular speed obtained with a fixed power (P=70) and a steering value varying from 0 to +100 (see section 2.5.7.2). The result is summarized in fig. 2.5.8.

As you can see in fig. 2.5.8, increasing the steering maintains almost constant ω_2 and decreases linearly ω_1 untill $\omega_1 \approx - \omega_2$. The ratio $R/d = \omega_1 / (\omega_2 - \omega_1)$ decreases apparently in inverse proportion in respect of the steering value, as shown in fig. 2.5.9.

Steering	Diameter	Radius
44	70	35
42	95	47,5
40	100	50
38	115	57,5
36	135	67,5
34	155	77,5
32	175	87,5
30	200	100
28	240	120
26	270	135
24	310	155
22	365	182,5
20	410	205
18	470	235
16	540	270
14	680	340
12	750	375
10	1140	570

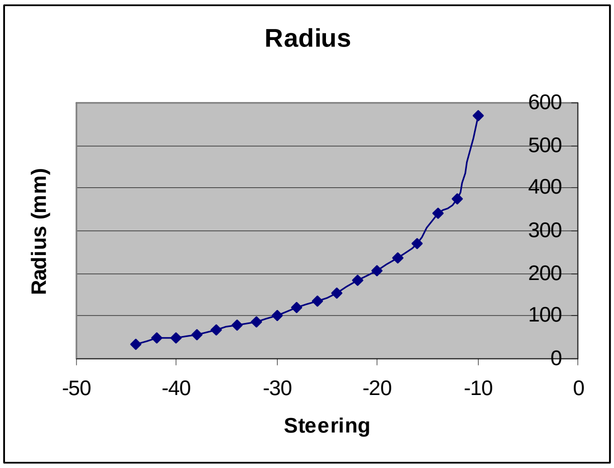


***Fig. 2.5.7** – Relation between Steering and Radius*

Steer.	ω_1	ω_2	ω_1 / $(\omega_2 - \omega_1)$	(-st+40) /st
0	564,7239	564,2873	-1293,5	
10	419,4042	555,563	3,080258	3
20	283,9617	554,582	1,0493	1
30	141,6721	554,0275	0,343568	0,333333
40	4,354432	552,7787	0,00794	0
50	-122,522	552,9425	-0,18139	-0,2
60	-277,014	552,4995	-0,33395	-0,33333
70	-414,975	551,408	-0,42941	-0,42857
80	-551,589	551,1325	-0,50021	-0,5
90	-574,262	550,4984	-0,51056	-0,55556
100	-573,674	550,3165	-0,51039	-0,6

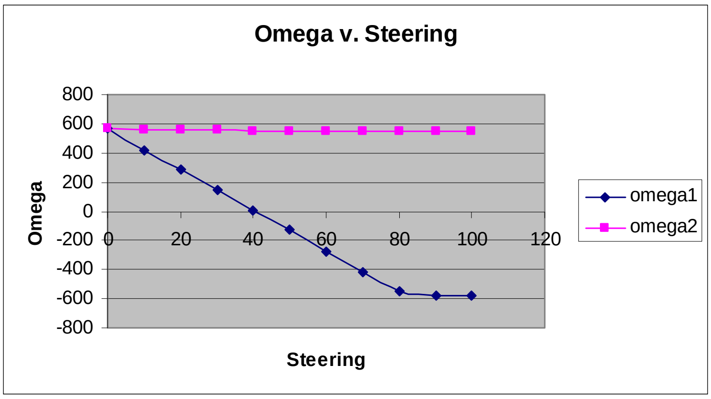


***Fig. 2.5.8** – Relation between angular speeds and steering*

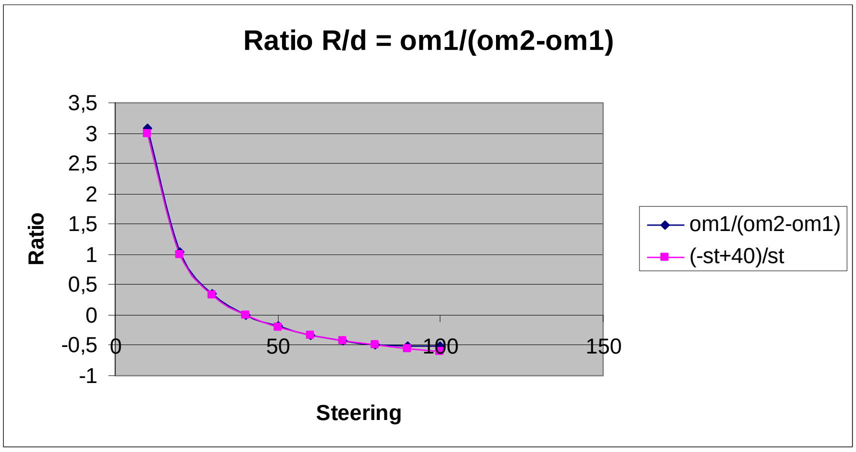


***Fig. 2.5.9** – Relation between R/d and steering*

In fact, assuming the following (S=steering):

$\omega_1 = k_1\ S + k_2$ (2.5.12)

$\omega_2 = k_3$ (2.5.13)

$R/d = \omega_1 / (\omega_2 - \omega_1) = (k_1\ S + k_2) / (k_3 - k_1\ S - k_2) =$

$= (S + k_A) / (-S + k_B)$ (2.5.14)

The estimation of these coefficients from the experimental data gives:

$\omega_1 = -13.9\ S + 558$ (2.5.15)

$\omega_2 = 552$ (2.5.16)

$k_A = k_2 / k_1 = (558 / -13.9) = -40$ (2.5.17)

$k_B = (k_3 - k_2) / k_1 = ((552 – 558)/ -13.9) \approx 0$ (2.5.18)

$R/d = (-S + 40) / S$ (2.5.19)

and the plot is an arc of an hyperbola with the vertical asymptote *S*=0 and the horizontal asymptote *R*/*d*=-1. To confirm the last result, in fig. 2.5.9 the obtained function is plotted together with the experimental data.

To conclude, even though the presented inspection made possible to evaluate the trajectory performed using a Move block with a given steering, it is even simpler to control the turning robot with two Motor blocks.

2.5.3 Didactical consideration

The following table summarizes the main problems that can be put when analyzing a turning robot.

	Problem	Pseudo-code of the action	Comments
1	How to build a robust turning robot?		
2	How to make the robot move forward with a given speed for a certain time?	GoForward (speed, time)	Just as for linear robots, the linear speed depends on the radius of the wheels. The power to be applied to the motors (i.e. the angular speed of the two wheels) is the same
3	How to make the robot move forward for a given distance with a given speed?	MakeStep (distance, speed)	Just as for linear robots, from the given parameters you should calculate the angle to be performed by both motors and the power to be applied to them.

4	How to make the robot pivot with a given angular speed Ω for a time t?	Pivot(Ω, t)	We know that for the robot angular speed it holds $\Omega=2\,\omega\, r / d \Rightarrow$ $\omega= \Omega\, d / (2\, r)$ $P = \omega / k_{\omega P} =$ $= \Omega\, d / (2\, r\, k_{\omega P})$
5	How to make the robot pivot for a given angle Θ with a given angular speed Ω?	Pivot(Θ, Ω)	We know that for the angle made by the robot it holds: $\Theta=2\,\theta\, r / d \Rightarrow$ $\theta= \Theta\, d / (2\, r)$ θ = motor angleP = $\omega / k_{\omega P}$ = $= \Omega\, d / (2\, r\, k_{\omega P})$
6	How to make the robot follow with its inner wheel (see fig. 2.5.2) an arc of circle of a given angle Θ, radius R and angular speed Ω?	Turn(Θ, R, Ω)	It is necessary to use the general formulas calculated in the previous section. The duration of the motion can be calculated in terms of θ (motor angle).

In the problems shown above, the wheel radius *r* and distance *d* have been conceptually considered constant parameters instead of function parameters because they are related to the building details. Their values will obviously influence the calculations internal to the functions to be realized.

There is a very general problem that might be mentioned at this point. The current standard NXT firmware does not support calculation and variables for fractional numbers (neither fixed nor floating point). It supports only 32-bit signed integer that can contain values between $-2^{31} = -2147483648$ and $+2^{31}-1 = +2147483647$. This must be seriously considered, when you make complex calculations. Two general rules addressing this problem are:

- Scale the constants to obtain numbers not too large (to avoid overflow the capacity of an integer) and without significant fractional digits.
- Leave an operation that implies truncation (the division and, if installed as an extension block, the square root) as the latest operation, when possible.

For example, if you have to calculate the following expression (in general, some values of the expression could be in variables):

124 * 13.2 / (5.7 – 2.5)

do actually the following ($\lfloor x \rfloor$ stands for floor(x), i.e. the greatest integer ≤ x):

57-25 = 32 -> var1

$\lfloor (124 * 132) / \mathrm{var1} \rfloor = 511$

whereas $\lfloor 124 / 32 \rfloor * 132$ results in = 3 * 132 = 396

As a further example, let us now propose the general solution for the problem 6 above. We would realize the function *turn(Θ, R, Ω)* with:

Θ robot angle in degrees

R motion radius in mm

Ω robot angular speed in degrees/s

We must now calculate the requested powers P_1 and P_2 to be respectively applied to the motors and the wheel angles θ_1 and θ_2 to be performed. We use now formulas (2.5.4) and (2.5.10):

$$P_1 = \omega_1 / k_{\omega P} = \Omega R / (r k_{\omega P}) \quad (2.5.20)$$

$$P_2 = \omega_2 / k_{\omega P} = \Omega (R+d) / (r k_{\omega P}) \quad (2.5.21)$$

$$\theta_1 = \Theta R / r \quad (2.5.22)$$

$$\theta_2 = \Theta (R+d) / r \quad (2.5.23)$$

If d=80 mm, r=28 mm, $k_{\omega P}$ = 8.15 degrees/s, and you want Θ=40 degrees, R=200 mm, Ω=20 degrees/s we obtain:

$P_1 = \Omega R / (28 * 8.15) = 20 * 200 / 228.2 = 17.52$

$P_2 = \Omega (R+80) / (28 * 8.15) = 20 * 280 / 228.2 = 24.54$

$\theta_1 = \Theta R / 28 = 40 * 200 / 28 = 285.71$ degrees

$\theta_2 = \Theta$ (R+80) / 28 = 40 * 280 / 28 = 400 degrees

We can check that:

$P_1 / (P_2 - P_1) = 17.52 / (24.54-17.52) = 2.495 \approx 2.5$

and

R/d = 200/80 = 2.5

Now, if we take into account the imprecision due to the integer calculations, we obtain:

$P_1 = \lfloor \Omega\ R\ 10/ 2282 \rfloor = \lfloor 20 * 200 * 10 / 2282 \rfloor = \lfloor 40000 / 2282 \rfloor = 17$

$P_2 = \lfloor \Omega\ (R+80)\ 10/ 2282 \rfloor = \lfloor 20 * 280 * 10 / 2282 \rfloor = \lfloor 56000 / 2282 \rfloor = 24$

$\theta_1 = \lfloor \Theta\ R / 28 \rfloor = \lfloor 8000 / 28 \rfloor = 285$ degrees

$\theta_2 = \lfloor \Theta\ (R+80) / 28 \rfloor = \lfloor 11200 / 28 \rfloor = 400$ degrees

Therefore, in addition to the intrinsic imprecision of the rotation sensor of the motor, the calculation causes a further error:

$\omega_1 = k_{\omega P}\ P_1 = 8.15 * 17 = 138.55$ $\Omega = \omega_1\ r / R = 138.55 * 28 / 200 = 19.397$

$\omega_2 = k_{\omega P}\ P_2 = 8.15 * 24 = 195.6$ $\Omega = \omega_2\ r / (R+d) = 195.6 * 28 / 280 = 19.56$

$\Theta = \theta_1\ r / R = 285 * 28 / 200 = 39.9$

$\Theta = \theta_2\ r / (R+d) = 400 * 28 / 280 = 40$

So we can conclude that the introduced errors are negligible. As a final remark, let us take into account also the imprecision in determining the two 'intrinsic' parameters *r* and above all *d* with a direct measure: it is advisable to use a first estimation of the two parameters for the calculations and then validate the parameters through a direct experience with the robot, tuning the initial estimation accordingly to the obtained motion. In particular, we can suggest to make the robot to pivot for a given angle (e.g. 360 degrees) with a simple program and to tune the factor *d*/*2r* that multiplies the motor angle θ in order to obtain the desired result.

2.5.4 Building Instructions for the Tiny Turtle

We now present a very simple instance of the turning robot class, initially not provided with sensors (apart from, obviously, the integrated rotational sensor), for which we will define 4 basic motion commands as in the case of the well known Logo turtle (fig. 2.5.10). For this reason we call it *tiny turtle*. Actually, the presented layout is not mandatory: any instance of the class could be easily adapted to this use, being based on two separately controlled motors. So, the building instructions are quite generic and aim at providing the simplest necessary robot.

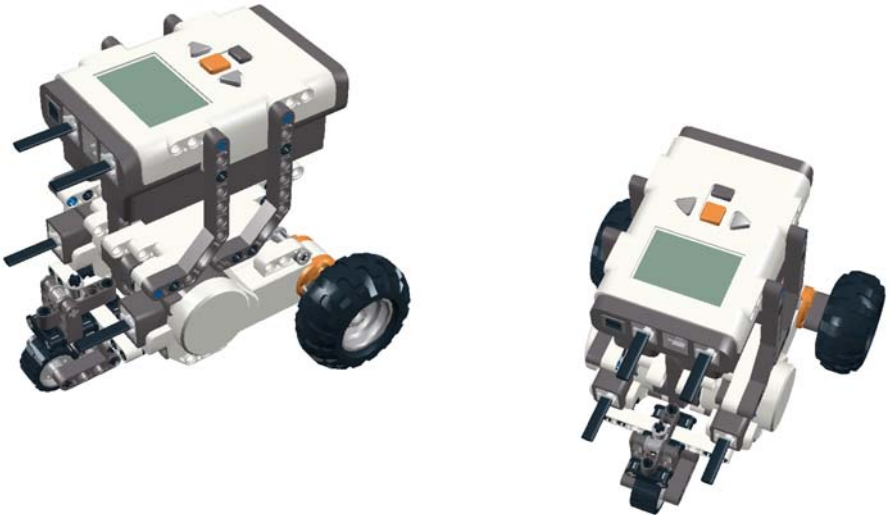


Fig. 2.5.10 *– The Tiny Turtle*

2.5.5 Programming the Basic Commands for the Tiny Turtle

We must prepare 4 basic commands (forward, backward, left and right) in terms of building blocks to permit the user to define in NXT-G the equivalent of a (simplified) Logo program. Therefore, we use the 'My block' technique of NXT-G to define such commands. Forward (Fd) and Backward (Bk) commands are straight motions and request a parameter specifying a distance measured in conventional units called '*steps*'. Left (Lt) and Right (Rt) commands make the robot pivot respectively counterclockwise and clockwise for an angle usually measured in degrees and specified by a command parameter. The duration of each movement is not relevant for a simple Logo-like turtle: thus, we could accept a standard speed profile and use an arbitrary power value even though we suggest to choose an intermediate value producing not too fast but also not too slow movements.

Recalling the formulas obtained in section 2.5.3 for Fd(distance) and Bk(distance) commands, we must apply the same power to two successive Motor blocks or use one Move block with the steering set to 0. Apart from an initial and a final very short transition, the robot moves with a rather constant speed; from (2.5.1) we know that the common motor angle θ should be set to $k_{d\theta}$\**distance* (*distance* measured in steps) with a reasonable constant $k_{d\theta}$. If there is no other specific reason to force this constant to some value, we suggest to set $k_{d\theta}$=1, i.e. 1 step = r 2 π / 360 mm with r the radius of the wheels in mm, corresponding to the minimum measurable rotation of the wheels of 1 degree.

The commands Lt(angle) and Rt(angle) must make the robot pivot for an angle Θ=*ang*. Because in this case from (2.5.9) it holds:

$\Theta = 2\ |\theta_{1,2}|\ r\ /\ d$

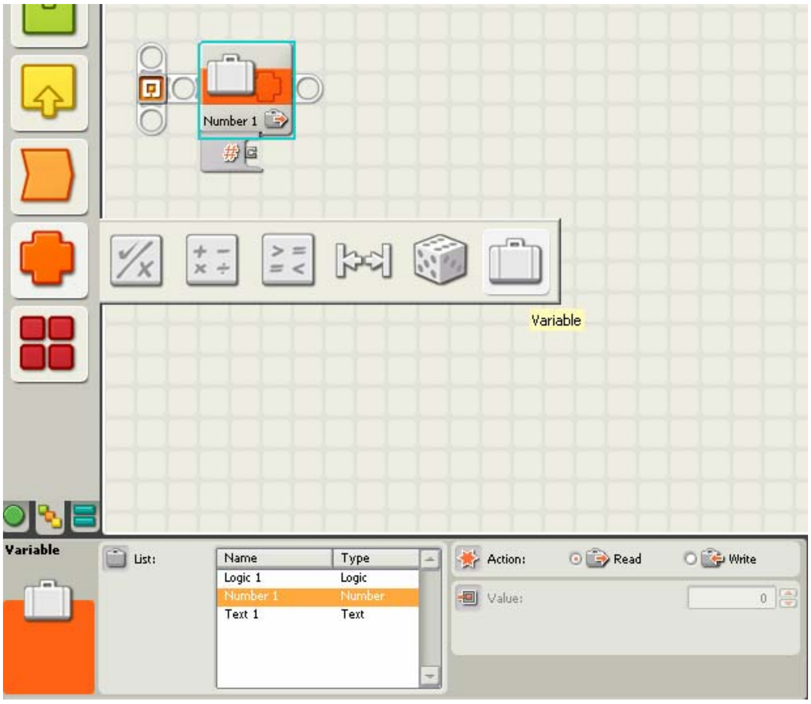
we must impose:

$|\theta_{1,2}| = \text{ang}\ d\ /\ 2\ r$

with *d* the distance between the two wheels.

Now, the description of the Lt block follows in the usual graphical form and then with the textual pseudo-code NXT-GTD. The other command blocks can be similarly programmed.

In a new (sub) program we add a numeric Variable block (the name for the moment is irrelevant, it is just a placeholder), representing the angle input parameter (fig. 2.5.11). Then, we must multiply for *d* and divide for 2*r*: it is very probable that these two values are integers with sufficient precision if measured in mm. Fig. 2.5.12 shows these operations with *d*=124 mm and 2*r*=56 mm. Notice the data wires necessary to connect the inserted blocks with each other.



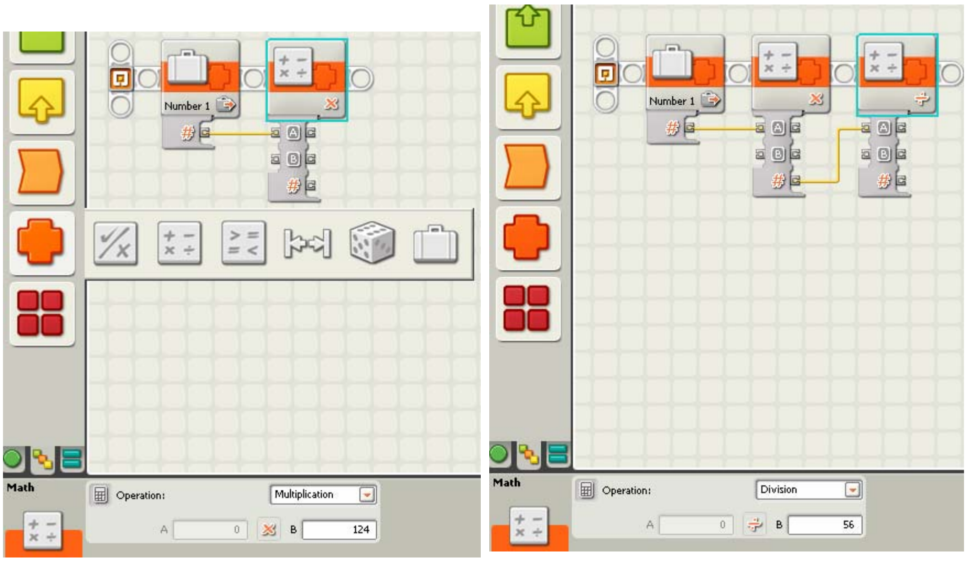
List:

Name	Type
Logic 1	Logic
Number 1	Number
Text 1	Text

Action: ● Read ○ Write

Value: 0

Fig. 2.5.11 *– The Lt block (I)*



Operation: Multiplication
A 0 B 124

Operation: Division
A 0 B 56

Fig. 2.5.12 *– The Lt block (II) (III)*

If either of the two is not an integer value, it is necessary to scale such values with the same constant K>1 for maintaining a sufficient precision:

$D = \lfloor d\,K \rfloor \qquad R = \lfloor 2\,r\,K \rfloor$

and then calculate $\lfloor angle*D/R \rfloor$. For example, if angle=45, d=83.5 mm. and r = 28 mm:

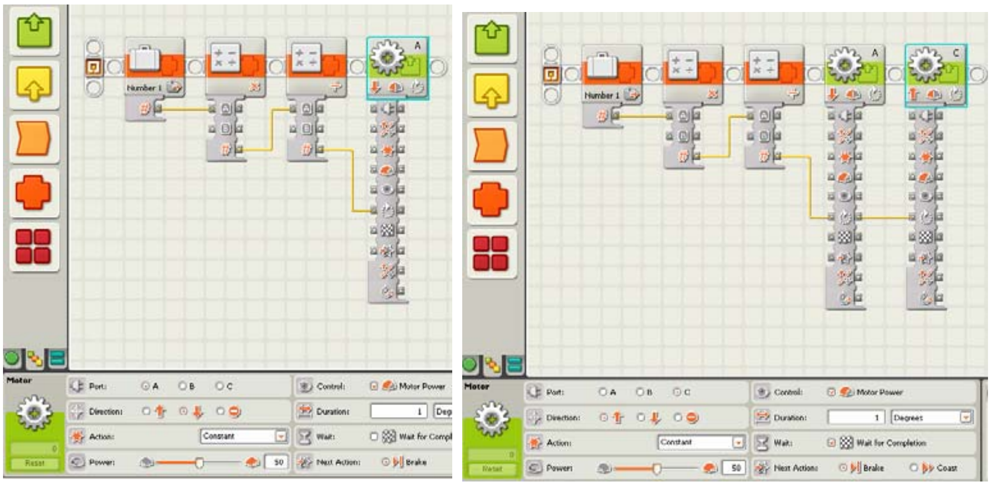
$\lfloor 45*\lfloor 83.5 \rfloor / 56 \rfloor = \lfloor 3735 / 56 \rfloor = 66$

With K=10:

$\lfloor 45*835 / 560 \rfloor = \lfloor 37575 / 560 \rfloor = 67$

Now, if we decide to control the motors with two distinct Motor blocks, with an intermediate power of 50(%) and assuming A and C the two respective control ports, the two new blocks have the same calculated absolute value in degrees for the Duration parameter but opposite Direction, and only the first one does not wait for completion (fig. 2.5.13).

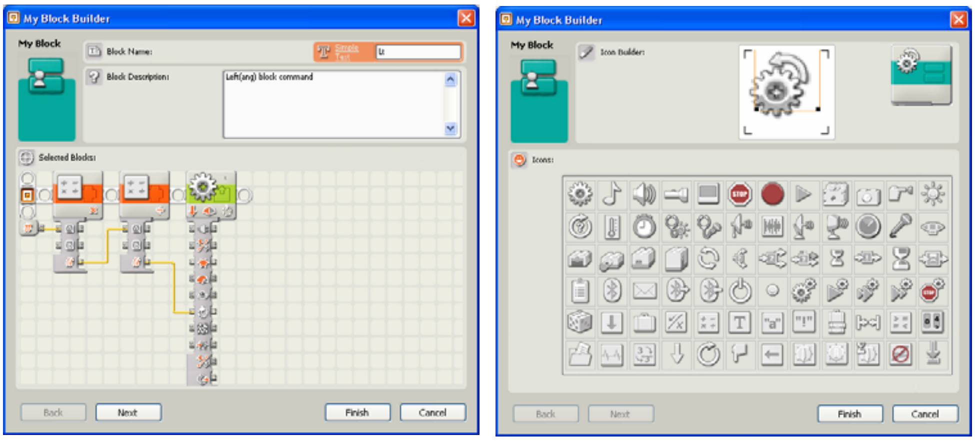
To define this ‘code’ as a new block using the ‘My block’ feature of NXT-G, select the last 4 blocks (Multiply, Division, MotorA and MotorC) and click on the ‘Create My Block’ button (or choose the equivalent menu item), assign the name Lt to the new user block (fig. 2.5.14 a) and a representative icon (fig. 2.5.14 b).



Port: ● A ○ B ○ C
Direction: ○ ↑ ● ↓ ○ ⊖
Action: Constant
Power: 50
Control: ☑ Motor Power
Duration: 1 Deg
Wait: □ Wait for Compl
Next Action: ● Brake

Port: ○ A ○ B ● C
Direction: ● ↑ ○ ↓ ○ ⊖
Action: Constant
Power: 50
Control: ☑ Motor Power
Duration: 1 Degrees
Wait: ☑ Wait for Completion
Next Action: ● Brake ○ Coast

*Fig. **2.5.13** – The Lt block (IV)* *(V)*



Block Name: Lt
Block Description: Left(ang) block command

*Fig. **2.5.14** – The Lt block a (VI)* *b (VII)*

By double-clicking the Lt block, you can edit its definition and, specifically, the name of the input parameter, which appears now as an input connection 'angle' on which you can attach a data wire to provide the angle of rotation (fig. 2.5.15).

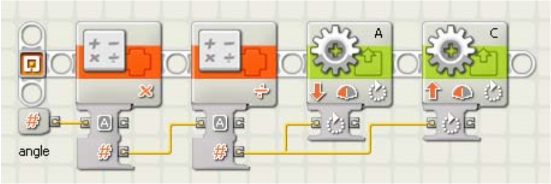


*Fig. **2.5.15** – The Lt block (VIII)*

This program expressed in NXT-GTD is as follows.

```
Lt(angle:degrees) : []
Mul1: MathOp(Type=MUL, A=Lt.angle, B=124)
Div1: MathOp(Type=DIV, A=Mul1.Res, B=56)
Mt1: Motor(Port=A, Dir=BK, Act=CONST, Pwr=50, PwrCtrl=ON,
  Dur=Div1.Res.DEG, Wait=OFF)
Mt2: Motor(Port=C, Dir=FD, Act=CONST, Pwr=50, PwrCtrl=ON,
  Dur=Mt1.Dur.DEG, Wait=ON, Next=BRK)
```

Once the 4 basic turtle commands are programmed, we can define a piece of code equivalent to some Logo code. For example, to make the turtle 'draw' a regular polygon of n edges each long l steps, in Logo we would say:

repeat n [fd l right 360 / n]

(in the turtle geometry, we know that 360/n is the amount of degrees the turtle must turn to have an internal angle of (n-2)*180/n degrees, which is required for a regular polygon).

In NXT-G the equivalent is shown in fig. 2.5.16, whereas the NXT-GTD version follows:

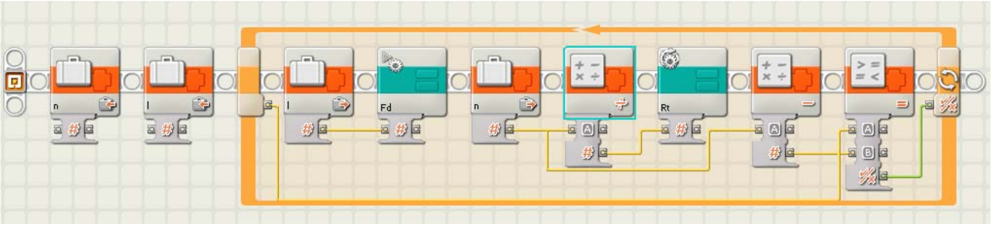


***Fig. 2.5.16** – Logo-equivalent code example*

```
VarDecl(Name=n, Type=NUM)
VarDecl(Name=l, Type=NUM)
Var(Name=n.NUM, Act=WR, Val=5)  -- n=5 pentagon
Var(Name=l.NUM, Act=WR, Val=300) -- l=300 steps
Loop1: Loop(Ctrl=LOGIC, Until=Cmp1.Res, ShowCnt=ON) [
        Vl: Var(Name=l.NUM, Act=RD)
        Fd(Vl.Val)
        Vn: Var(Name=n.NUM, Act=RD)
        D1: MathOp(Type=DIV, A=360, B=Vn.Val)
        Rt(D1.Res)
        S1:MathOp(Type=SUB, A=Vn.Val, B=1)
        C1:CmpOp(Type=EQ, A=Loop1.Cnt, B=S1.Res)
Loop1]
```

2.5.6 Some didactical Issues for the Tiny Turtle example

This example offers several ‘foods for thought’ that the teacher can exploit. First of all, the geometrical theory, which is on the basis of the definition of the 4 fundamental motion commands, is present in this section. Other interesting issues come from the ‘turtle geometry’ and regard the drawing of known figure (like the n-edge regular polygon described above), the simulation of physical phenomena, and others. Unfortunately, one of the most interesting aspects in Logo, the recursion, is (apparently) not supported by the NXT firmware.

During a preparatory stage, the teacher can propose the general problem of turning robots to the students and discuss with them if the building of a car with a steering sub-system is realizable. The teacher should help the students to identify the simplest two-motorized solution, also on the basis of real experiences (supermarket trolleys, agricultural machines, segways, etc.). A further investigation can make the student discover some general relations based on geometry and trigonometry. Then, the teacher suggests the realization of the turtle as an example of unsensorized turning robots. The discussion goes on about how to realize the 4 basic commands. After the actual realization of the turtle, the teacher suggests to prepare some Logo-style demo programs (for example, to verify that the turtle can follow a polygon already drawn on the plane). Finally, the students can discuss the correspondence between the theory and the practical realization. Some suggestions could be stimulated to add useful sensors to the turtle and to provide a reasonable way to use them.

2.5.7 Appendix

2.5.7.1 Investigation of the relation power - ω

The program to evaluate the relation between the power (0 .. 100) applied to both motors (steering=0) and the common angular speed obtained is based on the data logger schemata of section 2.7 and presented in fig. 2.5.17.

The equivalent NXT-GTD description follows.

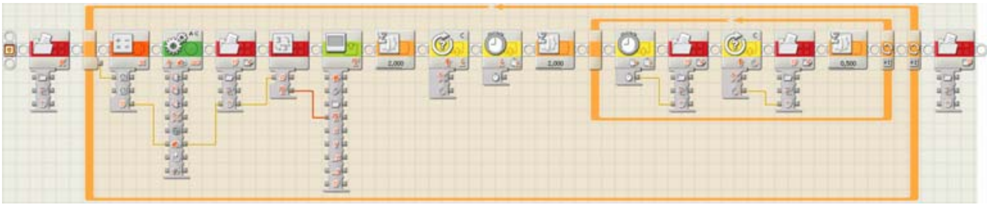


Fig. 2.5.17 – ω vs. P ‘plotting’ program

```
File(Act=DL, Name=omega)
Lo1: Loop(Ctrl=COUNT, Until=11, Dis=ON) [
  Mu1:  MathOp(Type=MUL, A=Lo1.Cnt, B=10)
  Mv1:  Move(Ports=AC, Dir=FD, StLt=A, StRt=C, Steer=0, Pwr=Mu1.Res,
        Dur=FOREVER)
  F1:   File(Act=WR, Name=omega, Type=NUM, Val=Mv1.Pwr)
  Nt1:  N2Txt(Num=F1.Num)
        Display(Type=TXT, Clr=ON, Txt=Nt1.Txt, PosX=8, PosY=32,
          Line=4)
        Wait(Ctrl=TIME, Until=2)  -- wait for motion stabilization
        RotSens(Port=C, Act=RESET, Cmp=??)
        Timer(Num=1, Act=RESET, Cmp=??)
        Wait(Ctrl=TIME, Until=2)  -- wait for motion stabilization
  Lo2:  Loop(Ctrl=COUNT, Until=10, Dis=OFF) [
        Ti1:   Timer(Num=1, Act=READ, Cmp=??)
        F2:    File(Act=WR, Name=omega, Type=NUM, Val=Ti1.Val)
        Ro1:   RotSens(Port=C, Act=READ, Cmp=??)
        F3:    File(Act=WR, Name=omega, Type=NUM, Val=Ro1.Val)
               Wait(Ctrl=TIME, Until=0,5)  -- wait for next sample time
        Lo2]
Lo1]
File(Act=CL, Name=omega)
```

The program repeats 11 times, applying orderly powers 0, 10, 20, . . ., 100, the acquisition of 10 samples of the motor angular position through its integrated rotation sensor. In the internal file "omega" such data appear as follows:

```
. . .
20      -- power applied
2002    -- sample time 1
293     -- sample value 1
2514    -- sample time 2
366     -- sample value 3
. . .
6606    -- sample time 10
963     -- sample value 10
30      -- power applied
. . .
```

The ω angular speed for a certain power is estimated as the average value of the 9 ratios:

$v_i - v_{i-1} / t_i - t_{i-1}$ with $1 \le i \le 10$

2.5.7.2 Investigation of the relation steering - ratio $\omega_1 / (\omega_2 - \omega_1)$

Similarly, as with the previous relation, we prepared a sampling program evaluating both average angular speeds at a sequence of different steering values (fig. 2.5.18).

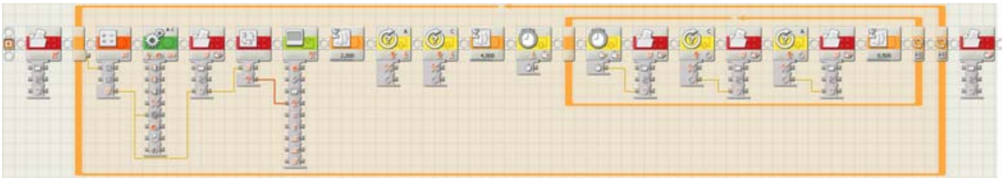


***Fig. 2.5.18** – ω vs. P 'plotting' program*

The equivalent NXT-GTD description is as follows:

```
File(Act=DL, Name=steer)
Lo1: Loop(Ctrl=COUNT, Until=11, Dis=ON) [
 Mu1: MathOp(Type=MUL, A=Lo1.Cnt, B=10)
 Mv1: Move(Ports=AC, Dir=FD, StLt=A, StRt=C, Steer=Mu1.Res, Pwr=70, Dur=FOREVER)
 F1:  File(Act=WR, Name=steer, Type=NUM, Val=Mv1.Steer)
 Nt1: N2Txt(Num=F1.Num)
      Display(Type=TXT, Clr=ON, Txt=Nt1.Txt, PosX=8, PosY=32, Line=4)
      Wait(Ctrl=TIME, Until=2)  -- wait for motion stabilization
      RotSens(Port=A, Act=RESET, Cmp=??)
      RotSens(Port=C, Act=RESET, Cmp=??)
      Wait(Ctrl=TIME, Until=4)  -- wait for motion stabilization
      Timer(Num=1, Act=RESET, Cmp=??)
 Lo2: Loop(Ctrl=COUNT, Until=10, Dis=OFF) [
      Ti1:   Timer(Num=1, Act=READ, Cmp=??)
      F2:    File(Act=WR, Name=steer, Type=NUM, Val=Ti1.Val)
      Ro1:   RotSens(Port=C, Act=READ, Cmp=??)
      F3:    File(Act=WR, Name=steer, Type=NUM, Val=Ro1.Val)
      Ro2:   RotSens(Port=A, Act=READ, Cmp=??)
      F4:    File(Act=WR, Name=steer, Type=NUM, Val=Ro2.Val)
             Wait(Ctrl=TIME, Until=0,5)  -- wait for next sample time
      Lo2]
Lo1]
File(Act=CL, Name=steer)
```

2.6 Data Logger

2.6.1 Introduction

When the main objective of a project-based activity is to discover or verify a general law that controls a phenomenon, or to make some statistics on the experiment, one usually needs to collect lots of data from the real world. The manual acquisition of experimental data, though interesting from an educational point of view, is subjected to unavoidable inaccuracies that can compromise the subsequent analysis. Even when acquiring data from the environment was not initially thought as necessary, this need may arise successively, when, for example, the behaviour of the robot does not adhere perfectly to what is expected, when any aspect of the environment is worth being more deeply and quantitatively analyzed and when it is interesting to study the time profile of a motion in detail.

The Lego Mindstorms NXT firmware permits us to use sensors not only for robot controlling purposes, but, also, to get samples from such inputs and to store them onto an internal file that can be subsequently uploaded on a PC for post-elaborations. The integration of data analysis in a project is a substantial way to organize the development of its stages, particularly during the investigation stage. To describe this feature through a simple but effective example, we prepared an experiment that we called ‘data logger’ (DL). In this experiment the goal is to study the uniformly accelerated motion and to deduce from the acquired data its fundamental quadratic law between space and time.

2.6.2 Building instructions

Because, as we have seen, the NXT servo-motors are speed-controlled devices, it is much simpler to use the natural gravity acceleration in order to impose a constant acceleration to a vehicle instead of forcing a linear increasing profile to its speed that would mean to increase correspondingly the applied power during the motion. Therefore, we built a very simple car on four free-to-rotate wheels without motors (fig. 2.6.1); the car is equipped with a sonar sensor to get space data. Left to move freely on a slope with a constant inclination θ (fig. 2.6.2), the car is subjected to an acceleration, which is the component of the g gravity acceleration parallel to the slope, whose absolute constant value is $g\ sin\ \theta$ (whereas the orthogonal component $g\ cos\ \theta$ is compensated by the static reaction of the plane).

2.6.3 Programming the data logging

The data acquisition regards the distance measured by the sonar sensor in respect of a fixed obstacle put on the starting side of the slope. Starting the acquisition before leaving the vehicle free to move down the slope under the influence of the constant acceleration lets the program store in a file the entire sequence of sampled values of the increasing distance during the motion. In order to analyze the motion, the teacher/student must reconstruct the relation between this sequence of samples and the time. To reduce the influence of the unavoidable delays introduced by the execution of the command blocks in the program, we suggest storing each sample value together with its sampling time with immediately successive commands.

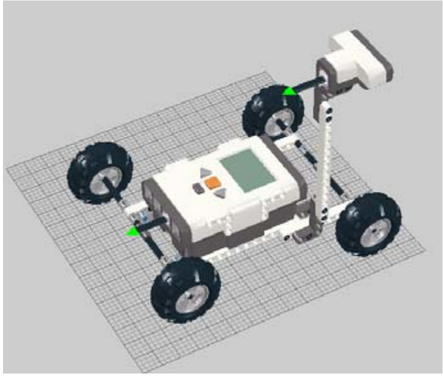


***Fig. 2.6.1** – The simple car*

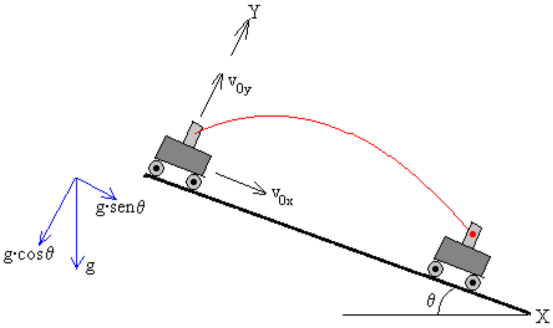


***Fig. 2.6.2** – The slope and the acceleration*

The NXT-G code of the program is shown in fig. 2.6.3 and corresponds to the following NXT-GTD description:

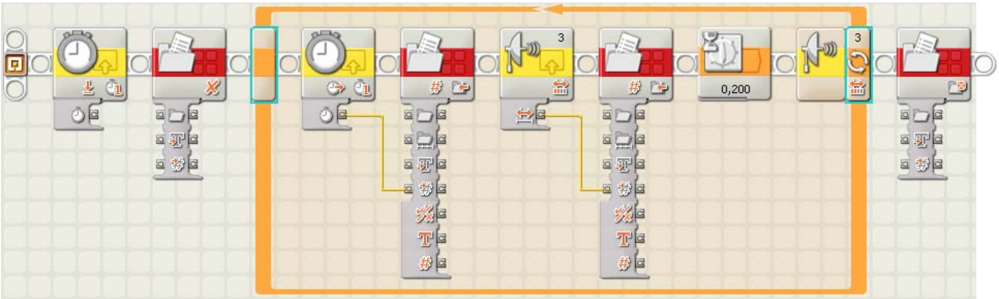
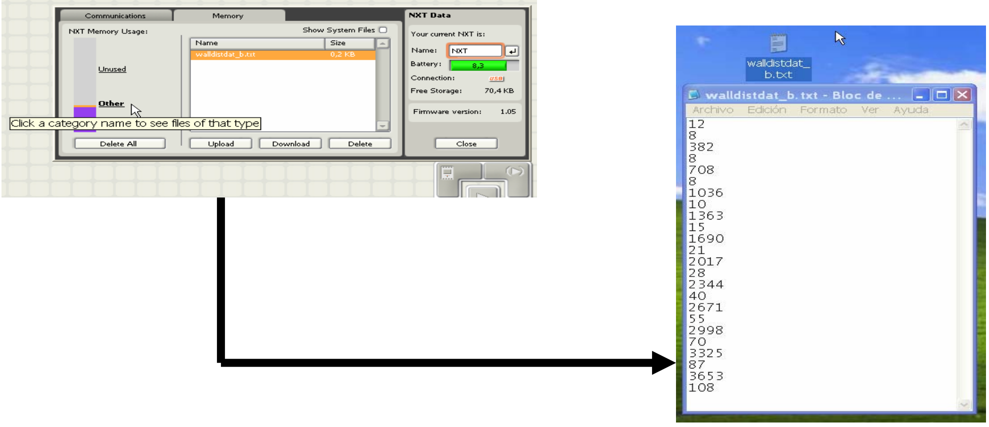


***Fig. 2.6.3 –** The Data Logger NXT-G program*

```
Timer(Num=1, Act=RESET, Cmp=??)
File(Act=DL, Name=dldata)
Lo1: Loop(Ctrl=SENSOR, Sensor=SONAR, Dis=OFF, Port=3, Until=GT 110,
  Show=CM) [
  Ti1:    Timer(Num=1, Act=READ, Cmp=??)
  F2:     File(Act=WR, Name=dldata, Type=NUM, Val=Ti1.Val)
  So1:   SonarSens(Port=3, Cmp=??, Show=CM)
  F1:    File(Act=WR, Name=dldata, Type=NUM, Val=So1.Dist)
         Wait(Ctrl=TIME, Until=0.2)  -- wait for next sample time
Lo1]
File(Act=CL, Name=dldata)
```

The first block resets the timer, whereas the second block deletes the file possibly produced in a previous session, to avoid new data being appended to the old ones. In the loop, the program with a period of (about) 0.2 s, samples the measure given by the sonar sensor and writes the value of the timer and the sample in the file. The two readings, that of the timer and that of the sensor, must be done with the minimum possible time separation. The loop ends when the distance reaches a maximum (the end of the straight path of the car) and the last block of the program closes the file. The recorded ASCII file with the acquired data can be uploaded onto the PC with the use of a specific NXT-G function (fig. 2.6.4). Time samples are given in millisecond unit, distances in centimetres.



Name	Size
walldistdat_b.txt	0,2 KB

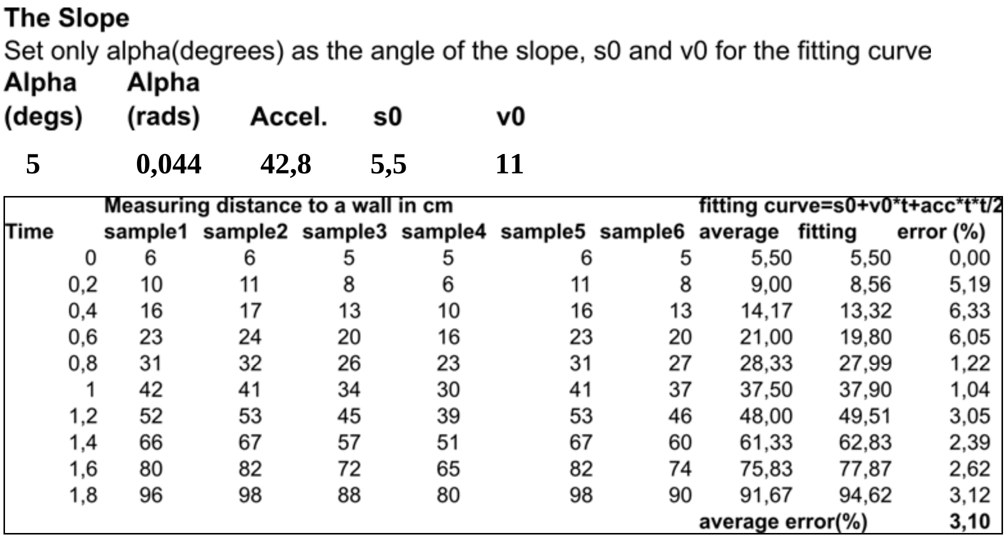
Your current NXT is:
Name: NXT
Battery: 8,3
Connection: USB
Free Storage: 70,4 KB
Firmware version: 1.05

walldistdat_b.txt - Bloc de ...

```
12
8
382
8
708
8
1036
10
1363
15
1690
21
2017
28
2344
40
2671
55
2998
70
3325
87
3653
108
```

***Fig. 2.6.4** – How to upload the data file*

The acquired data can be more effectively analyzed if reported in a spreadsheet and then plotted: in fig. 2.6.5 you can see the sequences of samples for 6 different repetitions of the experiment. The samples, connected by means of interpolation curves, are shown in the plotting (fig. 2.6.6) together with a fitting quadratic function $f(t)=s_0+v_0*t+a*t^2/2$ with a the assumed constant acceleration, and s_0 and v_0 chosen to make good enough the approximation with respect to the sequence of samples calculated as the average of the 6 corresponding measured samples. One of the most interesting findings that students should "discover" is that a physical phenomenon is only partially perfectly repeatable, due to noise errors and other physical inaccuracies (e.g. irregular friction, limited sensor precision, etc.). The plotting of the results of the repetition of the DL experiment can convince them, particularly when compared with the theory (fig. 2.6.7).



The Slope

Set only alpha(degrees) as the angle of the slope, s0 and v0 for the fitting curve

Alpha (degs)	Alpha (rads)	Accel.	s0	v0
5	0,044	42,8	5,5	11

	Measuring distance to a wall in cm						fitting curve=s0+v0*t+acc*t*t/2		
Time	sample1	sample2	sample3	sample4	sample5	sample6	average	fitting	error (%)
0	6	6	5	5	6	5	5,50	5,50	0,00
0,2	10	11	8	6	11	8	9,00	8,56	5,19
0,4	16	17	13	10	16	13	14,17	13,32	6,33
0,6	23	24	20	16	23	20	21,00	19,80	6,05
0,8	31	32	26	23	31	27	28,33	27,99	1,22
1	42	41	34	30	41	37	37,50	37,90	1,04
1,2	52	53	45	39	53	46	48,00	49,51	3,05
1,4	66	67	57	51	67	60	61,33	62,83	2,39
1,6	80	82	72	65	82	74	75,83	77,87	2,62
1,8	96	98	88	80	98	90	91,67	94,62	3,12
							average error(%)		3,10

***Fig. 2.6.5** – Experimental data on a spreadsheet*

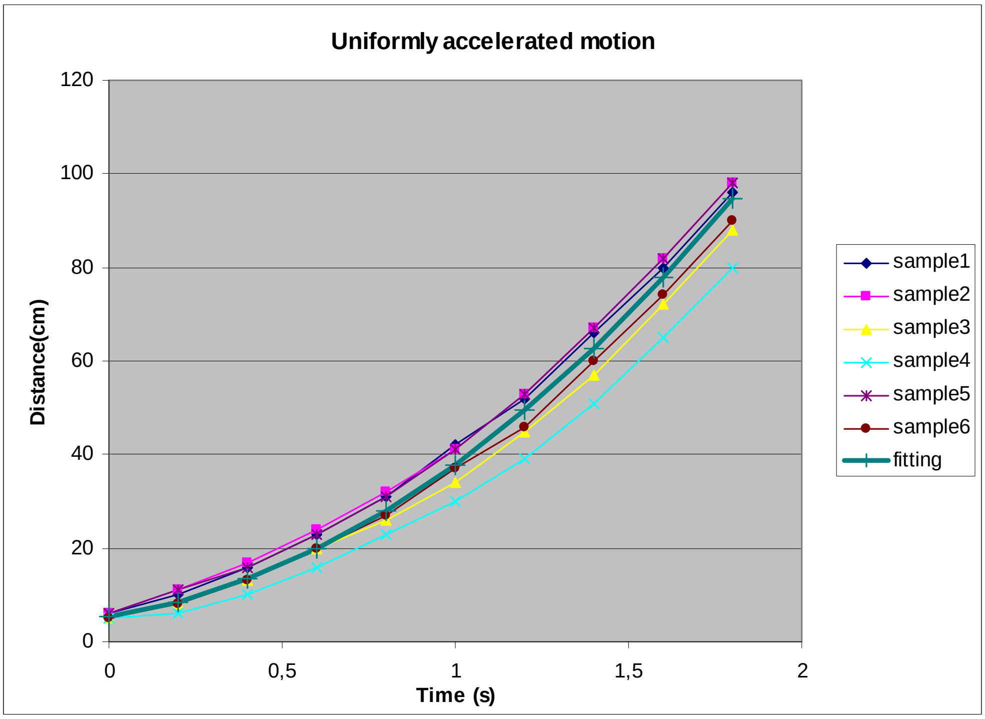


***Fig. 2.6.6** – The DL data plotting*

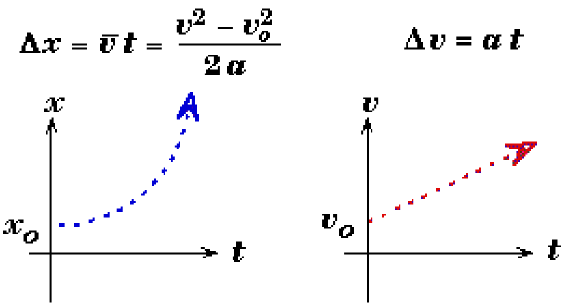


Fig. 2.6.7 *– The uniformly accelerated motion theory*

The accuracy of the acquisition could be improved by storing the sampling time and the associated sample value into two variables before copying them into the file in order to reduce the time separation of the two readings. Another possibility is to correct the stored sampling time of an amount calculated from an estimation of the delays introduced by the execution of the program blocks.

A variant of the presented DL experiment could use a rotational sensor instead of the sonar to provide the measure of the travelled space. In this case, you should use an external rotational sensor (not provided in the standard kit, for instance the old RCX sensor) because, if you connect the wheel to a NXT motor, the motor offers a great resistance when it is not executing a motion command, and therefore it does not leave the wheel free. Moreover, the student would be asked to deduce the linear space from the angle performed by the wheel.

2.6.4 Didactical Issues

After the experiment, the acquired data can be suitably displayed and used for a discussion among the students and the teacher:

- to discuss the evidence of the data in respect of the expected behaviour, trying to find reasonable justifications for possible deviations;
- to deduce laws, constraints, proofs and intuitions from the shared analysis;
- to acquire a deeper insight in the physical phenomenon under experimentation;
- to provide a new awareness, which is the basic condition whereupon new knowledge can be built with a constructivist teaching/learning approach.

The DL example can be used as a prototype to realize attractive, rather complex data acquisition experiments with one sensor and also with more than one sensor. In the latter case the reading of samples might be done as much synchronously as possible to permit correct correlations among the different sensor data. For instance, one could study the correspondence between the rotation of a motor, measured through its internal sensor and the motion of the whole vehicle, measured with the sonar in case of a linear motion, like in the DL example, or with a gyroscope or a compass sensor in case of a rotational motion.

Chapter 3

Robotics as Learning Tool

3.1 A methodology for designing robotics - enhanced activities

Authors: Kyparisia Papanikolaou, StassiniFrangou

During the last few years, robotics is being introduced in school education, from kindergarten to high secondary school, either as an interdisciplinary, project-based learning activity or as a learning activity focused on school subjects, such as Science, Maths, Informatics and Technology. The use of robotics in education ensures a learning environment that enables learners to control the behavior of a tangible model by means of a virtual environment, which actively involves learners in problem-solving and encourages them to carry out experiments and create their own programmable artefacts (Resnick et al. 1996).

In this chapter, we propose a methodology for designing robotics-enhnaced project-based activities for students. The aim is to provide teachers with an operational framework for structuring students' work in the process of building and guiding a robotic construction in order to enable students to develop specific competencies and attain learning outcomes.

3.1.1 Project-Based Learning

Project-Based Learning is a comprehensive teaching and learning approach meant to engage learners in sustained, cooperative investigation (Bransford & Stein, 1993). Projects focus on the creation of a product or performance, and generally call upon learners to choose and organize their activities, conduct research, and synthesize information. According to current research (Thomas, Mergendoller, & Michaelson, 1999; Brown & Campione, 1994), projects are complex tasks, based on challenging questions that serve to organize and drive activities, which, taken as a whole, amount to a meaningful project. They give learners the opportunity to work relatively autonomously over extended periods of time and culminate in realistic products. PBL environments involve authentic assessment tasks, teacher support but not direction, collaborative work, and reflection at individual and group level (Han and Bhattacharya, 2001).

Project-based learning as a method of teaching and learning is mainly based on contemporary learning theories, which argue that knowledge, thinking, doing and the contexts for learning are inextricably tied. We know now that learning is partly a social activity, taking place within the context of culture, community, and real life experiences. Knowledge construction has become a key term in describing a more active students' role in developing and creating their own knowledge (see for example McCormick & Paechter, 1999). It is central in describing the process of learning within problem-based and project-based learning.

Project-based learning (PBL) is also a model for classroom activity that shifts away from the classroom practices of short, isolated, teacher-centered lessons and, instead, emphasizes learning activities that are long-term, interdisciplinary, student-centered, and integrated with real world issues and practices.

PBL helps make learning relevant and useful to students by establishing connections to life outside the classroom, addressing real world concerns, and developing real world skills. PBL supports learners to develop a variety of skills including the ability to work well with others, make thoughtful decisions, undertake initiatives, and solve complex problems.

In the classroom, PBL provides many unique opportunities for teachers to build relationships with students. Teachers may fill the varied roles of coach, facilitator, and co-learner. Finished products, plans, drafts, and prototypes all make excellent "conversation pieces" around which teachers and students can discuss the learning that is taking place.

Components of Project-Based Learning. Key components of Project-Based Learning that should be considered in describing, assessing, and planning for projects, are (Han and Bhattacharya, 2001): Learner-centered environment, Collaboration, Authentic tasks, Multiple-presentation modes, Emphasis on time management, Innovative assessment.

Learner-centered environment: PBL should be designed to maximize student decision-making and initiative throughout the course of the project by involving learners in topic selection and, throughout the course of the project, by providing them with control over the production and presentation of artefacts. Additionally, projects should include adequate structure and feedback to help learners make thoughtful decisions and revisions. Learners should document their decisions, revisions, and initiatives, with the aim to enhance reflections on their learning process and acquire valuable data for assessing their work and growth.

Collaboration: PBL is aimed at the development of communication and collaborative skills, enhancing group decision-making, interdependence, integration of peer and mentor feedback by providing thoughtful feedback to peer and working with others as learners/ researchers.

Authentic tasks: PBL should relate to the real world stimulating learners to address real world issues that are relevant to their lives or communities.

Multiple presentation modes: It is important to support and prompt learners, in the course of the project, to effectively use various technologies as tools in the planning, development or presentation of their projects.

Time management: Learners should have control of their learning through the course of the project, planning, revising and reflecting on their learning. Given the time frame and scope of a project, all projects should provide adequate time and materials to support meaningful doing and learning.

Innovative assessment: Assessment should be an ongoing process of documenting learning through the course of the project. PBL requires varied and frequent assessment, including teacher assessment, peer assessment, self-assessment, and reflection. Assessment practices should involve learners through consistent documentation of the process and results of their work enhancing reflection and self-assessment throughout the project.

3.1.2 Designing projects for learning

Constructionism (Harel and Papert 1991) is reflected in PBL by the emphasis on (Han and Bhattacharya, 2001): (a) the design of a student-centered learning environment; (b) artefact creation as part of the learning outcome based on authentic and real life experiences with multiple perspectives.

In this context, learners are promoted to become ‘active builders of knowledge’ while working on a project, exprimenting, investigating concepts, confronting misconceptions. Especially, *Learning by Design* emerges from the constructionist theory (Gagnon and Collay, 2001) that emphasizes the value of learning through creating, programming, or participating in other forms of designing. The design process creates a rich context for learning. Learning by Design values both the process of learning and its outcomes or products. The essence of Learning by Design is in the construction of meaning. Designers (learners) create objects or artefacts representing a learning outcome that is meaningful to them.

Specific guidelines for effective Learning-by-designing provided by Resnick are: (see http://llk.media.mit.edu/projects/clubhouse/research/handouts/)

- Design projects that *engage kids as active participants*, giving them a greater sense of control and responsibility for the learning process.
- Design projects that *encourage creative problem-solving*.
- Design projects that are *interdisciplinary*, bringing together ideas from art, technology, maths, and sciences.
- Design projects that help *kids learn to put themselves in the minds of others*, since they need to consider how others will use the things they create.

- Design projects that provide *opportunities for reflection and collaboration*.
- Design projects that *set up a positive-feedback loop of learning*: when kids design things, they get new ideas, leading them to design new things, from which they get even more ideas, leading them to design yet more things, and so on.

Learning by Design strongly suggests that tasks should be based on hands-on experience in real-world contexts. The designers/participants should be given the option of multiple contexts so that they can devise multiple strategies when they get involved in a problem-solving process. Because the learning process is open and varied according to the student individual characteristics, learning preferences, skills, and knowledge, it is important that there is a balance among guided tasks, challenges, discussions and reflections. Collaborative work allows the learners to obtain feedback from both, peers and the instructor, who primarily plays the role of facilitator (Han and Bhattacharya, 2001).

In summary, the essence of Learning by Design lies in the experience of the learner as a designer and creator of an external, shareable artefact. Learners become more responsible for their learning through designing, sharing, piloting, evaluating, modifying their work, and reflecting on the process. The instructor acts as a facilitator and motivator by creating an open-ended learning environment and by challenging and scaffolding the learners in a balanced manner, while providing options with rich and varied feedback. Through this experience, learners are expected to construct meaning and internalize the learning process (Han and Bhattacharya, 2001).

3.1.3 Designing robotics-enhanced constructivist learning environments

The methodology that we use for designing and implementing robotics-enhanced projects integrates the main principles of constructivism, constructionism and problem-based learning. The main aim is to propose a ‘tool’ for designing robotics-enhanced learning activities that promote:

- authentic learning (using resources of real-life, occupational situations, or simulations of the everyday phenomena).
- social learning (technology supports the process of joint knowledge development. The available e-learning environments can support collaboration between fellow students, who can be at different schools, at home or abroad).
- meaningful-active-reflective learning (students work on experiments or problem-solving, using available resources selectively according to their own interests, search and learning strategies).

- problem-based learning (a method that challenges students to "learn to learn"; student groups are seeking solutions to real world problems, which are based on a technology-based framework used to engage students' curiosity and initiate motivation, leading so to critical and analytical thinking).

Designing a robot to do even a simple task can place extensive demands on students' creativity and problem-solving ability (Druin & Hendler, 2000; Erstad, 2002; Carbonaro, Rex & Chambers, 2004). Building and programming autonomous robots is an ideal context in which to situate a project-based learning experience, where learners work collaboratively to understand the problem, propose viable solutions and construct their artefacts. It is quite important a driving question or problem to set the stage and the project context to allow for a multitude of design paths. Then, students should collaborate over an extended period of time during a problem-solving activity. The result of this collaboration is the construction of an artefact that will be presented to a wider classroom audience. The production of an artefact, which is readily sharable with a larger community of learners, encourages students to make their ideas explicit, whilst it allows them to experience science concepts in a meaningful, personalized context (Penner, 2001).

Robotics-enhanced projects should encourage learners to engage in complex and ill-defined contexts. From the beginning, learners identify their topics and problems and, then, seek possible solutions. By participating in both, independent work and collaboration, learners improve their problem-solving skills, thereby developing their critical thinking skills. However, one of the problems that learners face in such learning environments is what strategies to employ, how to start and proceed with the problem they have to address. To this end, different approaches have been suggested (Han & Bhattacharya, 2001; Houghton Mifflin, 2007).

Generally, three phases are suggested in conducting Project-Based Learning: planning, creating and implementing, processing (Han & Bhattacharya, 2001):

1. in the "planning" phase, the learner chooses the project, locates the required resources and organizes the collaborative work. Through these activities, the learner identifies and represents a topic, gathers relevant information and generates a potential solution.
2. the "creating or implementing" phase: This phase includes activities such as developing and documentating, coordinating and blending member contributions, and presenting to class members. In this stage, learners are expected to build a product that can be shared with others.
3. the activities of the "processing" phase, include reflection and follow-up on the projects. In this stage, the learners share their artefacts, obtain feedback, and reflect on the learning process and the project.

Moreover, specific features that need to be considered in organising the above phases are as follows:

- A "driving question or problem", which is anchored in a real-world problem and ideally uses multiple content areas, should serve to organize and drive activities
- Opportunities for students to make active investigations, which enable them to learn concepts, apply information, and represent their knowledge in a variety of ways
- Collaboration among students, teachers, and others in the community so that knowledge can be shared and distributed between the members of the "learning community"
- The use of technology as cognitive tool in learning environments that support students in the representation of their ideas: cognitive tools such as robotic kits, computer-based environment guiding the robots, graphing and presentation applications, web-based resources.

Especially for organizing students' activity in robotics-enhanced projects, we follow the above three phases of project-based learning and we further extend the model proposed by Carbonaro et al. (2004) with processes & tasks that take place within a robotic project (see Table 3.1.1) organised in stages:

- Engagement stage: students are provided with an open-ended problem and get involved in defining the project. This stage requires the identification and representation of a scientific problem. Students work as a class putting their ideas into a question format. As they are doing so, they are identifying and representing a problem and different issues involved (e.g. brainstorming at class level).
- Exploration stage: students get familiar with LegoLogo, controlling devices and software, make hypotheses and test their validity in real conditions, provide initial ideas. Students are divided in groups in order to answer to simple questions and study specific cases in order to get familiar with the controlling devices and software (e.g. work in groups with worksheets – structured activity).
- Investigation stage: students search for resources and investigate alternative solutions. Students reconsider the problem and the different issues raised during the engagement stage, based on the experience they have gained through the exploration stage. At this stage, students in collaboration with the teacher, formulate the driving questions/problems which link with the learning goals of the project. The student groups undertake to solve the particular problems, investigate alternative solutions and provide arguments on their final proposals concerning the artefact and the software they have developed (e.g. they work in groups with worksheets, keep diary – open activity).

- Creation stage: students share and combine their artefacts, synthesize 'solutions' to the project reflect on their initial ideas. Students present their work in class and then each group works on the synthesis of a final 'product', including the artefact and the software (e.g. they work in groups with worksheets, keep diary – result in a product). This work may lead to similar solutions but also to innovative proposals.

- Evaluation stage: students share their ideas, products at class level, provide arguments on their final proposals and evaluate them. Alternative solutions are presented at class level and evaluated on the basis of the driving questions/criteria posed at previous stages of the project (stages of engagement, investigation). At this stage, students should critically judge their work, express their opinions, compare their works and reach a common proposal for the project (e.g. make p*resentations*, *discuss*, *peer evaluation*). Students should also reflect on and evaluate their collaboration.

The above stages are not linear, but, in many cases, highly iterative, e.g. the creation stage may include investigation or the investigation stage may include creation. The main aim of the various stages and the supportive content provided at each one of them (such as worksheets, resources) is to engage learners in meaningful design experiences. To this end, we should design for designers – that is, to design things that will enable learners to design things (Resnick & Silverman, 2005). Thus, what is important in designing a project and the appropriate worksheets at each stage of the framework is to drive students to imagine, realize, critique, reflect, iterate (Maeda, 2000), and according to Resnick & Silverman (2005) "encourage students to design and redesign their artefacts, to mess with the materials, to try out multiple alternatives, to shift directions in the middle of the process, to take things apart and create new versions".

In the following table 3.1.1 the title of each stage, a short description, resources provided to students, results/products and tasks that students might perform or participate in, are presented for each stage.

Table 3.1.1 - Stages of students' activities while working on robotics-enhanced projects.

Stage	Description	Resources	Result	Proposed Tasks
Engagement stage	Students may be provided with an open-ended problem and get involved in defining the project and main issues involved	An open-ended problem raw materials: sites, newspapers, videos, magazines, stories, cases	Project description Open issues	Study of raw material Discuss Express opinions/ideas Pose questions Negotiate Brainstorming
Exploration stage	Students get familiar with controlling devices and software, make hypotheses and test their validity in real conditions	Representative examples, general guidelines, educational materials, software	Artefacts with specific functionality Diary	Study samples of representative constructions/programs Observe Gather information/Searching Experiment Collaborate / Negotiate / Argumentation
Investigation stage	Students formulate the driving questions / problems, investigate alternative solutions	General guidelines that organize students' investigation / diary. Educational content	Driving questions / problems Artefacts addressing the driving questions Diary	Reflect on previously defined open issues Make hypotheses that they can test Plan Collect evidence Interpret Evaluate Keep diary Collaborate / Negotiate / Argumentation
Creation stage	Students share and combine their artefacts, synthesize 'solutions' to the initial problem	Guidelines for keeping diaries	Group products / solutions to the initial problem Diary	Evaluate previous work Share ideas Synthesize a product Keep diary Collaborate / Negotiate / Argumentation
Evaluation stage	Students share ideas & products at class level, evaluate final group proposals, synthesize the final product	Guidelines for peer evaluation and synthesis of a final product	Common accepted product	Present their products Discus Peer evaluation Interviews

References

Bransford, J.D., Stein, B.S. (1993): The Ideal Problem Solver (2nd Ed). New York: Freeman.

Carbonaro, M., Rex, M., Chambers, J. (2004): Using LEGO Robotics in a Project-Based Learning Environment. *The Interactive Multimedia Electronic Journal of Computer-Enhanced Learning,* 6(1).

Druin, A. and Hendler, J. (2000): Robots for kids: Exploring new technologies for learning. San Diego, CA: Academic Press.

Erstad, O. (2002): Norwegian students using digital artefacts in project-based learning, *Journal of Computer Assisted Learning* 18, 427-437.

Gagnon, G. W., Collay, M. (2001): Designing for Learning: Six Elements in Constructivist Classrooms. Corwin Press.

Han, S., Bhattacharya, K. (2001): Constructionism, Learning by Design, and Project-based Learning. In M. Orey (Ed.), Emerging perspectives on learning, teaching, and technology. Available at: http://www.coe.uga.edu/epltt/LearningbyDesign.htm

Harel, I., Papert S. (1991): Constructionism. Norwood, New Jersey: Ablex Publishing Corporation.

Houghton Mifflin's Project Based Learning Space. Available at http://www.college.hmco.com/education/pbl/background.html). Last access 01/11/2009.

Maeda, J. (2000): Maeda@Media. Rizzoli Publications. New York.

McCormick, R., Paechter, C. (1999): Learning and Knowledge. The Open University:Paul Chapman Publishing

Penner, D. E. (2001): Cognition, computers, and synthetic science: Building knowledge and meaning through modeling. In W. G. Secada, (Ed.) Review of Research in Education. (pp. 1–35). Washington, DC: American Educational Research Association.

Resnick, M., Silverman, B. (2005): Some Reflections on Designing Construction Kits for Kids. Proceedings of Interaction Design and Children conference, Boulder, CO. available at http://llk.media.mit.edu/papers.php

Resnick,M., Martin, F.G., Sargent, R., Silverman, B. (1996): Programmable bricks: Toys to think with. IBM Systems Journal, 35 (3&4), 443-452.

Thomas, J. W., Mergendoller, J.R., Michaelson, A. (1999): Project-based learning: A handbook for middle and high school teachers. Novato, CA: The Buck Institute for Education.

3.2 A representative example: “The BusRoute” project

Author: Stassini Frangou

In this section, we will present “The BusRoute” project. It is a project developed according to the model presented in the section 3.1. The project consists of five stages: *Engagement stage, Exploration stage, Investigation stage, Creation stage, Evaluation stage*. The presentation of this project is organized in two parts:

- Part 1: Description of the project (Teacher’s guide)
- Part 2: Students’ worksheets

3.2.1 “The BusRoute”: Description of the project (Teacher’s guide)

During this project, students construct a bus with the use of Lego building materials and program its run along a pre-defined route with the use of appropriate software. This project mostly addresses students without previous experience in the use of educational robotics.

Goals: This project may meet learning goals in the fields of Physics, Mathematics, Technology and Informatics, while fostering, at the same time, skills and attitudes. Learners, upon completion of this project, will be able:

Knowledge:

- to describe the basic characteristics of a robot (Technology);
- to describe and explain the operation of simple construction (gears, axles, blocks, transfer of motion) (Technology);
- to design and construct a moving vehicle with the use of all the appropriate materials (wheels, axles, motors) (Technology);
- to use suitable software and programming structures in order to set in motion and control that vehicle with the use of motors and sensors (use of icon commands, control commands, repeat commands) (Informatics);
- to calculate physical quantities affecting the design and operation of a bus, such as speed, distance, sense of direction (Physics, Mathematics).
- to compare and evaluate proposed solutions in both, the construction and the programming of the models.

Skills:

- to solve problems;
- to formulate assumptions and check their soundness;
- to express and evaluate arguments based on the data they have collected;
- to organize their work and keep the course of their project under control

Attitudes:

- to acknowledge and appreciate the contribution of science and technology to modern man's welfare.
- to realize the value of the group work and collaboration showing respect to everybody's individuality.

Duration: The overall project may cover 12-14 teaching periods, if wholly developed. However, a teacher may choose to cover only a few of the activities and, in such a case, its duration will be shorter.

Age group and prerequisites: This project is meant for Secondary School students (12-15 years old) who have a basic knowledge on computer functions (familiarity with an operating system, with saving and retrieving files). Lastly, it is assumed that the learners concerned have very little experience or none at all in robotics.

Table 3.2.1: *The BusRoute- Duration of each stage*

Stage	**Duration**	**Worksheets**
Engagement stage	2 hours	Worksheet 1 Worksheet 2
Exploration stage	4-5 hours	Worksheet 3 Worksheet 4 Worksheet 5 Worksheet 6
Investigation stage	2-3 hours	Worksheet 7 Worksheet 8
Creation stage	1-2 hours	Worksheet 9
Evaluation stage	1-2 hours	

Inclusion in the school curriculum: The project is interdisciplinary and may, depending on the way it is introduced and the emphasis given to its development, be included in the Technology course of Secondary School, in the Informatics course or in the Physics course (topic: speed measurement) and in the Mathematics course (topic: circle perimeter, ratios) of Secondary School.

Software/Materials: Educational robotics requires both, the use of suitable building materials for the construction of robot models and the use of suitable software for their programming. For such kind of activities, the structural materials proposed is that of Lego Mindstorms NXT kit. The robot model programming may be done with Lego MINDSTRORMS Education NXT.

Proposed application in class: The following description concerns a proposed application course in class. Its goal is to display tools and methods that can be utilized within the framework of educational robotics by means of current trends about teaching and learning. Therefore, in no way are teachers prompted to faithfully follow the course, but rather to modify it with a view to serving the requirements arising from each learner group and to meeting the teacher goals and the goals of the broader social group they belong to.

The project includes five development stages, which are not always clearly distinct, but constitute a wider developmental framework.

Engagement Stage (2 teaching periods)

The inclusion of this project in the rest of the curriculum may be done through the general issue of Public Transport. Public Transport, a subject matter of the Technology course, may become a study topic within an environmental program or an interdisciplinary activity within the frameworks of Physics, Mathematics and Technology. Upon the end of this unit, learners will be able:

- to state advantages from using Public Transport;
- to describe the characteristics of a robot structure,
- to specify desirable characteristics that a robot bus should combine in order to meet the requirements of transportation in city center

A starting point of the introduction may be a photograph or a short video with a relevant topic arising from a real story. Within the framework of the discussion, which will follow, students may refer to their own experiences and discuss questions such as:

- In which cases have they themselves or their family used a bus or a train?
- Who normally use Public Transport on a daily basis?
- In what respect does Public Transport have advantages as compared with a private means of transport?
- In what respect does Public Transport have disadvantages as compared with a private means of transport?
- What is the action taken by the State or local authorities in order to encourage the use of Public Transport?
- Is Public Transport friendlier to the environment?

First Teaching period

In **Worksheet 1** (presented in Part 2 of this section), 5 pictures are given, each one of which can give rise to discussion on various issues.

Picture 1: Morning traffic in Washington. 98% of the Americans think that Public Transport must be used. (http://www.theonion.com/content/node/38644). This may well give rise to a discussion about the traffic problem, pollution, over-consumption of fuel, waste of time in traveling from place to place.

Picture 2: Modern electric trains in the USA. These trains serve commuters (living 20-30 km away from town), thus mitigating traffic problems on motorways and shortening the time needed to get to a town. These trains have amenities serving people with special needs. It may give rise to a discussion on issues concerning the convenience of passengers (air-conditioning, special groups, children, mothers, elderly people) (http://www.movingtoportland.net/public_transportation.htm)

Picture 3: Tramway in Budapest. Budapest has a good Public Transport system, operating from 4:30 to 23:00. Selectively, certain lines provide night services. This may give rise to a discussion on transport service frequency. (www.budapesthotels.com/touristguide/bkv.asp)

Picture 4: Line buses in Guatemala. They are regional buses manufactured as school vehicles. Now, in the seat that was meant for two children, two to three adults are squeezed in, together with their luggage. Of course, failures and accidents are expected (antiguadailyphoto.com/2006/08/01/).

Picture 5: Map of the London Underground railway system. The London Underground system has 12 lines, which, in combination with the bus services and the surface trains, serve a very large area. The London underground system became target of terrorists in 2005. This last item may give rise to discussion on safety matters(http://www.propertyinvesting.net/cgi-script/csNews/image_upload/specialreports_2edb.London%20 Tube%20Map.gif).

At the end of this initial teaching period, students may draw up a list of arguments supporting the use of Public Transport (section 3.2.2, Worksheet 1).

Second Teaching period

In the second teaching period, students are introduced into the scenario of the following project: the construction of a robot bus moving along a specified route. This route may be proposed by the teacher or may be planned in cooperation with the students. This task may be related to the stories studied already by the students in the previous teaching period, as well as to the students' own experiences. In the case of the urban centers, a good proposal is a robot bus serving a neighborhood and linked with different means of transport, such as a train. Alternative proposal is a robot train running within a park. In any case, the route covered should provide the students with opportunities and challenges for research and exploration (experimentation). At the end of this teaching period, the functions that we want the robot bus to perform and the route we want it to cover are expected to have been clearly described.

As an example, Worksheet 2 (Part 2) presents a scenario, where the learners are asked to design a robot bus to serve a downtown area restricted to pedestrians. Positive points, as well as concerns arising from the use of such a solution are:

Positive Points	Concerns
Easy downtown access within a short time (traffic problem avoided)	Difficult access of all the residents to the bus terminus and stops. Need for development of intermediate private car parking areas.
Reduced need for downtown car parking areas and, as a result, increased areas of green.	Delays caused to timetables and problems referring to transfers to and from other means of transport (train).
Less downtown air and noise pollution.	Difficult access by special groups of people (elderly people, mothers with young children, disabled).
More consumers and increased downtown commercial growth.	

At this point, we propose a discussion on the basic characteristics of a robot structure. A robot is a structure which has a physical entity, can carry out actions, i.e. has a behavior, but what makes it different from any ordinary mechanical structure is that it contains within it the ‘control’ element. In other words, a robot structure may collect data from the environment, decides, depending on those data, upon the actions to be taken and performs those actions (data as input, instructions-program, action-behavior as output).

At this point, we can continue with the description of the route. The bus can have a terminus and bus stops in an area easily accessible to residents, e.g. near a public car park, near the inter-city bus station, near the train station. Along its route, it will have to stop at bus stops, where there are passengers and wait while passengers get on and off the bus. For the convenience of blind people, it would be advisable to provide sound signals during the time passengers get on and off the bus. It is expected to develop a different speed at points where the road is not particularly safe, as well as to spot any obstacles and get immobilized. It is, finally, expected to be in a position to follow a pre-set route and park in a specific place.

Exploration Stage (3-4 teaching periods)

At the exploration stage, students become familiar with the materials that they are going to use (construction materials and software). Through their actions at this stage, they acquire the necessary experience which will enable them, at the subsequent stages, to develop independently their ideas. The activities at this stage are structured in such a way as to display the aspects that can be utilized in the solution of the problem they are asked to address. At the end of this unit, learners will be able:

- to recognize and name the basic categories of building materials;

- to combine materials in simple working structures;
- to observe and explain the operation of simple machines;
- to state characteristics of simple machines (e.g. relationships between motor rotations and vehicle displacement);
- to make use of the basic icon-commands in order to program their models;
- to investigate and compare;
- to draw conclusions.

Third Teaching period

Learners recognize and name the given materials and relate them to the functions of a robot. The materials that can be utilized are gears, pulleys, beams, blocks and different kinds of sensors. The Try Me menu of the NXT brick can be tested in order to illustrate the function of sensors within NXT programs. The activities of Worksheet 3 may be carried out in small groups.

Fourth Teaching period

The students continue their exploration activity with simple structures aimed at the construction of a small bus capable of moving forward and backward (Worksheet 4). Typical car structures can be found in Lego Mindstorms Edu software: Lego Manual (pages 8-14) or Lego Mindstorms Edu NXT Software (Common palette/ 03 Drive Forward/ Building Guide)

The buses that are to be constructed will be able to move if:

- the motors are connected by a cable to Port A, B or C,
- the micro-processors NXT is switched on by pressing the orange button,
- My files/software files/Demo is selected by pressing the orange button.

The learners will have already been familiar with the construction of the model. It is very possible that they will have already begun to compare the bus models they have constructed, drawing conclusions in respect of speed, stability and functional capacity of each model.

Fifth Teaching period

During the fifth teaching period, learners can deal with their programming part. A brief introduction regarding the micro-processor operation may take place and, once the right connections of touch and light sensors and those of the motor have been made, learners can proceed with the programming environment (Worksheet 5).

Once the construction of the model is complete, the learners can proceed with programming. In Worksheet 5 the students are asked to develop two programs, which drive and stop the bus with the use of clock and touch sensors. Learners are also asked to compare those two programs, which solve the same “problem”, so that the advantages and disadvantages of each solution may be discussed.

Sixth Teaching period

During the sixth teaching period, the students will be exploring the light sensor (Worksheet 6). This activity aims at the learners’ understanding of the sensor operation enabling them to utilize it when carrying out their programming work at the subsequent steps. They connect the light sensor to the microprocessor and move with the sensor in space. They record the readings given by the sensor in the room’s various areas: in front of the open window, on the floor, on the table, in the black color area near the window, in the black color area at a darker point of the room.

Investigation Stage (3-4 teaching periods)

At this stage, the learners are asked to determine anew the problem which they worked on during the second teaching period in the light of the experience they gained with the materials and the software and to formulate a solution. After the end of this unit, the students will be able:

- to express questions and direct their research independently;
- to design, realize and evaluate a structural work;
- to design, realize and evaluate a programming solution.

Seventh Teaching period

In the seventh teaching period learners will be re-examining their original problem on the basis of the scenario presented during the second teaching period (Worksheet 7).

They will have to determine the structural characteristics which the bus model they will construct must have, as well as the functions it must fulfil. A route upon which testing by the teams will be performed is essential to be constructed in the area where the students will be working.

The teacher may discuss with the learners the criteria whereupon they will evaluate their structural work. These may be:

Criterion	**4**	**3**	**2**	**1**
Does it work as I originally planned?				
Does it always work?				
Can it be reused many times?				
Is it easy to use?				
Is it easy to construct?				
Is it safe to use?				

(1= low, 2= moderate, 3= very good, 4 exceptional)

The list will be completed by the class. This list must stay within the class on a notice board so that everyone can consult it.

Eighth – Tenth Teaching periods

Each group of students selects a question/subject from the list of bus' functions the have already create during the previous teaching period. They will research this question and they will propose an appropriate solution to the rest of the class

Examples of questions/ subjects for research can be:

- A study on the way in which the bus can turn.
- A study on stopping and waiting for passengers, as well as on selective stopping, depending on whether there are waiting passengers at a stop.
- Means by which it can serve disabled people while in the process of stopping etc. (e.g. sound signal)
- How it parks and how it starts off at the terminus.
- How it will be moving on a pre-defined track.
- How will it deal with situations of danger/obstruction?

The learners are guided in their exploration work through Worksheet 8. At the end of each teaching period, the learners are asked to record, for five to seven minutes, important events that occurred during the course of their lesson. They are asked to record the individual problems which they faced, what they tried, how they evaluate the results. The learners, at the end of this teaching period, are asked to present the results of their work, in other words, their suggested solutions to their peers. Their programs can be printed out alongside interpretative comments and be mounted on the notice board of the classroom.

Creation Stage (1-2 teaching periods)

At this stage, the students will be asked to put together in a creative manner all the parts they have constructed so far. The purposes served by those activities are:

- Formulation of questions and exploration (experimentation);
- Testing and evaluating the results of exploration;
- Combine software programs in a coherent final solution;
- Justification of their choices.

The students will be recording useful ideas, which were suggested by their peers during the exploration at both, the construction and the programming stages. Afterwards, they will be asked to put together a complete suggestion for the robot bus which will be moving on a specific route. The course of their project is recorded in the diary just as it was at the previous stage. When their project is complete, they must draft a report in which they will:

- Describe the structural characteristics of the robot bus.
- Describe the functions which it can fulfil.
- Support their choices with arguments.
- Record the advantages and disadvantages of their proposal.

Evaluation Stage (1-2 teaching periods)

At the evaluation stage the learners are asked:

- to present the results of their project;
- to justify their choices to the rest of the class;
- to utilize evaluation criteria.

Each team is asked to present its project and participate in the discussion which will be developed within the class. Suggested topics/questions which can be discussed in the class are:

Regarding the suggested solution:

- Effectiveness of the solution
- Stability during testing
- Originality
- Simplicity
- Safety

Regarding the procedure followed:

- Did you ask questions?
- Did you try more than one solution?
- Did you support your solutions with arguments?
- Did you make good use of the new information which was given to you?

Regarding cooperation (team work):

- Did you express your ideas, opinions to the team?
- Did the contribution of the rest of the team help in the completion of the project?
- Were there situations where your opinion differed?

In a similar way we can give feedback to the learners that took part in the activity. An evaluation worksheet can first be completed by the learners and, subsequently, by the teacher in a different color.

3.2.2 Part 2: Students' Worksheets

The Bus Route Project

Name……………………………………………………….

Date…………………………………….

Worksheet 1: Public Transport

1. In your team, study the following pictures. Give a title to each one of them and write it down in the following table.

Picture 1

Picture 2

Picture 3

Picture 4

Picture 5

	Title
Picture 1	
Picture 2	
Picture 3	
Picture 4	
Picture 5	

2. Write two reasons for which you would use the bus rather than the train. Share them with the rest of your team and complete your list.

I will use the bus/ train because…

The Bus Route project

Name……………………………………………………….

Date……………………………………….

Worksheet 2:A robot bus

1. Scenario: You work for your municipality in the transport sector. The downtown area is facing very serious traffic problems, especially when the shops are open. There is a proposal to pedestrianize the downtown area and to ban private cars in it during shopping hours. In this case, the development of a reliable transport system to and from the downtown for the residents is necessary.

Write down at least three positive points and three concerns that you see in this decision.

Positive points	**Concerns**

2. The Municipality proposes to get a supply of buses which will be programmed by means of computer systems and will run without a driver. How would you describe such a robot bus? Which is the particular characteristic that distinguishes a robot from other machines?

……………………………………………………………………………………

3. Describe clearly the route that the above robot bus will be following and the functions expected to perform.

……………………………………………………………………………………..

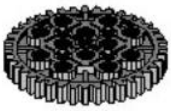
The Bus Route project

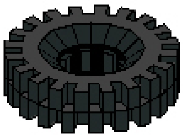
Name……………………………………………………….

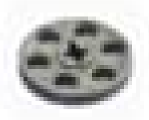
Date……………………………………….

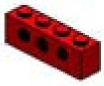
Worksheet 3: Getting to know the structural materials

1. Match the names found in the list on your right hand side with the respective items on the left hand side.









gear

pulley

tire

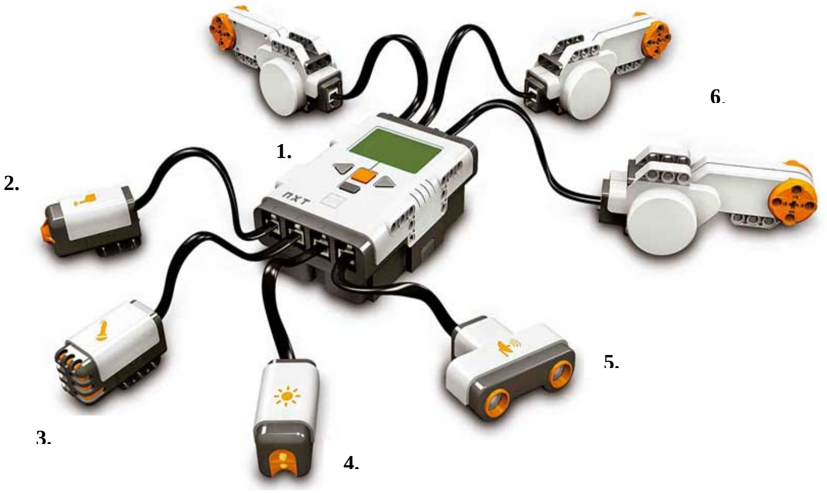
axle

beam

block

connector

2. Name the sensors, NXT brick and motors in the following picture. Please notice in which port each of these objects is connected.



3. Connect the light sensor, the touch sensor, the ultrasonic sensor, the sound sensor and a motor to the NXT brick Open the NXT brick and use the *TRY MENU*. Test the *Try-Light*, *Try-Touch*, *Try-Sound*, *Try –Ultrasonic, Try-Motor* functions. Observe the behavior of the robot in each case.

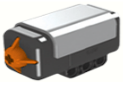
Program	**States of sensor**	**Actions**
Try-Touch	Press of touch sensor	Display of a smiling face
Try-Light		
Try-Sound		
Try– Ultrasonic		
Try-Motor		

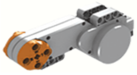
4. A “robot” is a structure that can:

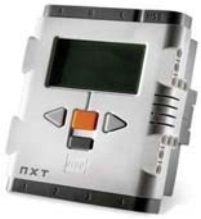
Collect data from the environment **INPUT** → Process those data and make decisions **PROCESSING** → Execute those decisions having a behaviour **OUTPUT**

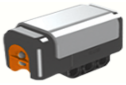
According to the above definition, match the following items with the functions they serve.

INPUT OUTPUT PROCESSING









The Bus Route project

Name…………………………………………………….

Date……………………………………….

Worksheet 4: Construction of a robot bus

1. In order to construct a robot bus it is necessary to use a NXT brick, motors and sensors, together with other structural materials. You may try to construct a small bus model like the following.

2. How many motors will be needed to make your bus?

to move forward:
backward:
to turn left and right:

3. Use additional materials to construct a small bus capable to move freely to all directions. How will you transfer motion from the motor to the wheels? Draft a sketch.

The Bus Route project

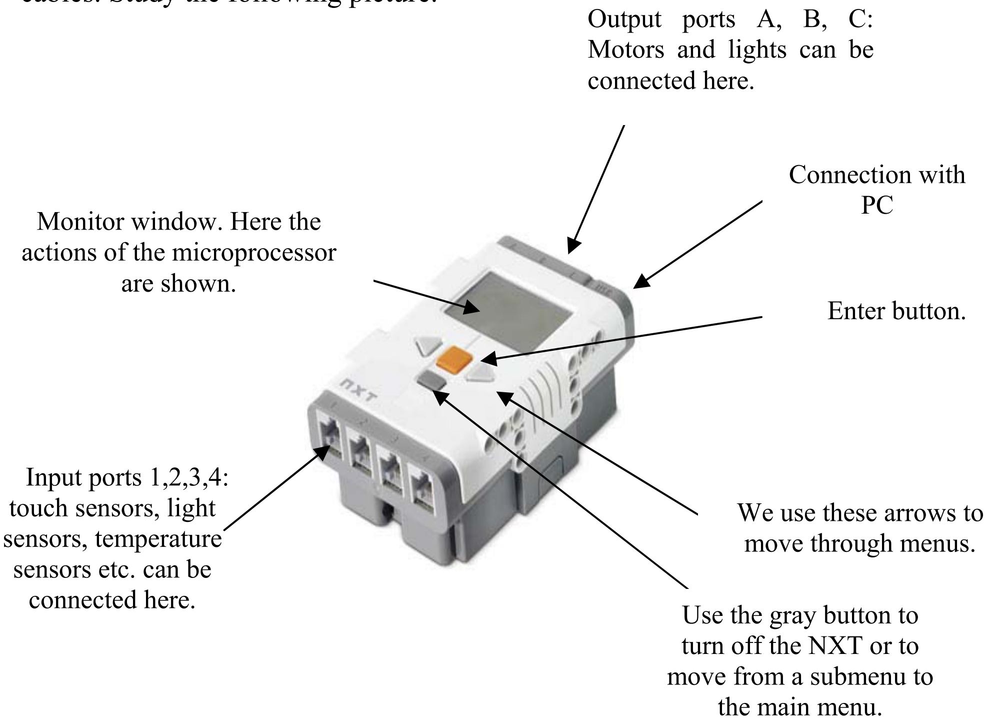
Name…………………………………………………….

Date…………………………………….

Worksheet 5: Programming a robot

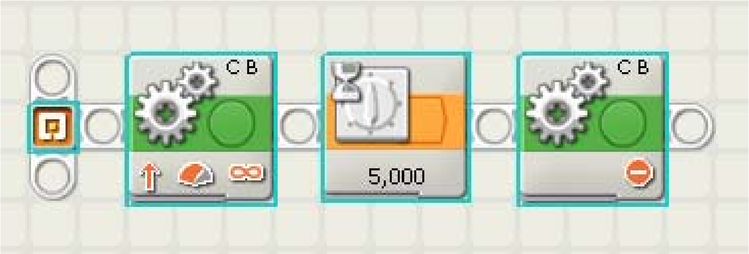
1. The NXT brick can communicate with the computer through a USB cable or a Bluetooth connection.

2. On the NXT brick, sensors, motors, lights are connected with cables. Study the following picture.


Output ports A, B, C: Motors and lights can be connected here.
Connection with PC
Monitor window. Here the actions of the microprocessor are shown.
Enter button.
Input ports 1,2,3,4: touch sensors, light sensors, temperature sensors etc. can be connected here.
We use these arrows to move through menus.
Use the gray button to turn off the NXT or to move from a submenu to the main menu.

Connect a motor, a light sensor and a touch sensor to the appropriate ports.

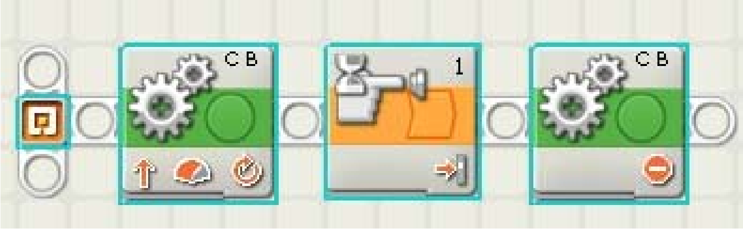
3. Open the LEGO MINDSTRORMS Education NXT software. In this program, the commands are symbolized by icons. What follows is a series of commands. What do you think is going to happen if NXT executes this program?



4. Create this program with the LEGO MINDSTRORMS Education NXT software. You will find all the icons in the Common Palette.

- Open the NXT brick.
- Connect the NXT with the PC through USB cable and download the program.
- Run the program by pressing the Enter button (orange button) 4 times.

5. Create on your computer a program which can stop the bus by means of a touch sensor. Run it.



6. Compare the use of the block *wait for time* (question 3) with the block *wait for touch sensor* (question 4) in the control of the bus's running interruption.

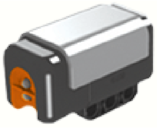
Interruption by clock	Interruption by touch sensor

The Bus Route project

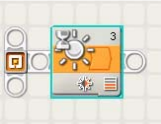
Name……………………………………………………….

Date……………………………………….

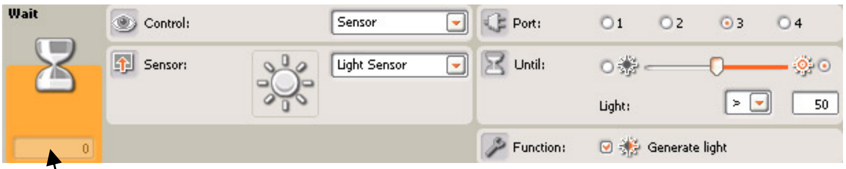
Worksheet 6: Use of light sensor



1. The Light Sensor enables your robot to distinguish between light and dark. It can read the light intensity in a room and measure the light intensity of coloured surfaces. Which port should the light sensor be connected with?



2. Connect a light sensor with the Port 3 of the NXT brick. Open Lego Mindstrorms Education NXT software and create a program with a light sensor only. Download and run the program.



Wait

Control: Sensor

Sensor: Light Sensor

Port: ○1 ○2 ● 3 ○4

Until: ○ ●

Light: > 50

Function: ☑ Generate light

0

The value shown here is the reading of the sensor

3. Move the sensor in space and observe its reading changes taking place. Complete the following table:

Position	**Value of Sensor**
In front of the window	
Towards a lit lamp	
On black color	

4. Construct and test a program which can stop the bus by means of a light sensor. Run it.

The Bus Route project

Name……………………………………………………….

Date……………………………………….

Worksheet 7:Reconstruction of the bus

In the light of the experience acquired, study again the scenario:

You work for your Municipality in the transport sector. The downtown area is facing very serious traffic problems, especially when the shops are open. The downtown area is going to be pedestrianized and private cars are to be banned during shopping hours. The transportation of the residents will be effected by programmed buses which will be run without a driver.

As a team, you are asked to write down in the following list the structural features that such a bus should have or the functions that it should perform in order to be in a position to serve the needs of the residents. Bear in mind that your town is to become famous by this original means of transport!

Subsequently, share your thoughts with the rest of the class and complete your list if necessary.

Structural features	**Behaviour**

Which are the criteria whereby you would evaluate a structure? Write down your criteria in the table given below. Then, configure a criteria list working jointly with the rest of the class.

Criteria

Based on the structural features that you have defined above, construct a bus in the light of what you have so far learned.

Describe the bus that you have constructed.

………………………………………………………………………………………

Which are the difficulties you encountered?

………………………………………………………………………………………

What experiments have you carried out?

………………………………………………………………………………………

How do you evaluate your bus using the criteria you have defined above?

………………………………………………………………………………………

The Bus Route project

Name…………………………………………………….

Date……………………………………….

Worksheet 8: Suggest a solution

1. State clearly the problem which you will be trying to solve.

………………………………………………………………………………………

2. Write down ideas that can be utilized in its solution.

………………………………………………………………………………………

3. Use your personal computer in keeping a diary of your project. Indicate :

Date:

Which problem you have tried to address?

………………………………………………………………………………………

What did you try?

………………………………………………………………………………………

How did it go?

………………………………………………………………………………………

Have you accepted or not that solution and why?

………………………………………………………………………………………

4. For the presentation of your work in class, prepare four slides. The first will present the problem/question that you researched, the second one will present the structural and functional requirements that your model had. The third slide will present the program you have created and the fourth slide will present your reflections. All the best to your presentation!!

The Bus Route project

Name…………………………………………………….

Date…………………………………….

Worksheet 9: Synthesize and create

1. Create and program a robot bus which will be serving your municipality residents in accordance with the features and functions that you have written down (consult your class notice board).

2. Write down questions and ideas regarding their solutions, utilizing, probably, the proposals put forward by the other teams as well. You may also consult the class notice board.

Questions	Proposed Solutions

3. Use your computer in order to continue your project diary. Indicate:

- Date:
- Which problem you tried to solve?
- What did you try?
- How did it go?
- Have you accepted or not that solution and why?

4. Draft a text supporting the solution you have proposed.

- Describe the construction work.
- Describe the functional capabilities that it has.
- Support with arguments your choices.
- Write down the advantages and disadvantages of your proposal, as well as your proposals for future improvement.

5. Prepare the presentation of your project.

In order to organize effectively your research work being with:

- Define the problem
- Search for information and give ideas that will lead to the solution
- Evaluate the ideas and select the most suitable
- Plan the solution on paper
- Implement, test and correct
- Evaluate based on specific criteria
- Describe the solution, argue for your choices
- Presentation

Keep in mind:

- When we do not know how to go on, we state clearly the question/problem that we are faced with.
- When something works then we deserve acknowledgement.
- We get to learn something new when something does not work.
- It is worth choosing the simplest way in doing something.
- If something makes sense to us, then it may make sense to other people as well.

Chapter 4

Teacher Training Course in Introducing Robotics in the Curriculum - The TERECoP Project Proposal

4.1 The training methodology

Authors: Kyparisia Papanikolaou, Stassini Frangou

Introduction

In this chapter we present the training methodology that we adopted through the training courses implemented in the TERECoP project and the training and evaluation materials used in the courses. In particular, the training methodology is constructivist in the sense that it is focused on learning experiences enabling trainees to build their own understanding of the technological and pedagogical perspectives of educational robotics. As far as the implementation of the courses is concerned, we adopted a combination of face-to-face meetings with online learning in order to enhance communication and collaboration among the course participants. However, each national team decided on specific aspects of the training context, such as the schedule, the trainee's profiles, and the activities developed through the course.

Research on the implementation of innovations show that it is not easy to change teacher behaviour (Fullan, 1991). When designing a teacher training course, it is useful to remember the educator's maxim "teachers teach as they were taught, not as they are told to teach". Thus, trainers in constructivist professional development sessions should better configure learning activities that teachers can apply in their own classrooms. It is not enough for trainers to describe new ways of teaching and expect teachers to translate from talk to action; it is more effective to engage teachers in activities that will lead to new actions in classrooms.

The training methodology that we adopted cite a constructivist precept: learning takes place as people build their own understanding of a subject or set of skills; the act of building understanding is more meaningful to learners than the memorization of facts or procedures. The training course is constructivistic in the sense that we provide experiences that enable the teachers to build their own understanding of the technological and educational perspectives of educational robotics and the design of robotics-enhanced activities (Papanikolaou, Frangou, Alimisis, 2008).

4.1.1 Design characteristics of the teacher training course

Constructivist professional development gives teachers time to make explicit their perceptions of learning (e.g. is learning a constructive process?), teaching (e.g. is a teacher an orator or a facilitator, and what is the teacher's understanding of content?), and professional development (e.g. is a teacher's own learning best approached through a constructivist orientation?). Furthermore, such professional development provides opportunities for teachers to test their perceptions and build new ones.

Based on the above ideas, we developed a constructivist teacher training course, during which teachers work on projects in order to build their own understanding of the technological and educational perspectives of educational robotics. In our case, teachers are trained based on the methodology for designing robotics-enhanced activities (see Chapter 3, Section 3.1.3), which is also the main focus of the training course. The particular course integrates the main principles of constructivism, constructionism and project-based learning aiming to create a powerful learning environment with a balanced whole of cooperative, learning- and teaching-focused approaches.

Learning tasks of the course were organized as small or large scale projects that encouraged trainees to design and develop their own products. Themes of the projects were either ill-defined by the trainer or freely chosen by the trainees. In that way, trainees had the opportunity to express their own ideas and to select themes close to their professional needs and personal interests. Therefore, trainees were ‘personally connected’ with their projects, a fact that forms an additional requirement for invaluable creative work and effective learning (Resnick, 1991).

The *active involvement* of the trainees in all the parts of the course was of high importance. A teacher training course can contribute to the professional development of teachers by forming relations between teachers’ existing experiences and the proposed new educational technologies. So, from the beginning of the course, trainees were encouraged to express themselves and to participate in all activities of the course through work and discussions in small groups, presentations in plenary sessions and publications on the e-environment. In this way, current ideas, beliefs and attitudes of the participants were made explicit and evaluated within the constructivist approach.

Throughout the course, all trainees were working autonomously. The role of the trainer was to facilitate the learning process by creating an interesting and stimulating learning environment: giving feedback at regular intervals, raising interesting questions, guiding the research concerned and synthesizing ideas. Trainees, on the other hand, were responsible for their work, they could follow their own path in their exploration and could develop their own ideas. They were supported in their work by appropriate resources, such as worksheets, representative examples and user guides. Diaries were kept where the activities of each session were recorded with a view to addressing the most important issues.

Finally, a constructivist learning environment is based on *cooperation*. Social interaction within small groups generates a fruitful learning environment, where ideas are expressed, discussed and developed. So, most of the learning tasks were performed by trainees working in small groups.

To enhance the sense of community and promote collaboration through the course an e-workspace was also maintained. Through this e-workspace, teachers are able to ask questions easily and efficiently outside the classroom, contact and ask trainers/peers to clarify doubts and problems, exchange ideas and share resources. Moreover, trainers are able to make announcements to the class and provide resources and support in a more efficient way. Finally, the e-workspace was used as a common 'meeting' space that promotes a sense of community among the teachers and of contiguity after the completion of the training course.

4.1.2 Constructivist views of learning and teaching

The main principles of constructivism, constructionism and project-based learning are building elements of the training course. These principles and their implications to teaching are briefly introduced in this section.

Constructivism. Central idea of constructivism is that human learning is *constructed*, that learners build new knowledge upon the foundation of previous learning. This view of learning sharply contrasts with one in which learning is the passive transmission of information from one individual to another, a view in which reception, not construction, is the key. Constructivism has roots in philosophy, psychology, sociology, and education. But while it is important for educators to understand constructivism, it is equally important for them to understand the implications this view of learning has for teaching and teacher professional development.

Two important notions orbit around the simple idea of constructed knowledge (Hoover, 1996). The first is that learners construct new understandings using what they already know. Learners come to learning situations with knowledge gained from previous experience, and prior knowledge influences what new or modified knowledge they will construct from new learning experiences. The second notion is that learning is active rather than passive. Learners confront their understanding in light of what they encounter in the new learning situation. If what learners encounter is inconsistent with their current understanding, their understanding can change to accommodate new experience. Learners remain active throughout this process: they apply current understandings, note relevant elements in new learning experiences, judge the consistency of prior and emerging knowledge, and based on that judgment, they can modify knowledge.

Key assumptions of the constructivist perspective are summarised below (Driver and Bell, 1985):

- What the students currently believe, whether correct or incorrect, is important;
- Despite having the same learning experience, each individual constructs on individual meaning;
- Understanding or constructing a meaning is an active and continuous process;
- Learning may well involve some conceptual changes;
- Learning is not a passive process, but active and depends upon the students taking responsibility to learn.

Constructivism has important implications for teaching that should be carefully considered when designing teaching and learning (Hoover, 1996):

- Teaching cannot be viewed as the transmission of knowledge from enlightened to unenlightened; constructivist teachers do not assume the role of the "sage on the stage" but rather "guides on the side" who provide students with opportunities to test the adequacy of their current perceptions;
- If learning is based on prior knowledge, then teachers must notice that knowledge and provide learning environments that exploit inconsistencies between learners' current perceptions and the new experiences. This challenges teachers for they cannot assume that all children understand something in the same way. Further, children may need different experiences to advance to different levels of understanding.
- If students must apply their current perceptions in new situations in order to build new knowledge, then teachers must engage students in learning, bringing students' current perceptions to the forefront. Teachers can ensure that learning experiences incorporate problems that are meaningful for students, not those that are primarily important to teachers and the educational system. Teachers can also encourage group interaction, where the interplay among participants helps individual students become explicit about their own understanding by comparing it to that of their peers.
- If new knowledge is actively built, then time is needed to build it. Ample time facilitates student reflection about new experiences, how those experiences line up against current perceptions, and how a different understanding might provide students with an improved (not "correct") view of the world.

This constructivist view of learning also influences the role of teachers. The main task that teachers are assumed to perform, according to constructivism, is no longer the transmission of knowledge, but the facilitation and coaching of learning (Korthagen, Klaassen, & Russell, 2000).

Constructionism. What is the difference between Piaget's constructivism and Papert’s “constructionism”? As Ackermann (2001) suggests “Beyond the mere play on the words, I think the distinction holds and that integrating both views can enrich our understanding of how people learn and grow. Piaget’s constructivism offers a window into what children are interested in and able to achieve, at different stages of their development. Piaget suggests that children have very good reasons not to abandon their worldviews just because someone else, be it an expert, tells them they are wrong.

Papert’s constructionism, in contrast, focuses more on the art of learning or ‘learning to learn’ and on the significance of making things in learning. Papert is interested in how learners engage in a conversation with [their own or other people’s] artifacts, and how these conversations boost self-directed learning and, ultimately, facilitate the construction of new knowledge. He stresses the importance of tools, media, and context in human development. Integrating both perspectives illuminates the processes by which individuals come to make sense of their experience, gradually optimizing their interactions with the world.”

Moreover, Papert also approaches the issue of relevance and emotional attachment with an observation that by adding new objects such as “cybernetic construction kits” for LEGO/Logo, children might “want to learn it because they would use it in building” (Harel and Papert, 1991).

Papert (1980) and later Resnick (1994) lay out a vision for learning-by-design which enables students to learn by participating in the design of digital environments, such as Digital Manipulatives (Resnick, 1994) and Logo (Papert, 1980). Within the constructionist framework, the learner is not a passive recipient of information; rather s/he is an active participant in the learning process, working to construct knowledge through experience, thus shifting the control of digital learning into the hands of learners. Papert (1980) describes four learning-by-design principles:

1. Individuals are active learners and control their own learning process;
2. Individuals create concrete, tangible evidence (artefacts) that reflect their understanding;
3. Artefacts are shared collectively as well as reflected upon individually to extend one’s understanding;
4. The learning problems and contexts are authentic, that is, they focus on solving a practical problem.

As an extension of constructivism, the constructionist approach involves learners building knowledge and meaning through the construction of something *external* or *shareable* (Papert, 1991). Furthermore, such a process also provides a motivating context for students to learn the subject matter and content and test their knowledge. Just as maintained by Puntambekar and Kolodner (2005) that when students are engaged in cycles of designing, evaluating, and redesigning, they have also the opportunity to confront their understanding and misunderstanding of concepts (p. 185). This means that the learner is a designer, rather than just the receiver of designed materials. The teacher is thus charged with designing/creating a learning environment within which the learner can explore and create. Facilitators later serve as advisors to learners, who are dealing with their own needs within the environment (Hannafin & Hill, 2002).

Papert (1991) described the value of construction as actively engaging participants in creating something that is meaningful to themselves or to others around them. Constructionism urges learners to build a context for learning through community-supported collaborative construction (Bruckman, 1998). In this way, a constructionist learning environment can provide learners a self-motivated and peer-supported environment

Educational robotics. Some general principles may be derived about how constructionism may be applied to educational robotics and the role of teacher in this context:

- Educational Robotics is not taught to add new competences to traditional curricula; actually, it is not taught at all. It acts as a problematic challenge to both, teachers and students, to address practical problems, where other competences can be exploited to find effective solutions that are hereafter used as argument of discussions and as source of new problems.
- Guidelines on using educational robotics can refer to specific (programming) languages and robotic architectures (kits), but they should not strictly depend on them. The goal should be how to instil a 'Logo spirit' when constructing, programming and moving robots: experimentations with different languages and robots could result in much more methodological validity. Moreover, suggestions (and not compulsory recipes) on how conducting discussions and on possible improving the given solutions must be supplied together with basic materials, such as constructing instructions, program skeletons etc.
- Even in robotics there is no "right/wrong" dilemma: the learning activity proceeds step by step refining the problem specs and improving the more or less acceptable found solutions. It will be very common that the teacher has to afford unpredictable or at least unknown situations during which he/she is co-learner with his/her students. These situations will spontaneously arise during the lab activities because of the nature of robotics itself, and they give new opportunity to teachers and students to try out their skills and eventually their 'believed' limits.
- If we want to emphasize the "constructional side" of digital technology in spite of its "informational side", educational robotics is a perfectly balanced synthe-

sis of “material” (the robot) and “immaterial” (the program) construction. In this sense, other activities, such as exchange of experiences and guidelines through the Internet, can be allowed without the risk of becoming prevailent over the mainstream activity.

Self-awareness, self-efficiency, self-regard, self-assessment, self-rewarding should be the focus in designing experiences with educational robotics. The role of the mediator is important as a co-learner during the developing and problem-solving phases.

4.1.3 The e-workspace

In order to enhance class communication and cooperation during and beyond the face-to-face meetings, we developed an e-workspace that we maintained throughout the course. To this end, we used the open source e-class platform offered by the Greek Universities Network and the University of Athens (http://eclass.gunet.gr).

Trainers used the ‘virtual class’

- to *provide* trainees with *resources* (course content, worksheets, presentations) and *support* such as timely information about the course content & scheduling, useful resources and links, on-time support through the public areas of ‘announcements’ and ‘forums’,
- to *promote a sense of community* among the members of the class (trainers and trainees) providing opportunities for communication/collaboration and sharing of resources during and beyond the face-to-face meetings.

The e-class was organized to support communication and collaboration at two levels: at *class* and *group* levels. To this end, we used *public areas* for all the members of the class with different rights for trainers and trainees, like the ‘Announcements’ area that permits trainees to make announcements to the class, the ‘Documents’ area that allows the trainers to upload content, whilst trainees can only download the available files, the ‘Agenda’ area that allows the trainers to describe the course structure with time and session information, the ‘Links’ area where the trainers may suggest interesting Internet sites to the trainees, the ‘Forums’ area for discussing topics, where trainers and trainees are allowed to create discussion topics and submit messages.

We also arranged *private areas for each group*, where trainees could upload their products when working with activities (such as programs or texts, the group diary kept at the end of each session, the materials of their own project), discuss topics, and exchange e-mails. This area was also accessible to trainers. In several cases, the trainees could share their group products if these were copied in the public area.

During the course, we used the *public areas as tools* for *administration purposes*, for example, for providing the course content and worksheets before each session and timely information about the course organization or each session, as well as the *public and private areas* for *teaching purposes, promoting reflection and social interaction.* For example, we used the public forum to organize a ‘helpdesk’ where everyone could submit a problem or provide a solution, to stimulate trainees introduce themselves and share their expectations, to make trainees express themselves in specific discussion topics, share and reflect on their peers’ ideas, experiences, and perspectives - e.g. trainees at the end of each session submit a comment on their learning experiences of the day or suggest interesting and useful links on the Internet.

References

Ackermann E. (2001): Piaget’s constructivism, Papert’s constructionism: What’s the difference? Future of Learning Group Publication.

Brooks, J.G., Brooks, M.G. (1993): In Search of Understanding: The Case for Constructivist Classrooms. Alexandria, VA: Association for Supervision and Curriculum Development.

Bruckman, A. (1998): Community Support for Constructionist Learning. Computer Supported Cooperative Work 7(1/2) 47-86.

Driver, R., Bell, B. (1985): Students' thinking and the learning of science: A constructivist view. School Science Review. pp. 443-456.

Fullan, M. (1991): The New Meaning of Educational Change (2nd edition). Toronto: The Ontario Institute for Studies in Education.

Hannafin, M. J., Hill, J. R. (2002): Epistomology and the design of learning environments. In R. A. Reiser, & Dempsey, J. V. (Ed.), Trends and Issues in Instructional Design and Technology. New Jersey: Merrill Prentice Hall.

Hoover, W. A. (1996): The Practice Implications of Constructivism. SEDLetter, Vol 9, No. 3.

Korthagen, F., C. Klaassen, T. Russell (2000): New Learning in Teacher Education. In: R.J. Simons, J.L.van der Linden & T.M. Duffy (Eds.). New Learning. Amsterdam: Kluwer Academic Publishers. P. 243-259.

Papanikolaou, K., Frangou, S., Alimisis, D. (2008): Teachers as designers of robotics-enhanced projects: the TERECoP course in Greece, In proceedings of the SIMPAR 2008 conference/Workshop “Teaching with robotics: didactic approaches and experiences”, University of Padova. Available at http://www.simpar-conference.org/

Papert, S. (1991): Situating Constructionism. In S.Papert and I.Harel (eds.) Constructionism. Ablex Publishing Corporation.

Harel, I., Papert, S. (1991) (eds.) Constructionism. New Jersey: Ablex Publishing Corporation.

Papert, S. Mindstorms (1980): Children, Computers, and Powerful Ideas. N.Y.: Basic Books.

Puntambekar, S., & Kolodner, J. L. (2005): Toward implementing distributed scaffolding: Helping students learn science by design. Journal of Research in Science Teaching, 42 (2), 185-217.

Resnick, M. (1991): Xylophones, Hamsters, and Fireworks: The Role of Diversity in Constructionist Activities, in S.Papert I..Harel (ed.) Constructionism Ablex Publishing Corporation, 151-161

Resnick, M. (1994): Turtles, Termites, and Traffic Jams. Cambridge, MA: MIT Press.

4.2 Course curriculum outline

Authors: Stassini Frangou, Kyparisia Papanikolaou, Dimitris Alimisis

In this section, we present a pilot training course for teachers, developed according to the methodology described in section 4.1. Firstly, we present the overall aims and objectives of this course, then we give a brief outline of the course curriculum and, lastly, we give a detailed description of the proposed teaching activities. Worksheets and other training materials can be found in the next section 4.3.

4.2.1 Overall aims and objectives

The overall aim of the course is to provide opportunities for teachers to examine how robotics technologies can be used to promote a constructivist-constructionist approach to learning under a co-operative and collaborative frame of work. The implementation of robotics-enhanced constructivist teaching and learning practices demands that teachers assume a new role. This means that opportunities, such as exposure to a number of critical examples, experience in designing computer-based robotics activities and integrating same in their classroom practice in constructivist ways, are of great priority. The goal is teachers to be convinced by their own personal experience about the potential of robotics technology as a learning tool.

In this course, we assume that technology alone cannot affect teaching practices. Our curriculum design follows an innovative constructivist perspective with an emphasis on aligning computer and robotics technology with subject matter and learners' needs for the purpose of constructing meaning in social learning environments. In such learning environments, the focus is not on the individual, but on interactive systems that include individuals interacting with each other, instructional materials, subject matter, and tools. Computer-based robotics is an innovative technology that can create a rich interactive environment encouraging constructivist learning.

Specific objectives: More specifically, our objectives are:

- to familiarise trainees with an appropriate robotics-based learning environments (Lego Mindstorms system) and a set of critical examples and activities that can support constructivist teaching and learning in science and technology subjects,
- to enable trainees to use robotics technology in a way that can contribute to the realisation of:
    - meaningful learning based on students' own team work with teaching materials,
    - authentic learning using learning resources of real-life, occupational situations or simulations of the every day phenomena,

- social learning through the use of e-learning classes,
- active-reflective learning, working on experiments or problem-solving and using available resources selectively, according to their own interests, search and learning strategies,
- project-based learning seeking solutions to real world problems, which are based on a technology-based framework,

- To create a community of practice between educators and teachers with a view to making easy and sustaining teachers' professional development in using robotics tools to support their students' learning by active exploration and social construction of new knowledge.

The expected impact is teachers to be trained in a way that robotics technology-enhanced learning will play an important part in their future work as teachers or professional educators. Trainees are expected:

- to develop innovative collaborative strategies in their classes and promote the development of e-learning communities,
- to select exploratory learning activities that can support social constructivist teaching and learning,
- to use the proposed tools in real classroom situations,
- to design, build and program their own robotic models and develop their own projects for their students.

4.2.2 Outline of the training course curriculum

The pilot training course is organized in six modules and its total duration is 36 teaching periods (of 45 minutes). It provides an initial training (a) in constructing and programming a robot and (b) in developing robotics projects for students. In particular, the six modules are:

1. Introduction to the course.

2. Building a **'Didactic contract'** aimed at presenting to trainees the rationale and the means which are going to be used during the course in question.

3. Robotics as learning object aimed at introducing basic constructing and programming features of robotics technology

4. Theoretical framework embracing learning theories and the appropriate background for designing robotics-enhanced projects.

5. Introduction to the **methodology for developing robotics projects** and designing such projects.

6. Evaluation of the course based on semi-structured interviews and questionnaires.

An outline of the course and the estimated duration of each module can be found in table 4.2.1.

***Table 4.2.1** Course outline*

Training Course Curriculum Outline		
module	**title**	**Duration (teaching periods)**
C.1	**Introduction**	1
C.2	**'Didactic contract'**	1
C.3	**Robotics as learning object**	**13**
C.3.1	Introduction to LEGO NXT and sensors	2
C.3.2	A first approach to construction of robots	3
C.3.3	A first approach to programming robots	3
C.3.4	"The cat, the mouse and the master" project	3
C.3.5	The "data logger" project	2
C.4	**Theoretical framework**	**4**
C.4.1	Constructivism and Constructionism	2
C.4.2	Why robotics in education?	1
C.4.3	Project-based learning	1
C.5	**Methodology for developing robotics projects**	**14**
C.5.1	Model for organizing robotics projects	1
C.5.2	An example of a robotics project: "BusRoute"	3
C.5.3	Working on a new robotics-enhanced project	7
C.5.4	Presentation and evaluation of the trainess' projects	3
C.6	**Evaluation of the course**	**3**

The exact timetable of the course can be formed according to the needs of the trainees. It can be taught over a period of 4 weeks (9 teaching periods per week) or over 3 weekends (12 teaching periods every weekend) - one weekend per month. In order to enhance class communication during and beyond the face-to-face meetings, we suggest maintaining through the course an e-workspace (find a description of the e- space at section 4.1). A useful tool in collecting information about the work done by trainees during the meeting, their thoughts or other issues concerning the implementation of the course is the diary. Each group spends a few minutes at the end of each session to write a few lines on an electronic diary. Suggested questions to be included in their diaries are:

- What did you do during this session?
- What was the best thing that happened to you during this meeting?
- What was the worst thing that happened to you during this meeting?
- What are you thinking to change next time?
- Other Comments

Finally, most of the activities of the course are carried out in small groups. In order to ensure that groups can work independently we provide worksheets and other resources which can be found in section 4.3.

4.2.3 Proposed way of course implementation

C.1 Introduction (1 teaching period)

This teaching period aims at 'breaking the ice' between trainees and trainers, at helping trainees relax and get to know each other's names and personal/professional information and at identifying individual learning needs and goals, expectations and possible learning difficulties.

In particular, the trainer(s) introduce(s) himself/themselves. Then, the trainees are asked to form groups of 4-5 people and each one to introduce himself/ herself to the rest of the group in 2-3 minutes. Additionally, trainees are asked to provide personal/professional information, to express individual learning needs and goals, expectations and possible learning difficulties. Lastly, one representative from each group briefly introduces the members of his/her group to the plenary. Also, trainees and trainers are asked to post a message on a relevant topic at the discussion forum of the e-class briefly introducing themselves (forum-topic “Class members”)**.**

Alternatively, after this introduction, the trainees in groups of two interview each other for 5 minutes and then introduce themselves in 2 minutes to the whole class.

C.2 Didactic contract (1 teaching period)

The aim of this module is:

- to ensure a consensus between trainer and trainees on the training objectives, content and methods
- to generate interest in the topic of the training

- to make arrangements necessary for the smooth running of the course
- to give a few key ideas on pedagogical issues

The trainer presents the overall aim, the specific objectives of the course, the training methodology and the expected training results. The trainees are asked to express their own expectations, opinions, suggestions and ideas. This module finishes with an agreement between trainer and trainees on the aforementioned issues (and on everything else that may emerge in the training class) leading to the formulation of a "didactic contract". This “didactic contract” is uploaded by the trainer in the Documents area of e-class. Some relevant papers become available for trainees through the e-class and are suggested for reading.

C.3 Robotics as learning object

This module is focused on the introduction of the materials included in the Lego Mindstorms Education NXT kit, and the Lego Mindstrorms Edu NXT software. It is organized in five parts. In the first part, trainees are familiarized with the materials which they are going to use in the construction of their model. In the second part, they get involved in constructing a robot car. The remaining parts of this module are introducing the basic programming feature of the software.

C.3.1 Introduction to Lego Mindstorms NXT brick and sensors (2 teaching periods)

Trainees form small groups of 3 or 4 members and one Lego Mindstorms Education NXT kit is given to each group. They are working in groups and they identify the sensors, the motors and the construction parts, such as blocks, axles etc. of their kit. The trainer makes a brief introduction to NXT brick functions and then the groups are asked to experiment with the touch sensor, light sensor and servomotor in order to become familiarized with them and their parameters by carrying out the activities of the Worksheet C.3.1. At the end of this section, a discussion about the technical characteristics of each sensor takes place in plenary.

C.3.2 A first approach to construction (3 teaching periods)

During the second part, trainees in groups construct a car- robot with two motors. To this end, they use instructions included in the official guide and the Lego Mindstorms Edu NXT software. They are also introduced to the Lego Digital Designer software(http://ldd.lego.com/). At the end of this part, a discussion-evaluation of their experiences through the construction of the robot-car takes place. The trainers and trainees agree on a set of criteria for evaluating robotic constructions.

C.3.3 A first approach to programming: Moving around (3 teaching periods)

The third part is focused on the Lego NXT programming environment and the development of virtual models that guide robots with varying configurations, i.e. motors’ activation using basic programming blocks within the NXT software.

The trainees, working in groups, undertake specific introductory activities to the programming environment of Lego Mindstorms Education NXT. The initial project is to design a program that moves a robot along the sides of a square. To this end, an appropriate worksheet is given with specific instructions (Worksheet C.3.3).

Then, the trainees develop their first program and investigate the relation between power of motor and speed of the car robot they have already constructed. The factors which influence the final speed of the car robot is discussed in plenary. Then, they are asked to investigate left and right turns with both, ‘move’ and ‘motor’ blocks and, finally, they develop their own blocks for left turn of 90^o and right turn of 90^o. Each group upload the blocks they develop through this activity on the private documents’ area of the group in the e-class. Then, the groups are asked to make their robot move on a square path (final programs are also uploaded).

Additional experimentation can be conducted by the trainees in order to make the robot car turn left or right at an angle. During these activities all groups will create programs with blocks such as ‘move’, ‘motor’, ‘record’, ‘loop’, while they will have also defined their own blocks. Advanced functions like ‘record/play’ can be also introduced to trainees.

C.3.4 The “cat, the mouse and the master” project (3 teaching periods)

The “cat, the mouse and the master” project is an activity introducing basic programming structures and statements of the Lego Mindstorms Education NXT programming environment. Initially, a mock up with black spots is put on the ground simulating the area where the cat is moving - each black spot corresponds to a mouse!. The groups should adapt their robotic construction in order to make it work on the mock up as a cat running after a mouse. Three activities that gradually introduce trainees to different programming concepts of varying difficulty and complexity are performed. Each activity sets a specific challenge-problem to the trainees:

- At first, they should make the cat run after the mouse and stop when it reaches a black area (the mouse!) using a light sensor, the loop block, and developing their own blocks (Worksheet C.3.4.1),
- Then, the cat’s behaviour should be ‘extended’ to be able to stop for a while and make a sound when the master touches her. To this end, the cat robot should be equipped with a touch sensor. Trainees should also extend the program using condition blocks, and blocks like Display, Sound, Wait For (Worksheet C.3.4.1),
- Lastly, they should use variables in order to make the cat move on a spiral path (Worksheet C.3.4.2, Appendix 4).

At the end of each activity, trainees are invited to present their work and discuss with others their ideas. Different solutions are compared.

C.3.5 The 'data logger' (2 teaching periods)

This part is aimed at enabling trainees to use the data logging functions of the software. These functions are very useful in all science projects. Trainees are asked to study a ready-made program, to collect time and distance data from a moving robot and, finally, to design a graphical representation of the corresponding data that give information about the motion of the robot (Worksheet C.3.5,).

The Lego Mindstorms Education NXT v2.0 has an extra feature for real time and remote data logging.

C.4 Theoretical framework

C.4.1 Constructivism and Constructionism (2 teaching periods)

The first activity of this module is to discuss the question "How the use of: robotics technology in school class could change the traditional teacher-centred teaching model?"

The question can be discussed through a brainstorming activity where the trainer encourage trainees to express freely believes and attitudes on that issue and utilizes the experience and creativity of all participants. The trainer summarizes all the ideas which emerge from the brainstorming.

Then, the trainees are separated in groups of 3-4 people and they are asked to discuss issues concerning constructivism and constructionism through Ackermann's paper "Piaget's constructivism, Papert's constructionism: What's the difference?" Each group presents a part of the paper to the plenary session and the trainer makes a synthesis of trainees' answers and presents his/her own ones (if they are different from those presented by the trainees) focusing on principles such as :

- the knowledge background and culture of the learner plays an important role in learning,
- learners construct their own understanding and do not simply mirror and reflect what they read,
- learning is an active, social process,
- the necessity for collaboration among learners, in direct contradiction to traditional competitive approaches,
- learning is most effective when part of an activity the learner experiences as constructing a meaningful product,
- constructionist learning involves students in drawing their own conclusions through creative experimentation and making social objects,

- teachers have to adapt to the role of facilitators and not transmitters of subject matter. The constructionist teacher takes on a mediational role rather than adopting an instructionist one. Teaching "at" students is replaced by assisting them to understand problems in a hands-on way.

After that discussion, the trainees are encouraged to write their opinion on the same topic in the forum of their e-class.

C.4.2 Why robotics in education? (1 teaching period)

The trainees are separated in groups of 4-5 persons (the synthesis of the groups might be different from the previous one). Each group is asked to read a part of the article: Resnick M (2002). Rethinking Learning in the Digital Age. In *The Global Information Technology Report: Readiness for the Networked World*, edited by G. Kirkman. Oxford University Press, which bas been available to the trainees through the e-class. Then, they are asked to upload a summary of this paper and their comments in the forum of e-class. A presentation of their opinions is also made to the plenary session. The trainer synthesizes and summarizes all relevant ideas and adds his own comments on the educational value of robotics.

C.4.3 Project-based learning (1 teaching period)

The trainees are separated in groups of 4-5 people (the synthesis of the groups might be different from the previous ones). They are invited to study the paper Carbonaro M., Rex, M. & Chambers, J. (2004). Using LEGO Robotics in a Project-Based Learning Environment. *The Interactive Multimedia Electronic Journal of Computer-Enhanced Learning*, 6(1). Retrieved 22/9/2008, from http://imej.wfu.edu/articles/index.asp and then to write down and present three main advantages of the project-based learning against the traditional teacher-centered teaching model.

The trainer makes a synthesis of trainees' answers and presents his/her own ones (if they are different from those presented by the trainees) focusing on helping trainees to recognise the educational advantages of project-based learning as a model for classroom activities that shifts away from the classroom practices of short, isolated, teacher-centered lessons and, instead, emphasizes learning activities that are long-term, interdisciplinary, student-centered, and integrated with real world issues and practices.

C.5 Methodology for developing robotics projects

Projects are long term activities that bring together ideas and principles from different subject areas. Teaching and learning through projects seems to be a complex and demanding activity for teachers and students. As a part of this training course this module aims to provide the trainees with “a hands on” experience in designing robotics projects. In particular, during this module the trainees will:

- Reflect upon basic features of a robotics project which is developed according to constructivist and constructionist principles,
- Study an example of a project developed according to the theoretical framework proposed in previous lessons (five stages),
- Analyze each stage of the project according to the type of activities performed by teacher and student,
- Apply the same model to a subject of their interest and develop their own project.

The module ‘Methodology for developing robotics projects’ may cover 14 teaching periods.

C.5.1 Model for organizing a robotics project (1 teaching period)

During this teaching period trainees are working in small groups (4-5 people) on Activity 1 of the worksheet C.5.1 for 20 minutes.

Activity 1 **20 minutes**

(Working in groups of 4)

In previous sessions of this course, we have discussed thoroughly the constructivist learning approach and its implications in teaching. Concerning Robotics in Education, we have illustrated features of learning by constructing artifacts and we have discussed the constructionist approach in teaching and learning.

1. Make a list of seven features that a robotics project should have in order to serve constructivism and constructionism perspectives of teaching and learning.

2. Be prepared to present your list to the rest of the class.

After the end of this activity, trainees present their list to the rest of the class and the trainer summarizes their answers (20 minutes). If necessary, the trainer can add more features in the list. (Additional information can be found in Appendix 2 Project –Based learning: Important features, section 4.3).

Then, the trainer presents, with a slide show, the five stages of a project. This presentation can be enriched with a short description of each stage. Trainees can also be asked to combine the list of features they created during Activity 1 with the five stages mentioned above. Stages and their description can be found in Chapter 3.1 and in Appendix 3 of section 4.3.

C.5.2 An example of a robotics project: "The BusRoute" (3 teaching periods)

At this stage, trainees are going to work on a project example in order to explore and elaborate on its main stages through an authentic experience. As a project example, we use here the project "The BusRoute" that we present in section 3.2. A brief outline of "The BusRoute" project can be seen in table 4.2.2

Table 4.2.2. *Outline of "The BusRoute" project*

Stage	Duration (teaching periods)	Teaching Theme	Worksheets
Engagement stage	2	Public Transport A robot bus	Worksheet 1 Worksheet 2
Exploration stage	4-5	Getting to know the structural materials Construction of a robot car Programming a robot Use of light sensor	Worksheet 3, Worksheet 4, Worksheet 5, Worksheet 6
Investigation stage	2-3	Construction of the bus Suggest a solution	Worksheet 7 Worksheet 8
Creation stage	1-2	Synthesize and Create	Worksheet 9
Evaluation stage	1-2	Presentations & Discussion	

Trainees, in small groups, may go through the "The BusRoute" project stage by stage (or alternatively, each group undertakes one stage). They study the description of each stage and they carry out the activities of the relevant worksheets (relevant materials can be found at section 3.2). Then they complete the activity 2 of the worksheet C.5.2. In particular they complete the following table with ideas, interesting elements they found in the project and new ideas that they think as important.

Stage	Teaching Strategies-tools	Students activities	Teacher activities

Table 4.2.3 *Strategies, tools and activities in each stage of the project.*

Stage	**Teaching Strategies - tools**	**Students' activities**	**Teacher activities**
Engagement stage	Stimulate interest with pictures, videos or newspaper articles. Make real world connections. Discussions in plenary session. Discussions in small groups. Handouts.	Share experiences with peers and teacher. Provide ideas for creation of questions and artifacts. Define the project. Determine collaboratively evaluation criteria.	Provide open ended activities. Facilitate discussions. Summarize.
Exploration stage	Guided explorations. Introduction of new tools, skills and materials. Handouts.	Follow instructions. Experimenting. Observe. Gather information. Conclude. Compare.	Create a learning environment. Give assignments. Provide structured set of inquiry steps for learner to follow. Reinforce learning.
Investigation stage	Formulate the problem and analyze it. Choose one problem to investigate (one for each group). Diary.	Ask and refine questions. Refine evaluation criteria. Select appropriate tools. Planning. Experimenting. Apply knowledge in new situation. Evaluate. Communicate ideas and findings to others.	Challenge students by asking questions. Mediate group work.
Creation stage	Select resources. Create final artifacts. Diary.	Synthesize. Collaborate and negotiate. Evaluate. Draw conclusions.	Ensure individual and group learning.
Evaluation stage	Presentation. Peer evaluation. Evaluation Rubrics.	Present final product. Evaluate. Compare. Give and take feedback.	Facilitate group evaluation and self- evaluation.

Finally, they share their thoughts in a plenary session. The trainer synthesizes all ideas in a common table and uploads this file to the documents area of the e-class. This table may look like Table 4.2.3.

C.5.3 Working on a new robotics-enhanced project (7 teaching periods)

During this session, trainees are expected to use the methodology for designing robotics-enhanced projects in order to develop a new project**.** Trainees, working in groups, are encouraged to use materials created and experiences gained during the previous sessions of the course in order to develop a new project.

This session is the “creation stage” of the pilot course, so trainees should be gradually become capable, through the course, of contributing to the work at this stage. To that direction, a very helpful activity that can be carried out at any session of this course is to ask trainees to search the web for interesting robotic constructions and share their findings with other members of the class through the e-class space.

As an introduction to the work of this section, trainees in groups are asked to write and upload an idea (or ideas) of a project they would like to develop further. The ideas of each group are presented to plenary sessions and all participants can contribute with useful comments.

The trainer keeps giving the trainees some general guidelines helping them to describe the work expected from them (worksheet C.5.3). Then, the trainees are asked to agree on a set of criteria/rubrics according to which their projects are going to be evaluated. An example of rubrics can be found in worksheet C.5.4.

Finally, each group takes the responsibility to organize its work in the laboratory or at home and submit their projects on time.

C.5.4 Presentation and evaluation of the projects (3 teaching periods)

During this session, trainees present their work to the rest of the classroom. Presentations consist of: (a) demonstration of the function of the robot involved, (b) description of the project with emphases on the exploration and investigation activities expected to be preformed by students.

The work of each group is commented and evaluated by (a) the group (self evaluation),(b) any other group of trainees (peer evaluation), (c) the trainer (or trainers). The evaluation can be done by the rubrics agreed on at previous stages or in any other way.

C.6 Evaluation of the course (3 teaching periods)

The evaluation of the course can be based on:

- **Group diaries:** After the end of each session, each group keeps notes in a text file. In this diary, we expect to find information about: activities carried out during the session, time taken for each activity, the results of each activity, the collaboration among trainees, the robots they created and the proposed programming features.
- **Electronic portfolios** of the works produced by each group through the pilot course.
- **Structured interviews** conducted in groups at the last meeting of the course (Worksheet C.6.2),
- Individual **questionnaires** (Worksheet C.6.1)

4.3 Training materials

Authors: Stassini Frangou, Kyparisia Papanikolaou, Dimitris Alimisis

Introduction

In this section you can find all the worksheets used during pilot training courses. Each worksheet can be identified by the number of the module that belongs (C.3.1). The way to use each worksheet is described at the corresponding module of section 4.2.

Table 4.3.1. *Course outline*

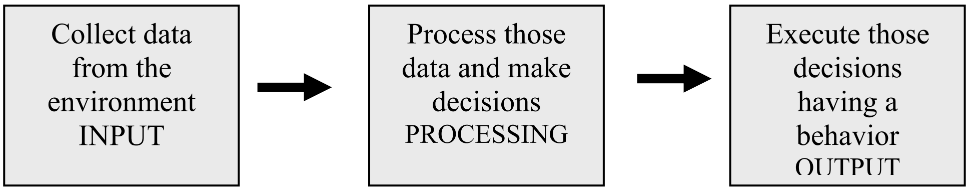
Training Course Curriculum Outline			
Module	**Title**	**Hours**	**Materials**
C.1	**Introduction**	**1**	
C.2	**'Didactic contract'**	**1**	Appendix 1
C.3	**Robotics as learning object**	**13**	
C.3.1	Introduction to LEGO NXT and sensors	2	Worksheet C.3.1
C.3.2	A first approach to construction of robots	3	
C.3.3	A first approach to programming robots	3	Worksheet C.3.3
C.3.4	"The cat, the mouse and the master" project	3	Worksheet C.3.4.1, Worksheet C.3.4.2, Appendix 4,
C.3.5	The 'data logger'	2	Worksheet C.3.5
C.4	**Theoretical framework**	**4**	
C.4.1	Constructivism and Constructionism	2	Worksheet C.4.1
C.4.2	Why robotics in education?	1	Worksheet

			C.4.2
C.4.3	Project-based learning	1	Worksheet C.4.3
C.5	**Methodology for developing robotic projects**	**14**	
C.5.1	Model for organizing robotics projects	1	Worksheet C.5.1, Appendix 2
C.5.2	An example of a robotics project: “BusRoute”	3	Worksheet C.5.2, section 3.2
C.5.3	Working on a new robotics-enhanced project	7	Worksheet C.5.3
C.5.4	Presentation and evaluation of the trainess’ projects	3	Worksheet C.5.4
C.6	**Evaluation of the course**	**2**	Worksheet C.6.1, Worksheet C.6.2

Worksheet C.3.1: Introduction to NXT brick and sensors

NXT brick and sensors-LEGO MINDSTRORMS NXT Edu Software

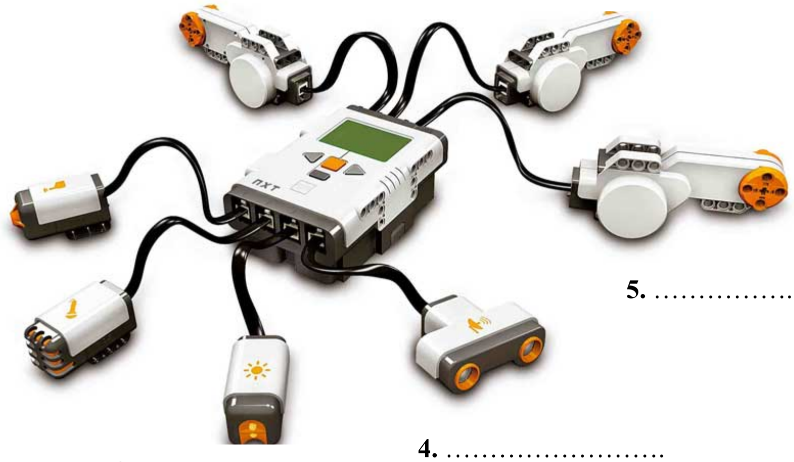
1. A **“robot”** is a structure that can:


Collect data from the environment INPUT
Process those data and make decisions PROCESSING
Execute those decisions having a behavior OUTPUT

According to the above, characterize the following items according to their function.

Item	**Function**
Light sensor	
NXT display	
Play of sound	
NXT brick	
Servomotor	
Touch sensor	

2. The NXT brick can communicate with the computer through a USB cable or a Bluetooth connection. On the NXT brick, sensors, motors, lights are connected with cables. Name the sensors, and motors in the following picture. Please notice to which port each of these objects is connected.



6. …………………….

1. ………

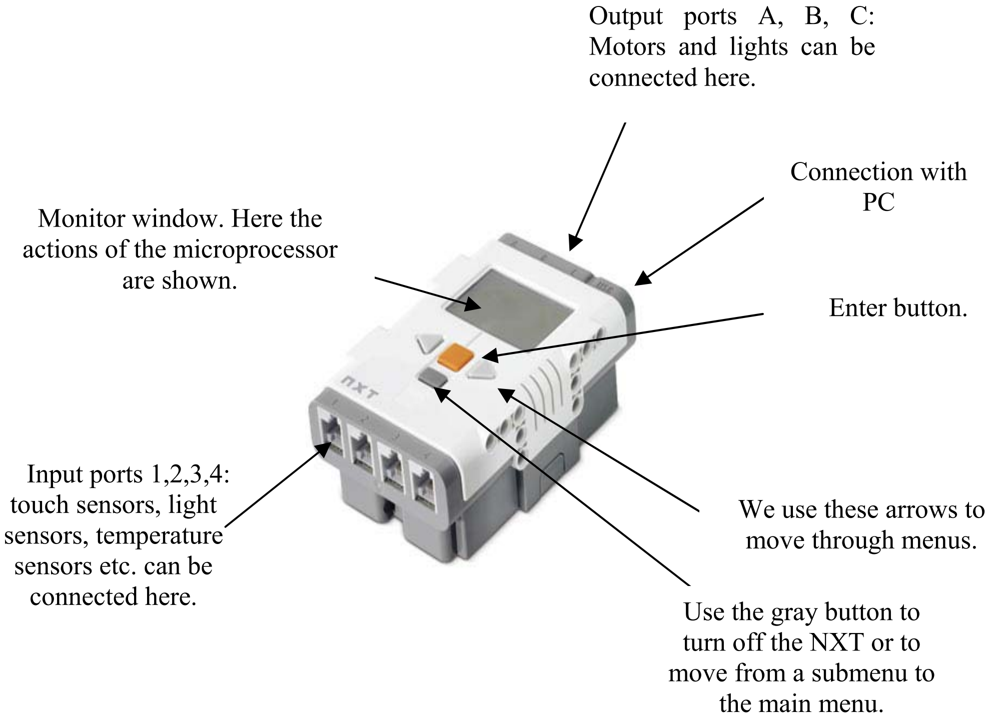
5. …………….

2. …………

4. …………………….

3. ……………

3. Parts of the NXT: On the NXT brick, sensors, motors, lights are connected with cables. Study the following picture.


Output ports A, B, C: Motors and lights can be connected here.
Connection with PC
Monitor window. Here the actions of the microprocessor are shown.
Enter button.
Input ports 1,2,3,4: touch sensors, light sensors, temperature sensors etc. can be connected here.
We use these arrows to move through menus.
Use the gray button to turn off the NXT or to move from a submenu to the main menu.
NXT

4. Menu *My Files*: Open the NXT brick and explore the NXT menu. Remember:

Orange button: On/Off, Enter

Light Grey arrows: navigation left and right

Bark grey: Clear, go back

Select *My Files* folder. Which subfolders can you find?

1……………….…..…… 2……………….…..…….. 3…..…………….……….

4. ………………………………

Open subfolder *Sound Files* and run the file *Startup*.

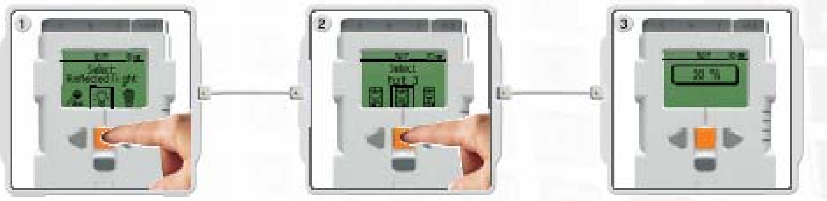
5. Menu View: The behaviour of a robot is usually based on the values of its sensors (e.g. light sensor). We can see the values of each sensor connected to our NXT by the View menu.

5.α Measurement of Reflected Light: The Light Sensor enables the robot to read the light intensity in a room, and to measure the light intensity on colored surfaces. You can test the Light Sensor in different ways using *View*.

Step 1: Connect the Light Sensor to the NXT (port 3). Select View in the NXT display.

Step 2: Select Reflected light

Step 3: Select the right port (port 3) and see the value of the light sensor on your NXT display.



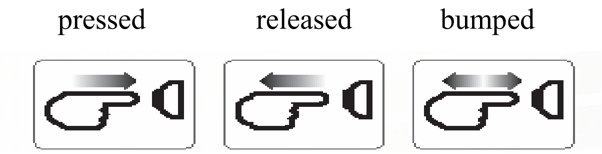
Hold the Light Sensor close to the different colours in your surrounding and see the different readings. Write your observation in the following table.

Colour	Value
White	
Green	
Blue	
Light grey	
Black	
In front of the window	
………………………	

Can the robot distinguish the color of a surface?

………………………………………………………………………………………

5.b Touch sensor: The Touch Sensor is a switch: it can be pressed or released.



See the current Touch Sensor value on the display using View menu.

Step 1: Connect the Touch Sensor to NXT (port 1). Select View in the NXT display.

Step 2: Select the Touch icon.

Step 3: Select the right port (port 1) and see the value on your NXT display. Press and release the button of the sensor.

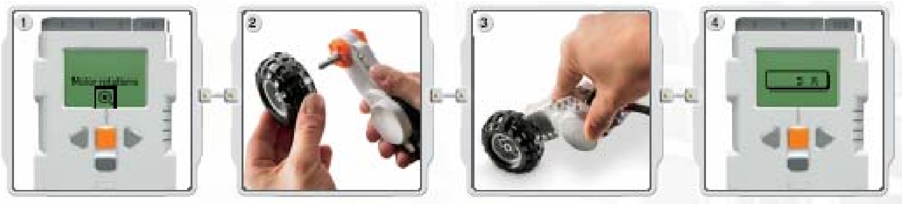
Condition	**Value**
Button Pressed	
Button released	

5.c Interactive Servo Motor: The Interactive Servo Motors have a built-in Rotation Sensor. The rotational feedback allows the NXT to control movements very precisely. The built-in Rotation Sensor measures the Motor rotations in degrees or full rotations. To test the rotation sensor connect the servomotor to NXT

Step 1: Select View in the NXT display.

Step 2: Select Motor degrees.

Step 3: Select the right port (port A or B or C). Now try to attach a wheel to the Motor and measure the degrees by pushing the wheel over the floor.



Can the robot distinguish the direction of rotation?

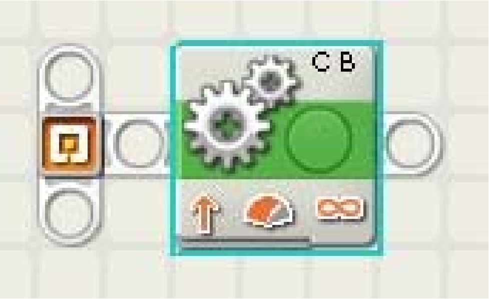
……………………………………………………………………………………

Test the option *Motor rotation*…

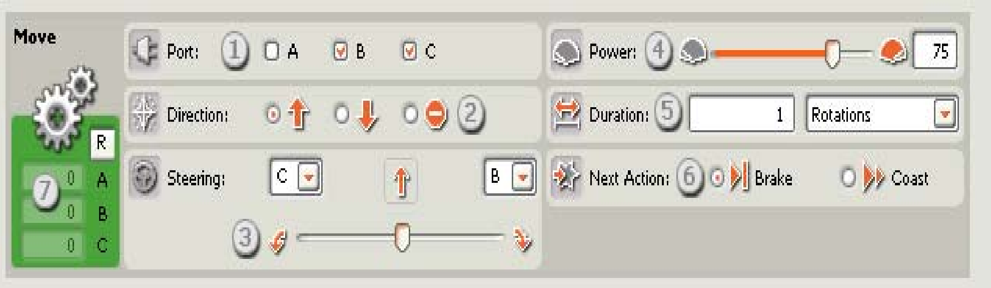
Worksheet C.3.3 A first approach to programming

Moving Around

1. Moving forward: For programming the behavior of a robot we may use the Lego Mindstorms Edu NXT software. This is a graphical programming environment. All available blocks (commands) can be found on the left hand side of the screen.



1.a. Open the Lego Mindstorms Edu NXT software and drag a Move block from the Common Palette on the workspace. Move Block can set our robot to go forwards in a straight line, backwards or to turn by following a curve. *Duration* (5), *power* (4), *direction* (2) of the motion can be defined by the properties of the block (configuration panel, lower part of the screen). Set the parameters of the configuration panel as below. (For more information on the properties of any block you can use the Help menu of the software).



Move

Port: 1 □ A ☑ B ☑ C

Power: 4 75

Direction: ○ ○ ○ 2

Duration: 5 1 Rotations

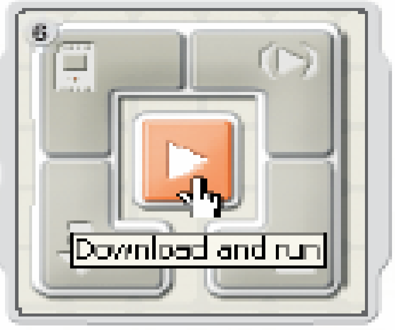
Steering: C B

3

Next Action: 6 ○ Brake ○ Coast

7 R
0 A
0 B
0 C

1.b. You have already made your first program. You can save it as *move_fd.rbt*. Can you guess what is going to happen if you run this program?



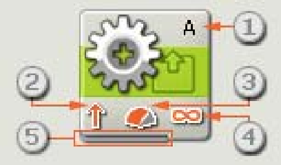
1.c In order to test this program connect your NXT to you computer and open it. Press the Download button of the *Controller* (on the right side of your screen). You can run the program from the software (press the *Rum button*) or from the *NXT* buttons (*Files/Software files /move_fd/run*). (Make sure that the motors are connected to the right ports)

2. Speed and Power: The power of motors defines the speed of your robot. Make measurements of the distance traveled by your robot in a specific time interval for different values of motor's power. Fill the following table with your measurments.

POWER	Time (sec)	Distance (cm)	Speed (cm/sec)

Use the grid to construct a power-speed graph. Can you figure out the exact formula for speed?

3. Turn Left: The Motor block allows for precise control of one motor's speed. You can find it at the *Complete Palette/Action/Motor*. (For more information on the properties of any block you can use the Help menu of the software).



3.a Write a program which changes the direction of your robot by 90 degrees to the right. Test your program. Save it as *right_turn.rbt.*

3.b. Sketch your program below making some notes on the appropriate properties of each block (duration, power etc.)

3.c. You can make a new command block which turns your robot to the right and save it for later use.

Step 1: Select the blocks of your *right_turn* program by your mouse.

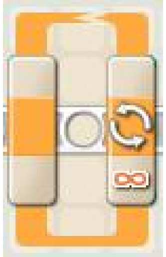
Step 2: With the blocks selected, choose the *Make a New My Block* command from the *Edit* menu at the top of the NXT software interface. This will open the first screen of the *My Block Builder* wizard.

Step 3: Give the name *Right_turn* to your new *My Block*. Then clik *Next*.

Step 4: Use your mouse to drag an icon representing turn into the editing box. Click *Finish* when you are done. Your new *Right_turn* block will appear in your current program and in the *Custom* palette, which is accessible by clicking the right-most tab at the bottom of the programming palette.

Now if you need to make your robot to turn right you can use this block.

3.d Use the *Move* or *Motor* blocks, the *Right_turn* block and a *Loop* block to program your robot to move on a square path. Save your program as *square.rbt*. Upload your file to eclass at the area of your group.



4. Make an investigation and propose a program that changes the direction of your robot by any angle. Upload your work at your group area of e-workspace.

5. Use a few minutes to complete your diary. Write your ideas, thoughts or comments and upload it at your group area.

Worksheet C.3.4.1: The Cat, the Mouse and the Master (I)

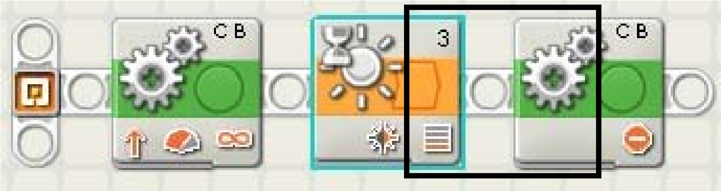
When a robot acquires senses and control…

In this activity, you will progressively construct a 'cat-robot' simulating the movement of a cat chasing mice. You will also need a mock up with small black areas around to put the robot move.

1.A Cat chasing a Mouse: *block, loop structure, light sensor*

1.a The **cat-robot** should make use of the C, B motors and should, also, include a light sensor, which provides input to port 3. Note that in order to connect the light sensor, follow the building instructions provided in the Lego Mindstroms Edu NXT software, area of the Robot Educator/Common palette/16.Detect Dark Line /Building Instructions / Light Module Down).

Open the Lego Mindstorms Edu NXT software. Develop the following program that simulates the movement of a 'cat-robot' that chases mice and name it as "cat-robot.rbt"! The program may have the following form and is based on the 'Wait Light' block.



1.b Make the cat-robot move: Let us suppose that your robot is a cat chasing a mouse (black spot on the floor)! Which are the individual steps that make up the cat's behavior? **State them** one after the other:

1. ………………………………………………………………………………

2. ………………………………………………………………………………

3. …..……………………………………………………………………………

4. ……………………………………………………………………………..

5. ……………………………………………………………………………..

State the cat's behaviours, as recorded during the preceding step, in correspondence with the commands included in the program and the necessary settings for each command.

Behaviour	Commands	Conditions / Settings

Run the program. In order to achieve the desired behavior, most probably you will need to make some changes to the control condition of the Wait Light block.

1.c Develop your first procedure or block **(My Block)**. Store the above program in a block (My Block) and name it 'mouse-stop'. By creating a 'MyBlock' command in your program, you may re-use the particular series of commands whenever you wish, provided that each time you call the block with that name!

Tip: Use the following commands: Menu selection: *Edit - Make A New My Block.*

1.d Use the block 'mouse-stop' and develop a program that makes the cat stop for a while, when encountering a dark area, and then, change direction and continue to chase mice, i.e. looking for black areas on the white mock-up. **Put** below a screenshot of your program (use Alt - PrintScreen to save the current screenshot)

1.e Loop construct: Modify the above program so that the cat should look for mice for 30 seconds!

Tip: In order to form the loop structure, select from the left hand side menu the Common palette, the Loop block (attention with the control condition that terminates the Loop Block!).

Put below the screenshot of your program (use Alt – PrintScreen to save the current screenshot)

Save the final version of the program as cat_mouse_groupxxx.rbt (where xxx represents the number of your team) at your group area of e-workspace.

1.f State in which ways you can terminate the loop structure:

2. The Cat encounters its Master: *Selection structure, Touch sensor, Display, Sound, Wait For*

What is going to happen if the cat's Master appears? At his/her touch, it will, probably, miaow, smile, stop for 3 sec and, then, will continue undeterred ☺ its chasing effort. In that case, the cat's behavior changes depending on the context. This behavior can be programmed using the *Switch* Block.

2.a You need a Touch Sensor in your robot (port 1).

In order to connect the sensor follow the building instructions provided in the Lego Mindstrorms Edu NXT software, area of the Robot Educator/Common palette/ 18.Detect Touch /Building Instructions / Touch Module Front.

2.b Develop and test a program that simulates the cat's behavior when chasing mice and it suddenly meets its master.

State the behavior that you want the cat-robot to have when its master touches it.

If [the cat gets a touch] ***then***

……………………………………………………………………………………..

Enter the above functions in your program "cat_mouse_groupxxx.rbt". Make use of the blocks *Switch, Display, Sound*.

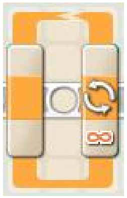
Tip: The *Switch* block is located at the left hand side menu *Common palette* (attention to the condition that controls the touch sensor!). Find there, also, the block *Display* (appearance of icon, text, or sensor values on the NXT display), *Sound* (sound production) and *Wait For* (stops the motion). Consult, also, the Brief Guide at the end of this worksheet.

Save the program as 'cat-mouse-man_groupxxx.rbt'.

Save the worksheet as 'Worksheet 3.4.1 _groupxxx.rbt'

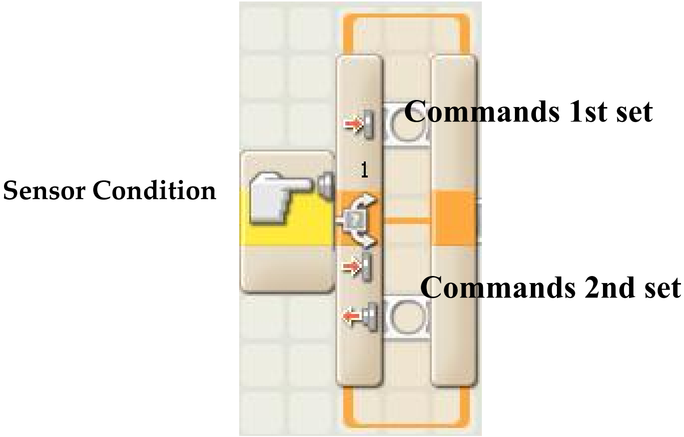
<u>**Brief Guide**</u>

'Display' Block: This block is used in order to display an icon, a text or to draft something on the NXT brick monitor.
'Sound' Block: This block is used in order to produce a sound. The sound files are saved separately in the NXT in a special folder.



'Loop' block: All of the blocks that are placed within a Loop are repeated for a specific number of times (count), for a specific time period (time), or until a certain condition is met (sensor) / forever (unlimited) according to the control condition which is selected.

'Switch' block: When the sensor condition is set 'true', the program performs the command set No. 1; otherwise, it performs the command set No. 2. This command may take more than one conditions.



‘Wait for’ block: delays the performance of the next block for the time defined in the appropriate parameter. In this case: wait for 1 second.

Worksheet C.3.4.2 The Cat, the Mouse and the Master (II)

Selection construct, Variables and Data Input

In this activity, you will progressively make the ‘cat-robot’ move in a spiral route.

1. Open the Lego Mindstorms Edu NXT software. Search for the file spiral_display.rbt. You may ask your trainer for help. Study this program and try to describe the kind of behavior that you expect your robot to execute once it has run that program.

Comment on the actions performed by each of the blocks of the program “spiral_display.rbt”. You can add comments above each block by using the tool *Comment tool* of the tool bar.

2. Run the program: What kind of actions does your robot perform?

……………………..………………………………………………………………

After the observation of the cat’s behavior add or modify the comments on the actions performed by each block of the program “spiral_display.rbt”. This time use CAPITAL (upper case) letters.

……………………..………………………………………………………………

Save your program as “cat_spiral_move_groupxxx.rbt” at your group area of e-workspace.

3. What is the name of the variable that controls the distance run by your robot before turning?

……………………..………………………………………………………………

Which is its initial value?

…………………………………………………………………………

Which is its final value, as quoted in the NXT screen?

……………….…………………………..

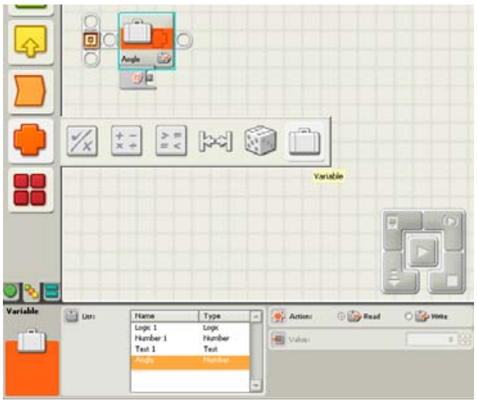
Save your worksheet as “Worksheet4-3-2-groupxxx.rbt” at your group area of e-workspace .

Brief Guide

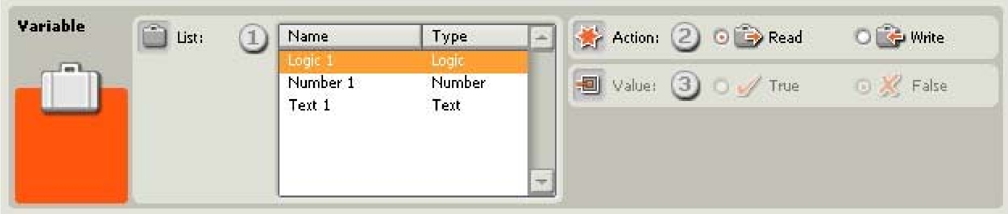
Variable: The definition and use of variables in the Lego Mindstroms Edu NXT environment is carried out in 3 steps.

Step 1: Variable definition: Select from the *Edit* menu / the *Define variables* option. Define the name and the type of the variable (numeral, logic, text).

Step 2: Variable initialisation: Select from the Complete palette the command “Variable”. You can see the settings of this block at the following picture:



Parameters of the variable block



Variable

List: ①

Name	Type
Logic 1	Logic
Number 1	Number
Text 1	Text

Action: ② ● Read ○ Write

Value: ③ ○ True ○ False

1. Select from the list the name of the variable you want to use in your program (variable definition is described in Step 1).

2. Select the action that the Variable block will perform. By selecting the *Write* command, the variable acquires a value. With the *Read* option, the value of the variable becomes available in the program.

3. Area wherein the value of the variable is defined.

Step 3. We recall the value of the command and we can use it as input in other commands (option *Read*).

<u>**Math Command:**</u> It allows the execution of arithmetic operations. It can take input values from other variables.

<u>**Number to Text Command:**</u> It accepts as input a numerical value which it converts into a text and can display it, on the NXT display.

Worksheet C.3.5 The "data logger"

Linear motion

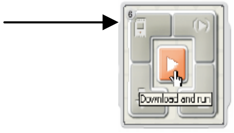
1. The NXT through the *File Access* Blocks can collect, save and retrieve data from timers or sensors.

To collect and save the data you need three *Access blocks*. These three blocks could be positioned right next to each other or could be spaced throughout your program.

The first *File Access* block (with "*Delete*" selected in its configuration panel) would clear content of the file.

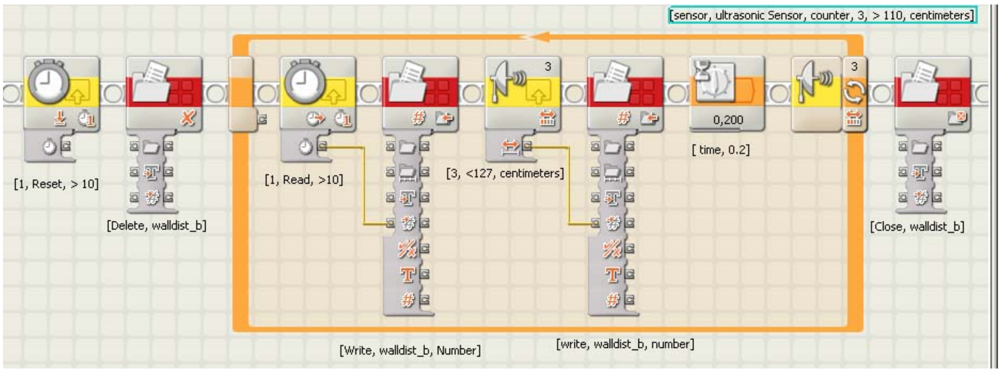
The second (with "*Write*" selected in its configuration panel) would write data to a file.

The third *File Access* block (with "*Close*" selected in its configuration panel) positioned somewhere later in the program would close the file.



To retrieve the data you should access NXT memory through NXT Window button (on the Controller: NXT window/Memory/Others) and Upload the *.txt file to your computer).

2. A simple data logging program that collects data from a timer and the ultrasonic sensor is shown at the following picture (comments in brackets [ ] are the parameters of each block). Try to make this program on your computer.



[sensor, ultrasonic Sensor, counter, 3, > 110, centimeters]

[1, Reset, > 10]

[Delete, walldist_b]

[1, Read, >10]

[Write, walldist_b, Number]

[3, <127, centimeters]

[write, walldist_b, number]

[time, 0.2]

[Close, walldist_b]

2.a Please identify: The timer block, the ultrasonic block, the three File Access block and their functions.

2.b. What is the name of the data file created by this program?

2.c Which is the condition that terminates the loop?

2.d What kind of data is going to be collected if you run this program?

3.a Create a parallel sequence beam by moving the mouse pointer over the starting point and pressing and holding your mouse button while you move it. Double click to finch this action. And blocks in order to make your robot moving forward with steady speed. Save your program as walldistance.rbt at your group area.

3.b Add an ultrasonic sensor (find instructions in Lego Mindstrorms Edu NXT Robot Educator/Common palette/ 14.Detect distance/Building Instructions / Ultrasonic Module). You are ready to test your program. Make all necessary arraignments in your workspace and test it.

3.c Download and run it. Retrieve the data file.

4. Propose at least two activities, suitable for students who have already been introduced to linear motion with constant velocity, which will make use of this set of data.

1st activity …………………………………………………………………………….

……………………………………………………………………………………..

……………………………………………………………………………………..

2nd activity …………………………………………………………………………..

……………………………………………………………………………………….

………………………………………………………………………………………

5. Suggest a program which can collect data for uniform accelerated motion. Upload your program.

6. Find more information about the functions of *timer*, *ultrasonic*, *file access b*lock at the *Help* menu of the software.

7. Take a few minutes to complete your diary. Write your ideas, thoughts or comments and upload it at your group area.

Worksheet C.4.1 Constructivism and Constructionism

1. Read the main points of the following paper with the members of your group (20 min).

Ackermann E., (2001) Piaget's constructivism, Papert's constructionism: What's the difference? Future of Learning Group Publication

2. Discuss and write down the main similarities and differences between constructivism and constructionism (20 min).

3. Choose one representative from your group to present to the whole class your work (5 min)

Worksheet C.4.2 Why robotics in education?

1. Read a part from the article (20 min):

Resnick M (2002). Rethinking Learning in the Digital Age. In The Global Information Technology Report: Readiness for the Networked World, edited by G. Kirkman. Oxford University Press.

2. Discuss in your group the ideas presented by the author. Point out three reasons you consider as important for the introduction of robotics in school education. (20 min)

3. Upload in the forum area of the e-workspace a summary of what you have discussed (5 min).

4. Choose one representative from your group to present to the whole class your work (5 min).

Worksheet C.4.3 Project based learning

1. Read the main points of the article (20 min):

Mike Carbonaro, Marion Rex, Joan Chambers (2004) Using LEGO Robotics in a Project-Based Learning Environment, *The Interactive Multimedia Electronic Journal of Computer-Enhanced Learning* 6 (1) 2004

2. Discuss in your group and point out three main advantages of the project-based learning against the traditional teacher-centered teaching model. (20 min)

3. Upload in the forum area of the e-workspace a summary of what you have discussed (5 min).

4. Choose one representative from your group to present to the whole class your work (5 min).

Worksheet C.5.1 Model for organizing a robotic project

Activity 1 (20 minutes, work in groups of 4 trainees)

In previous sessions of this course we have discussed thoroughly constructivism learning approach and its implications in teaching. Concerning *Robotics in Education* we have illustrated interesting features of learning by constructing artifacts and we have discussed the constructionist approach in teaching and learning.

1. Make a list of seven features that a robotic project should have in order to serve constructivist and constructionism perspectives of teaching and learning.

…………………………………………………………………………………….
…………………………………………………………………………………….

2. Be prepared to present your list to the whole class.

Worksheet C.5.2 An example of a project: "The BusRoute"

Activity 1 (35 minutes)

"The BusRoute" is a robotic project designed for students of secondary education. Its duration is 12-14 teaching periods (45 minutes). The project follows the project model presented in previous sessions and it is developed in five stages: Engagement stage, Exploration stage, Investigation stage, Creation stage, Evaluation stage.

Stage	**hours**	**Teaching Theme**	**Worksheets**
Engagement stage	2	Public Transport A robot bus	Worksheet 1 Worksheet 2
Exploration stage	4-5	Getting to know the structural materials Construction of a robot car Programming a robot Use of light sensor	Worksheet 3, Worksheet 4, Worksheet 5, Worksheet 6
Investigation stage	2-3	Construction of the bus Suggest a solution	Worksheet 7 Worksheet 8
Creation stage	1-2	Synthesize and Create	Worksheet 9
Evaluation stage	1-2	Presentations & Discussion	

We suggest you to select one of the above stages and study with your colleagues the description of this stage and the relevant worksheets. The materials are available through e-workspace (each group must select a different stage to study)

Activity 2 (10 minutes)

After the end of the Activity 1 discuss in your group the following issues:

1. What kind of difficulties may a teacher face during the implementation of this stage?

2. What kind of difficulties may the students face during the implementation of this stage?

3. Complete the following table with activities that may be included in the stage you have studied.

Stage	Teaching Strategies-Tools	Student activities	Teacher activities

4. Be prepared to share your thoughts with the rest of your class.

Worksheet C.5.3 Working on a new robotics-enhanced project

Developing a robotics-enhanced project: specifications

We propose that your work includes either an activity or a project. Both cases should include actions involving students, promoting experimentation, exploration, open problems and self-evaluation.

Follow the following pattern for the description of your work by answering each question included in each stage.

Expected Results: What learning outcomes are expected (outcomes may be classified in knowledge, skills, attitudes and values)?

Describe the stages of the project and develop the appropriate materials by answering the questions of each stage.

Engagement Stage: What is the real problem set by this project? How are you going to involve your pupils in its formation?

Exploration Stage: Which are the materials that students will use in their work? Which of the basic software functions will they utilize? How are the pupils going to be organized?

Investigation Stage: Which of the open problems you would like your pupils to investigate? How are the pupils going to be organized in relation to the open questions that they will investigate?

Creation Stage: How should the diary of the pupils be structured in order to reflect their actions and thoughts?

Evaluation Stage: What kind of evaluation are you going to use: self evaluation, peer evaluation teacher evaluation? Select criteria and formulate rubrics.

Stage	**Description Objectives**	**Resources**	**Expected Products**	**Actions of students**
Engagement				
Exploration				
Investigation				
Creation				
Evaluation				

Brief Description of the project: Upload in the e-workspace a brief description of your project and other materials that may support your proposal.

Worksheet C.5.4 Presentation and evaluation of the projects

Rubrics for assessing a robotics project

	Novice	Apprentice	Practitioner	Expert
Authenticity	Content and skills are connected to later use in school only.	Content or skills are somewhat connected to life outside of school.	Content and skills are clearly connected to life outside of school, such as the work world.	Content and skills are highly relevant by connecting to students' lives right now.
Open-Ended	Task has only one correct response.	Task allows limited room for different approaches.	Task allows for different approaches based on the same content/skills base.	Task allows students to choose different assessment measures for the task
Complexity	Task contains different skills, most lower order.	Task contains many different skills and content.	Task contains many different skills and content, including higher level thinking.	Task contains many different skills and content, including higher-level thinking. Task contains opportunities for students to choose some of the skills and content.
Curricular Connection	Task is loosely connected to key skills and content in curriculum.	Task is clearly connected to key skills and content in curriculum.	Task is connected to key skills and content in curriculum. time frame and scope of task match time frame and scope in the curriculum.	All tasks are clearly connected to national curricular standards.

Appendix 1 . “Didactic Contract”

During the module 2 (Agreeing on a "didactic contract") trainers and trainees discuss the overall aim, the specific objectives of the course, the training methodology and the expected training results. This discussion is expected to result in a “didactic contract” that will offer a better adjustment of the course to the trainees’ needs and interests. What follows is the “didactic contract” agreed with the trainees in the pilot TERECoP training course held in ASPETE, Athens (April-May 2008).

The overall aim of the course is to provide opportunities for teachers to examine how robotic technologies can be used to promote a constructivist-constructionist approach to learning under a co-operative and collaborative frame of work. The implementation of robotics-enhanced constructivist teaching and learning practices demands that teachers play a new role. This means that opportunities, like exposure to a number of critical examples and experience in designing computer-based robotic activities and integrating them in their classroom practice in constructivist ways, are of great priority. The goal is teachers to be convinced by their own personal experience for the potentiality of robotic technology as learning tool.

In this course we regard that technology alone cannot affect minds. The curriculum design will follow an innovative constructivist perspective with emphasis on aligning computer and robotic technology with subject matter and learners’ needs for the purpose of constructing meaning in social learning environments. In such learning environments the focus is not on the individual but on interactive systems that include individuals interacting with each other, instructional materials, subject matter, and tools. Computer-based robotics is an innovative technology that can create a rich interactive environment encouraging constructivist learning.

The specific objectives are

- to familiarise trainees with appropriate robotics-based learning environments including Lego Mindstorms NXT system and a set of critical examples and activities that can support constructivist teaching and learning in science and technology subjects
- to enable trainees to use robotics technology in a way that can contribute to the realisation of
    - meaningful learning based on students’ own team work with teaching materials
    - authentic learning using learning resources of real-life, occupational situations, or simulations of the every day phenomena,
    - social learning though the use of e-learning classes
    - active-reflective learning working on experiments or problem-solving and using available resources selectively according to their own interests,

search and learning strategies

- project-based learning seeking solutions to real world problems, which are based on a technology-based framework

- to create a community of practice between trainers and teachers for facilitating and sustaining teachers' professional development in using robotic tools to support their students' learning by active exploration and social construction of new knowledge.

Pedagogical and didactical approach: Constructivist-constructionist pedagogy and a learner-centered didactical approach will be applied taking into consideration learner's characteristics for an effective technology-enhanced learning design. A collaborative e-learning environment will support the course based on the belief that the inherent dynamics of a necessary mutual process are likely to be more conducive to meaningful transformation, carrying so a sense of greater potential for development.

The expected impact on trainees is to be trained in a way that robotic technology-based learning will play an important aspect of their future work as teachers or professional educators. Trainees are expected to be able to

- develop innovative collaborative strategies in their classes supported by the development of e-learning communities
- select exploratory learning activities that can support social constructivist teaching and learning.
- use the proposed tools in real classrooms situation.
- design, build and program their own robotic models.

Appendix 2. Project –Based learning: Important features

In previous sessions of this course the constructivism approach and its implications in teaching has been thoroughly presented. Concerning Robotics in Education interesting features of learning by constructing artefacts have been illustrated. Finally constructionist approach in teaching and learning has been discussed. At that point we suggested that the appropriate way to implementing robotics in a learning process is through projects. A robotic project with the following features may serve constructivism and constuctionism approach to teaching and learning.

A robotics-enhanced project may focuses on creation of a product (artifact) that reflects learners' abilities and learners' understanding. Therefore project activities should be organized around a question (driven question) or a theme which can guide learners progressively through the learning process. The driven question or theme of the project should be open ended in order to serve different learning goals and different learning styles. It is also very important for a successful project, that the theme of the project is significant and meaningful for the learners. For example, projects which deal with real life problems provoke students' interest and motivation.

In a project learners are actively involved in the formation of the driven question and in the description of the final product. Clarification of the goals of the project and the criteria for assessment of the final product (rubrics) are collaboratively agreed by student and teachers in the beginning of the project. Learners organize their work by themselves and they work autonomously over extended periods of time.

Teacher is a facilitator /mediator of the learning process. S/he creates a learning environment and provides support for the learners. S/he allows them to take as much responsibility for their own learning as they can. Teacher keeps the balance between guiding his/hers students through learning activities and challenging them with interesting questions for further investigation.

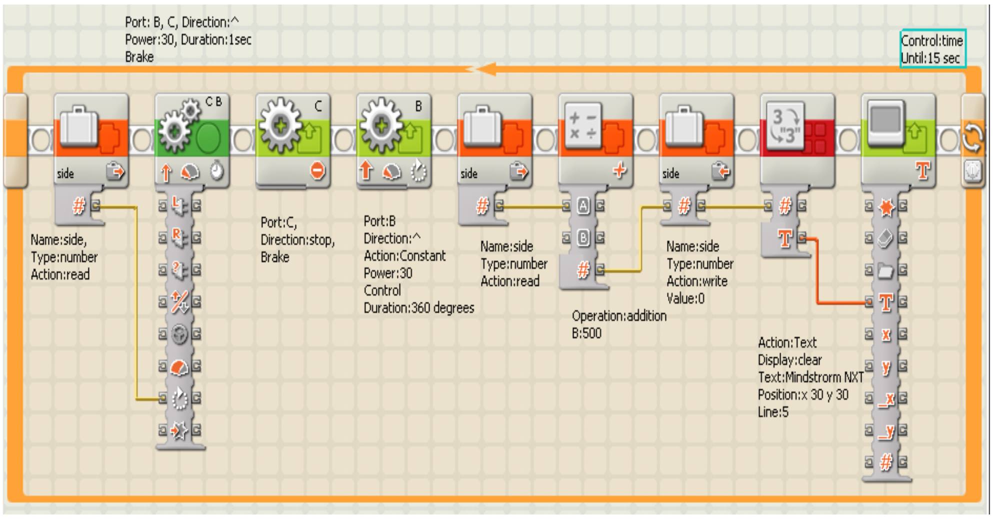
Communication in group level and in the classroom is an important element of a project. Learners express their ideas and test their understanding through their collaboration in small group or in the classroom. Feedback from each other and the teacher give learners the opportunity to improve their work and meet the learning goals of the project.

Appendix 3. Methodology for organizing robotics-enhanced projects

Engagement stage:	Students are provided with an open-ended problem and get involved in defining the project. This stage requires the identification and representation of a scientific problem. Students work as a class putting their ideas into a question format. As they are doing so, they are identifying and representing a problem and different issues involved (e.g. brainstorming at class level).
Exploration stage:	Students get familiar with LegoLogo, controlling devices and software, make hypothesis and test their validity in real conditions, provide initial ideas. Students are divided in groups in order to answer to simple questions and study specific cases in order to get familiar with the controlling devices and software (e.g. work in groups with worksheets – structured activity).
Investigation stage:	Students search for resources and investigate alternative solutions. Students reconsider the problem and the different issues rose during the engagement stage based on their experience gained through the exploration stage. At this stage students in collaboration with the teacher formulate the driving questions/problems which link with the learning goals of the project. The student groups undertake to solve the particular problems, investigate alternative solutions and argument on their final proposals concerning the artifact and the software the developed (e.g. work in groups with worksheets, keep diary – open activity).
Creation stage:	Students share and combine their artefacts, synthesize 'solutions' to the project reflect on their initial ideas. Students present their work in class and then each group work on the synthesis of a final 'product' including the artifact and the software (e.g. work in groups with worksheets, keep diary – result in a product). This work may lead to similar solutions but also to innovative proposals.
Evaluation stage:	Students share their ideas, products at class level, argument on their final proposals and evaluate them. Alternative solutions are presented at class level and evaluated based on the driving questions/criteria posed in previous stages of the project (stages of engagement, investigation). At this stage students should critically judge their work, express their opinions, compare their works, and reach a common proposal to the project (e.g. make presentations, discuss, peer evaluation). Students should also reflect on and evaluate their collaboration.

Appendix 4

Description of the spiral_display.rbt (to be used with worksheet 3.4.2)



Port: B, C, Direction:^
Power:30, Duration:1sec
Brake

Control:time
Until:15 sec

Name:side,
Type:number
Action:read

Port:C,
Direction:stop,
Brake

Port:B
Direction:^
Action:Constant
Power:30
Control
Duration:360 degrees

Name:side
Type:number
Action:read

Operation:addition
B:500

Name:side
Type:number
Action:write
Value:0

Action:Text
Display:clear
Text:Mindstrorm NXT
Position:x 30 y 30
Line:5

4.4 Monitoring and Evaluation of the Training Course

Authors: Dimitris Alimisis, Stassini Frangou, Kyparisia Papanikolaou

4.4.1 Introduction

For the evaluation of the training course the following methods are proposed for use during the training activities and at the end of the course:

- Monitoring and personal notes by the trainer: recording the trainees' work and commenting thereon
- Keeping a written personal or team diary by trainees themselves upon the end of each meeting: the trainer studies the diaries on each occasion and provides feedback at the next meeting
- Video-recording of trainees' work (at least of the most significant phases)
- Evaluation of trainees' worksheets
- Evaluation of the e-class forum
- Evaluation of trainees' products
- Team interview of trainees at the end of the course.
- Filling a written questionnaire by trainees at the end of the course.

Some evaluation tools appropriate for the implementation of the above mentioned methods are presented in the next lines.

4.4.1 Trainees' (personal or team) Diary

Date:

Name(s)………………………………………………………………………………..

What was the best thing that has happened to you today during the lesson? ………………

What was the worst thing that has happened to you today during the lesson? …………….

Comments ………………………………………………………………………………..

4.4.2 Trainer Diary

Date ………………………….

Trainer **……………………………………………………..**

The work of team …………….. today…

1. robot construction
   - has not been attempted
   - has been unsuccessfully attempted (description)
   - has been attempted with partial success (description….)
   - has been completed without assistance from the teacher (description….)
   - has been successfully completed with assistance from the teacher (what kind of assistance?….)
2. in constructing the robot the team used …
   - insufficient pieces
   - just the basics
   - variety of materials
   - variety of materials in creative fashions
   - great variety of materials with high creativity which added functionality to the robot
3. experimented and tried varied work designs
   - many
   - quite a few
   - some
   - minimal
   - none
4. constructed a program on the PC aimed at controlling the robot's behavior
   - not attempted
   - unsuccessfully attempted (explain)
   - attempted with partial success (explain)
   - completed without assistance from the trainer

- successfully attempted with assistance from the trainer (what kind of assistance?)

5. the work of team … today

- exceeded to a great extent my expectations
- exceeded my expectations
- met my expectations
- did not meet my expectations
- did not meet at all my expectations

4.4.3 Questionnaire (for trainees)

This questionnaire has been designed in such a way as to give us the necessary feedback on your part and help us to improve the TERECoP training course in which you have participated.

Name:
………………………………………………………………………………………..

A. Evaluation of Training Method

1**.** How do you evaluate the participation/involvement of the trainees in the training course activities?

- Very active
- Quite active
- Moderately active
- Barely active

Please explain….

2. What do you think of the balance that existed between the practical activities (on the part of the trainees) and the presentations (on the part of the trainers)?

- There has been proper balance
- I would like more practical activities
- I would like more presentations

Please explain…

3. What do you think of the support provided by the trainers?

- Very satisfactory

- Quite satisfactory
- Moderately satisfactory
- Unsatisfactory

Please explain…

4. What do you think of the duration of the course?

- Satisfactory
- I would like longer duration
- I would like shorter duration

Please explain…

5. Mention any difficulties that you came across during the course…

B. **Evaluation of Training Material**s

What do you think of the training materials (worksheets, examples by means of software, presentations etc.)

- Very useful
- Quite useful
- Moderately useful
- Barely useful

Please explain…

C. Evaluation of the e-class

1. What actions do you think the e-class has supported within the framework of the course? ……………………………………………………………………………..

2. State a positive experience from your communication via the e-class ………….:

3. State a negative experience from your communication via the e-class ………….

4. Comment on the e-class tools in terms of their usefulness in the activities that they incited or supported and the services that they offered. …

e-class tools	**comments**
Agenda	
Area of Plenary Sessions	
Areas of Team Discus-	

Documents	
Announcements	
Links	

D. Learning Experience and Incorporation in School Reality

1**.** Have you found in this course anything interesting that you could implement as a teacher in school class with your students?

2. in your opinion, in which of your school courses that you teach (or that you will teach in the future) you feel that the robotics training activities, can be utilized?

3. How do you assess the learning value that robotics training activities can add to teaching school lessons of your specialty?

- Very important
- Quite important
- Somewhat important
- Not important at all

Please explain…

E. Evaluate each one of the course's constituent parts, as mentioned below, with a mark from the following scale:

6 = excellent, **5** = very good, **4** = satisfactory, **3** = moderately good, **2** = deficient **1** = very deficient

Parts of the Course	**6**	**5**	**4**	**3**	**2**	**1**
a. Training content						
b. Training method						
c. Support provided by the trainers						
d. Training materials						
e. Training results						
f. Electronic class						

F. What would you propose for the improvement of this training course? ……..

Please add any other comments that you think they could be useful in evaluating the course ………………………………………………………………………

4. 4. 4 Semi-structured interview

Trainees discuss the following questions in teams for 30 minutes and, then, share their ideas with trainer in a plenary session.

1. Which of the tools/ techniques/ actions used in this course you think that they have followed the constructive teaching and learning approach? What would you propose towards that direction? …………………………………………………….

2. Which of the experiences /knowledge/ skills that you have acquired from that seminar you think you can implement in your class or anywhere else? ……………

3. How can robotics (hardware and software) be incorporated in your school class? ....

4. What benefits do you expect for your students? ...……………………………..

5. What difficulties do you expect that you might encounter? …………………….

6. What kind of support you would need? ……………………………………….

4.5 Experiences and recommendations from the training course implemented in Athens

Authors: Dimitris Alimisis, Stassini Frangou, Kyparisia Papanikolaou

4.5.1 Introduction

The training course implemented in Greece was held at the premises of the School of Pedagogical and Technological Education (ASPETE) in Athens, and was organized in 5 face-to-face meetings that each one lasted for 6 teaching periods of 45 minutes, during 3 Fridays/Saturdays afternoons.

20 of the 23 trainees participated in the final evaluation at the end of the training course. As to their specialization, there were 2 Mathematicians, 1 Physicist, 5 Engineers, 8 Informatics teachers and 4 Primary School teachers (10 males, 10 females). The trainees were volunteers in the course (some of them work as teacher trainers) and only 2 of them had a previous experience in educational robotics.

For the evaluation of the course, a series of tools were used. This section presents and discusses some of the evaluations made by the trainees themselves regarding the educational methodology applied in the course, as recorded in the diaries that they kept on the e-class right after the end of each meeting and their responses to the written questionnaire which was given to them at the end of the course.

4.5.2 The diaries

A selection of typical statements from the diaries of the groups of trainees concerning the course training methodology is shown in table 4.5.1.

Table 4.5.1. *Typical Statements from the Diaries of the Trainees written at the end of the meetings*

Group	*"What was the best thing that happened to you today?"*	*"What was the worst thing that happened to you today?"*
A	We experienced the joy of creation, we built it up and it proved operational. We managed to complete an exercise and to park the bus.	Very little practical application, quite a lot of writing on the *Word*.
B	The whole process of the robot construction and its ensuing programming was very pleasant and creative The construction process and that	We did not manage to make our robot move along a square … We ran out of time for experimentation with all the activities proposed in the work sheets. There should

	of experimentation with the scenario of coil.	have been more time available for experimentation and testing.
C	The creation of the robot and the implementation of the initial programs. When, ultimately, the robot moved along a square on the floor we rejoiced like young children! The activity where we tried to discover the function of the variables within a program … The fact that our group managed to propose a number of ideas for implementation in class.	The activity at the project's assembly stage "The bus running". The assembly stage was impossible to be implemented and we were compelled to talk theoretically.
D	The success with the cat robot which proved able to catch the mouse and turn tail at full speed… The exploration of the robot. It was great fun!... The collective effort (one section by each group) in constructing the bus. It was a very interesting and very well organized approach.	
E	The discussion of the construction ideas and the experiment with the power-speed. The result of the commands given to the cat robot to miaow The discussion about construction ideas.	Little time available for practical work … The hurried process regarding proposals for teaching strategies. The activity did not convince us as to its targets and what is looking for.

F	When we collectively constructed the NXT, set it in operation and carried out the activities Out contact with the NXT has begun to attract our attention and we are already anticipating the learning scenarios to be included in classes The use of robot construction based on projects (collective work – development stages) When we readily completed the program whereby the bus-robot stopped when it encountered any obstacle	When the lesson was over! Difficulty in the exploration and understanding of the variables. If there is not adequate time available for the completion of the activities, learning is senseless. The seminar should have lasted longer.

4.5.3 The questionnaire

The responses of the trainees to a selection of questions included in the questionnaire are presented in the following lines.

A. *What do you think of the participation / involvement of the trainees in the program's activities?*

The trainees, overall, describe their participation / involvement in the program as very or quite active. Their responses focus on their active involvement in the activities concerning robotic constructions and their programming, on team works, as well as on the debates that took place during the program in the classroom and via the e-class. The appraisal of their participation in the "theoretical" part of the program was, likewise, positive. ("*We took an active part even in the theoretical part, where our active involvement was encouraged*").

B. *What do you think of the proportion that existed between practical activities (by the trainees) and the presentations (by the trainers)?*

75 % think that there had been proper balance between practical activities (by the trainees) and presentations (by the trainers). They think that the presentations were necessary for the support of the practical activities ("*The presentations had appropriate duration at critical points in the progress of works and their completion*"; "*I think that there was a kind of concordance, where the trainers showed us certain basic things and we developed same through our activities*".).

25% ask for more practical activities: "*A greater involvement in activities and their exploration would have generated great interest*"; "*In general terms, there* *had been balance, but I would have liked more weight to have been given to the practical activities, since they are more attractive. Besides, we are "full" of theoretical presentations*".

C. *What do you think of the support provided by the trainers?*

The trainees describe the support provided by the trainers as very (95%) or quite (5%) satisfactory: "*They supported ideas, encouraged efforts, proposed solutions*". They describe their support as "discrete and when there was need for it". They describe, likewise, the support provided via the e-class as satisfactory.

D. *What do you think of the duration of the program?*

30% consider the program's duration as satisfactory, while 70% would have liked more hours spent on it. Some of them explain that it was their first contact with the programmable robotic constructions and needed more time to become more familiar with them ("*In quite a few of cases I felt that time was against me and I think that I needed more time available*"), others would have liked more time as they wanted to do more things ("*to swim in deeper water*", "*there were things we did not have time to do, although they were interesting, e.g. collection of data from the environment*"). Others think that they needed more time for the preparation of their assembly works and that through longer engagement in the subject matter in the classroom, they could have produced better projects.

E. *State any difficulties that you encountered during the program*

They mention difficulties in respect of programming activities ("*certain Switches seemed too heavy for me*", "*I would have liked more engagement and experimentation activity in the programming field for the development of the necessary skills*"), difficulties in respect of cooperation between teams outside seminar when dealing with the assembly side of their group project, difficulties with the material available: they would have liked more Lego blocks available and one set per each trainee.

F. *What do you think of the training materials (work sheets, software examples, presentations et al.)*

75% consider as very useful and 25% as quite useful the training material given to them. They explain that the material "*gave ideas and outlets regarding the pedagogical approach, as well as regarding the educational utilization…*"; "*with effective subject sequence…*"; "*it succeeded in involving us under normal conditions in the rationale and philosophy of both, Lego Mindstorms and the logic of robotics in education*". They considered as positive the fact that "*there had been presentation of a comprehensive work, which enabled them to study all its stages*". Some of them would have liked more material "*for additional stimuli…*" or for "*homework*".

G. *What in the program seemed interesting to you and worth-while using in class, as a teacher, with your pupils?*

They mention the teaching means used, such as the e-class and the work sheets, the training method and the teaching approach based on the cooperation of teams. Especially appreciated was the cooperation between teams in processing the theoretical texts, which took place at the first meeting; it was found useful for their own work in their capacity as trainers of other teachers.

H. *Evaluate each one of the parts of the program, stated in the table shown below, by marking them as follows:* ***6****= excellent,* ***5****= very good,* ***4****=satisfactory,* ***3****= moderate,* ***2****= inefficient* ***1****= very inefficient*

The average marking was as follows:

Table 4.5.2. *Evaluation marks for each one of the parts of the course*

	mark
Educational content	5.60
Educational method	5.45
Support by the trainers	5.80
Educational material	5.40
Educational results	5.25
Electronic class	5.00

4.5.3 The trainees' products

During the course, the trainees had to design their own projects based on the proposed methodology. Six of the seven groups of trainees developed and submitted interesting projects. All the groups worked with the Lego Mindstorms kit and programmed the robotic construction with the Lego Mindstorms Education NXT version 1.0. Below we provide brief presentations of the six projects.

Project 1: selector of recycled garbage

This group consisted of two mathematicians and two computer scientists. According to their project, school students are invited to construct a simulation of a selector of recycled garbage able to identify the colour of different objects - normally garbage bags come in special colours (see Fig. 4.5.1).

The selector decides if the object is to be recycled or not based on its colour, and accordingly puts the object in the appropriate bin. The robot is equipped with two belts and a light (or colour) sensor.

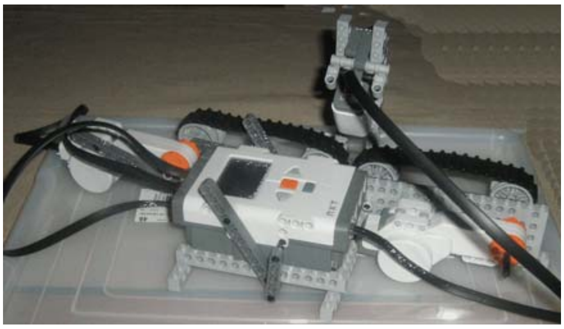


Fig. 4.5.1. *Selector of recycled garbage*

The sensor checks the colour of the objects and activates one of the two belts accordingly. Worksheets for school students were also produced by the trainees following the proposed project-based learning methodology. Students are supposed to work in groups in a laboratory equipped with computers and some Lego Mindstorms kits.

Project 2: autonomous irrigation system for water management

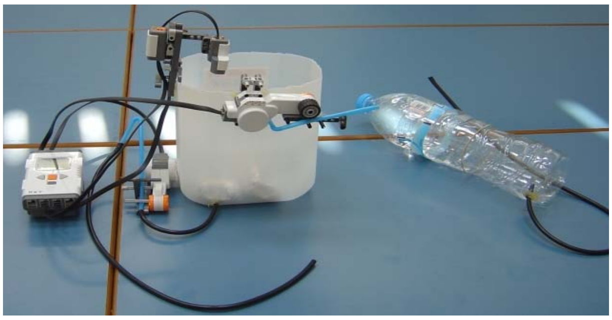


Fig. 4.5.2 *Autonomous irrigation system for water management*

This group consisted of a mechanical engineer and a computer scientist. Through this project school students are invited to design and construct an autonomous irrigation system for water management. The basic functions of this system are: (a) filling up a tank and control of the water level, (b) control of watering from the tank during the night.

The main challenges set by this project concern (a) avoiding water loss while filling up a tank, i.e. the tank must not be overflowed and (b) automatic provision of water from the tank when it is getting dark and the climate conditions favour watering. The characteristics of the system can be changed or enriched by students' ideas. Lego Mindstorms NXT kit, a plastic tank and watering pipes are used for the construction of the system (see Fig.4.5.2). Its behaviour is being arranged through sensors controlling the behaviour of motors. The project is organized in 5 stages following the proposed project–based learning methodology and aims, in addition to other objectives, to sensitise students about the ecological management of water resources.

Project 3: Organizing seats in a theatre

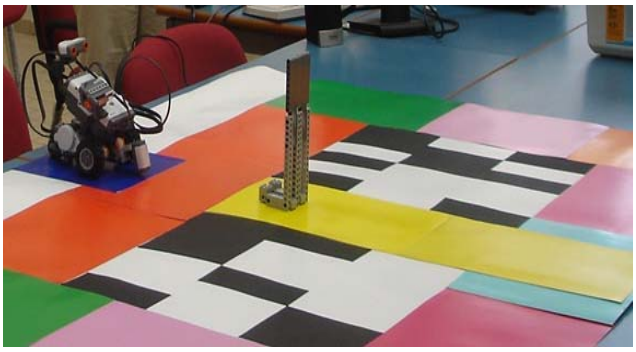


Fig. 4.5.3 Organizing seats in a theatre

This group consisted of a computer scientist and two physics teachers. In this project, school students are invited to construct and program a robot able to follow a predefined route in order to count the free seats in a theatre or cinema or ground or class etc., and inform the person in charge about the free seats of the whole place or a specific section (see Fig. 4.5.3). Extending the project, this robotic construction might also place audience in appropriate places according to their ticket.

Project 4: Easy parking

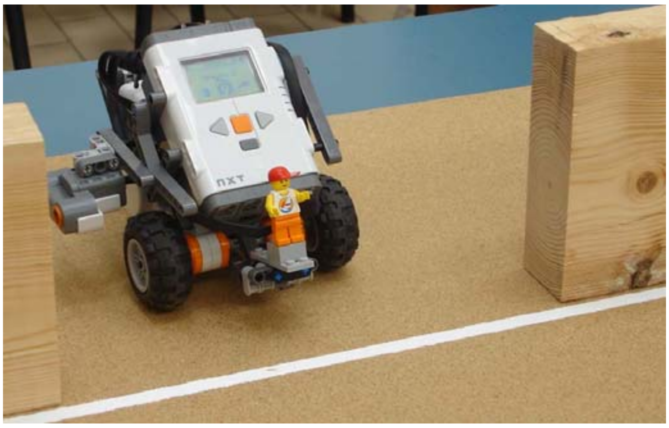


Fig. 4.5.4 easy parking

This group consisted of a computer scientist and an architect. In this project students are invited to construct a car-robot able to perform ‘easy parking’ on a mock up having several obstacles (see Fig. 4.5.4). The robot has to identify the blank spaces, to avoid obstacles by turning left or right, to stop, and finally to park at free car parking places.

Project 5: A moving car

This is an introductory project developed for primary education. In this project pupils are gradually guided to cultivate basic construction and programming skills. They have to construct a car robot and make it move forward, backward and turn left or right. Then a challenge is set e.g. to move the car through a specific route and then move it freely in any path. This project can be expanded to a game with many challenges.

Project 6: The catapult

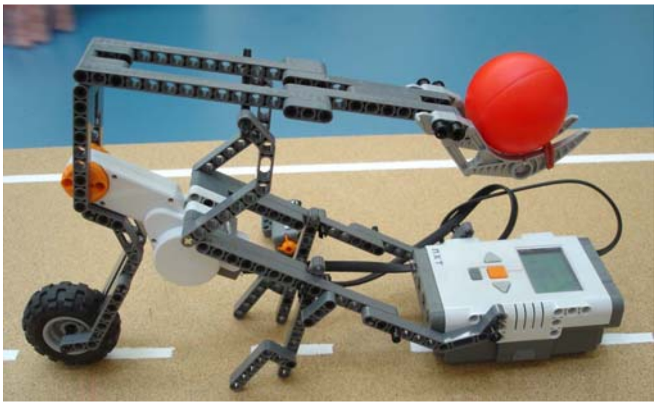


Fig. 4.5.5. *The catapult*

This group consisted of a mechanical engineer and two computer scientists. The project was designed for students of 15 and 16 years old. Students are invited to construct a robotic arm with one motor by following simple instructions (see Fig. 4.5.5). Then they should program it to throw small balls in a basket (projectiles). In order to make it work effectively, students should conduct experiments with the parameters involved like the length of the robotic arm, the motor power, the projection angle, the horizontal distance etc. Experimental data are collected and represented in graphs using appropriate software. Detailed examination of these graphs can help students to investigate relationships among the parameters involved. Finally students may continue playing a basketball game!

4.5.4 Conclusions and recommendations

In the training course a balanced whole of collaborative, learning- and teaching-focused approaches was adopted. The course evaluation was based on the trainees' products through the course and mainly on the projects they developed, on the questionnaires filled by the trainees and on semi-structured interviews at the enfd of the course. The preliminary results prove the potential of the training approach.

From the diaries kept by the trainees it appears that their statements as to "the best thing that happened to them" during the meeting involved focus on the practical activities, the creation of their own engineering structures and their programming. ("*We experienced the joy of creation, we built it up and it was operational*", Group A). It appears, indeed, that they enjoyed their work ("*When, ultimately, the robot moved along a square on the laboratory floor we rejoiced like young children*" Group C). Already from the initial activities some of them started thinking of scenarios for inclusion of similar activities in their own school class (Group F).

Conversely, among "*the worst things that happened to them*" they mention cases of "*very little practical work and quite a lot of writing*» (Group A), and, in some cases, lack of time required for the completion of their work. Their preference for practical work and their negative attitude towards "*theoretical presentations*" is also clear from the fact that they recorded among their negative experiences the case (assembly stage "*Bus Route*") where, because of lack of adequate time, their practical work was substituted for theoretical discussion (Group C).

Their positive experiences include the project-based learning method that they followed in their work and the exploration, experimentation and creation features included in that method, although some of the groups found that the proposal formulation process in respect of teaching strategies, which the trainees were asked for, was "*hurried*" and non-convincing (Group E) and that the implementation of the project "*bus route*" stage was not feasible (Group C).

From the questionnaire responses it appears that the training methodology of the course ensured their active participation, there was fair balance between practical activities on the part of trainees and presentations on the part of the trainers although some of them request even more activities and fewer presentations. They seem happy with the support and encouragement given to them by the trainers, while most of them asked for longer duration of the course. The difficulties mentioned by them are focused on robot programming elements (switches, variables) and the cooperation beyond the course, given that each group had only one Lego Mindstorms set at their disposal.

They evaluate the teaching materials used in the course as very useful. They state that the above materials as well as the training methodology followed in the course are worth applying either in their school classes with their pupils or in training fellow teachers (some of them work as trainers of teachers). The quantitative evaluations for the course range between very good to excellent (table 4.5.2) regarding all the aspects of the course that were evaluated.

Lastly, for the course improvement they propose even more emphasis to the construction work, additional and more complex examples of robotics activities, greater activation of the groups, increased sharing of ideas and projects between trainees via the internet and expansion of that communication to reach teachers of other European countries cooperating in the TERECoP project.

The above evaluations made by the trainees were confirmed by the group interview that took place at the end of the course (Papanikolaou et al. 2008).

The trainees' projects that were presented and discussed in the final session of the course followed the 5-stages methodology for designing robotics-enhanced projects that had been worked out during the training course. The presentation of the projects and the relevant materials (worksheets etc.) produced by the trainees indicate that the trainees efficiently adopted the proposed methodology. The projects address authentic problems from real life ('recycling garbage', 'saving water resources', etc.) and engage students in problem solving through exploration and investigation activities that exploit sufficiently the potential of the robotics technology.

Finally, the trainees’ answers and comments to the questionnaires and during the interviews offered evidence about the potential of the training course including the training methodology, the educational content, the use of e-class, the learning experiences and the integration of robotics in the school reality.

Training methodology: Trainees recognised their active participation in all the sessions of the course and their creative involvement even in the theoretical parts that introduced constructivist and constructionist principles and the methodology for designing robotics-enhanced projects. Several trainees emphasised that the educator’s axiom ‘*teachers teach as they are taught, not as they are told to teach*’ was really followed in the course. They admitted that they had a real experience of constructivism (“*It was for me a lesson of knowledge construction*”, “*Constructivism was present all the time in the course*”, “*this course was substantially different from the courses I had attended in the past*”).

Some comments focused on the synthesis of the groups, doubting about the efficiency of the criterion of personal relations for group formation purposes. Especially the group of the primary school teachers noted that “if a teacher of Informatics participated in our group, s/he would have helped us a lot…” Other trainees emphasised that the cooperation of teachers coming from different disciplines (maths, science, informatics etc.) is necessary for the successful implementation of the projects in school settings given that the projects are normally interdisciplinary.

The communication and cooperation between trainees and trainers was appreciated by the trainees as very supportive and helpful (“*we achieved a common language*…”) However, they suggest that the duration of the course should be extended and the development of their own project –or most of it- should take place during the course.

Concerning the *educational content* they very much liked the activity-orientation. They also liked that they had a real case of a project (‘*The Bus Route*’) to analyse the different stages of the methodology. They suggested that more examples and activities for homework would be also useful.

Concerning the use of *e-class,* most of the trainees evaluated the central role of the e-class during the face-to-face meetings and beyond them in enhancing social interaction and promoting a positive sense of community. They found the use of the web-based class as an interesting and useful experience that they wish to exploit in their work as teachers or trainers, although they think that its administration is a quite time consuming task. They acknowledged the timely provision of information, course content, and support when necessary. They mentioned that the discussion forum was mainly used for posting messages and not for real discussions since most discussions took place through face to face communication.

Learning experiences and integration of robotics in the school reality: Trainees appreciated the potential of educational robotics as a teaching tool but also as a subject in different disciplines such as technology, informatics, and engineering. A critical issue for integrating robotics-enhanced projects in the schools was how an interdisciplinary project may fit to the current school curriculum and schedule. Interesting ideas were proposed for integrating educational robotics in schools such as working interdisciplinary projects or research programs running out of the school schedule involving students from different levels e.g. engineers from technological education working with high school students. Trainees seem also to worry about the management of big classes during the implementation of robotics-enhanced activities in school settings ("*It will be difficult for one teacher to manage a school class of 30 students…*") and the cost of the necessary equipment.

Finally, trainees highly appreciated the opportunity to create their own project ("*a serious gap would have been left, if I had not worked on a new project within my group*"). They recognised that at the end of the course, they felt capable to implement the robotics technology in their school class ("*I understood how to exploit these new ideas and technologies in my school class*").

As an epilogue, we shall use the following very pointed statement made by one of the trainees, as, in our view, it describes in a concise manner the success of the methodology we applied: "*What I enjoyed most of all in the seminar was its planning. I liked the fact that we initially functioned as learners in the activities proposed, so that we crossed over to the other side, as a start, and, subsequently, we undertook the role of a teacher and came to the level of activity planning. I think that this kind of planning provided a quite comprehensive image of the usefulness and utilization possibilities of robotics in education*".

References

K. Papanikolaou, S. Frangou, D. Alimisis, (2008). *Teachers as designers of robotics-enhanced projects: the TERECoP course in Greece,* In proceedings of the SIMPAR 2008 conference/Workshop "Teaching with robotics: didactic approaches and experiences", University of Padova, 2008. Available at http://www.simpar-conference.org/

4.6 Experiences from the TERECoP course at the Charles University in Prague

Author: Daniel Tocháček

4.6.1 Introduction

A pilot course was organized by the *Department of Information Technology and Education* during the summer term of the academic year 2007/2008. This optional subject was offered to the 4th class students of the 5-year Master degree studies of the Technical and Information Education, in combination with another general educational subject for the 2nd level of elementary and intermediate schools (secondary schools). Twelve students enrolled for the course, two trainers were involved in its preparation and realization. The students attended six educational sessions, each lasting 3.5 hours. Beyond these sessions, students could take advantage of regular consultations. Due to the compulsory pedagogical practice, mandatory for the 4th year students, the course was divided into two parts: the first two sessions dealt predominantly with the theoretical problems of constructivism, constructionism and robotics, while the next sessions involved mostly practical activities. Alike as in the other partners' international courses, the goal was to prepare the future teachers of the Technical and Information Education for a factual use of the acquired pieces of knowledge and skills from the field of robotics in the constructivist education during their pedagogical practice.

4.6.2 The training course

The lessons took place in a specialized ICT laboratory of our department. Students had available personal computers with installed programmes, necessary for their working (the Software LEGO Digital Designer and the Software Mindstorms Edu NXT Programmer). All course participants had the possibility to work independently on their practical assignments – there were more computers and building sets available than students - however, working in couples has proved to be more practical. The necessary supporting materials were available through the standard learning management system LMS Moodle, currently employed at the department, e.g. for curricula, theoretical resource texts, instructive presentations, demonstration programmes etc.

The course has been realized in compliance with the standardized curriculum, created and approved by the project partners (see chapter 4). The only introduced modification had been splitting the course into two parts, in connection with the obligatory student work experience.

The individual sessions had the following contents:

1st Session – *introduction to the problem area; motivation; theory (1st part)*

- Students have been acquainted with the course curriculum, with the supporting course Learning Management System Moodle, with the educational materials.
- Interactive presentation of the working possibilities of the LEGO Mindstorms NXT building sets and of the relevant software.
- First part of the theory explanations (constructivism, constructionism, robotics).

2nd Session – *theory (2nd part)*

- Second part of the theory explanations (constructivism, constructionism, robotics).
- Discussion.

3rd Session – *practical activities – familiarization with the LEGO Mindstorms NXT building sets*

- Practical activities with the LEGO Mindstorms NXT building sets, familiarization with the sets, building of simple models according to instructions, programming with the use of example programmes and tutorials.

4th Session – *practical activities – advanced work with the LEGO Mindstorms NXT building sets*

- Practical activities with the LEGO Mindstorms NXT building sets, building of more complex models and their independent programming.
- Preparation of designs of advanced models with the use of the Lego Digital Designer software.
- Identification of assets and liabilities and discussion pertaining to the educational use of assembled models.

5th Session – *practical activities – robots employed as an educational object and training instrument*

- Demonstration of various educational projects with the employment of robots, constructed with the use of the LEGO Mindstorms NXT sets.
- Preparation and implementation of one´s own educational projects (just a part of activities, with their continuation out of school).

6th Session – *conclusion – project presentation, evaluation of the course*

- Presentation of the worked-out educational projects.
- Assessments of projects, evaluation of the course.

Approximately in the last third of the course, students were engaged in their own individual projects, pertaining to the educational use of robotics in the constructivist education. Results of their work were presented at the end of the course, when the evaluation of the course was also carried out. The resulting outputs formed the basis for the course appreciation, besides taking into account other available sources (e.g., outcomes from discussions with students, results of continuously tackled assignments, etc.).

4.6.3 Training experiences and results

During the entire course, in our capacity as the course organizers and educators, we had noticed an unusual excitement over both, the assigned tasks and the related non-compulsory exercises (that holds good especially for the area of practical dealings with the LEGO Mindstorms NXT sets). On the other hand, at the final stage, when the formal outputs of activities had to be presented in a prescribed standard form, we were astonished by the students' reluctance to complete successfully the started work. Approximately just one third of the student projects was of a high quality and was presented in a timely manner. The remaining students had finally met the requirements, but with a delay and their projects were of poorer quality.

Four projects have attracted attention:

School bus: The project deals with the simulation of the school bus task. The fully equipped robot uses three different sensors – optic for the perception of the line representing the way to school, sonar for passengers' identification, and touch for the final stop. The task of the robot is to go the whole way, stop by every pupil staying next to the way, arrive with all passengers at school and finish there.

Vehicle that observes the speed limits: This project is based on the use of pen fixed on the specialized arm. Robot can put it down and draw the line during the movement. The task is divided to two steps. It is necessary to set the motor of both wheels to the level on which the drawn line is straight first. Then the experiment is realized. The robot goes for 5 seconds, for 3 of them with the pen switched down. From the known time and measured distance the speed could be calculated. Finally the table of investigated speeds corresponding to particular engines sets is build.

Security guard: This project brings the theme of alarm designing to classroom. Pupils are introduced to the possibilities of different systems by the discussion first. Then the functions which could be fulfilled by the Lego Mindstorm are found – touch sensor for door opening, sonar for object penetration, general sound detection, smoke detection etc. Connection with the possible use of GPS and mobile connection is also discussed. Finally pupils make experiments with own robots. The task includes the schedule for round trips, violators warning and alarm switching.

Economy train: The main aim of this project is to improve the classroom environment, make pupils more active and involve the innovative pedagogy into the curriculum designed in correspondence with the new Czech educational system. The activity uses the normal wheel based robot equipped with the light and sound sensors. The robot as a train goes following the drawn line as a track. The economy principle is taught by the fact that the train stops on the station only if the sign by the hands clap appears.

All the projects mentioned above have been thoroughly elaborated, with an above-standard involvement of students in the implementation of project goals; besides that, the projects overreached the ICT region, penetrating into other disciplines.

Along the class sessions, a range of practical findings have been achieved, over the scope of the defined TERECoP international curriculum and beyond the experience of other European partners – so many findings that, presently, it was impossible to make a full use of them. It is hoped that these outcomes will be found useful in future courses at home and abroad. Let us introduce two examples:

The utilization of multimedia in the courses has proved to be very important. Throughout the courses, we took photos of some of students´ working processes and of some of students´ results and recorded them in video sequences. Unfortunately, these activities had not been performed systematically. We had realized only during the course that a well-thought out capitalization of the digital photography and of multimedia could be of a considerable use.

For the future, we might think of procuring a quality video recording of all important course parts and of using the records, for instance, for a distant support of other courses and the relevant assessments. We also intend to recommend to our students to make the best out of using photos and video records in their individual projects. Of benefit there might be, e.g., records of working procedures when constructing robots, records of robot activities, factual presentations regarding preparations of particular project segments, instructional pictures, videos, and the like.

A relatively profound support had been extended to the course participants on the part of the trainers. It still might be more extensive in the future. Even a material support might be envisaged, next to the present contentual assistance - for instance by lending set boxes to students out of the faculty, enabling them to work at home on their own projects, with a consecutive successful attestation of the results achieved at school.

4.6.4 Conclusions

Without any doubt, principles, methods and processes of the constructivist education have their indispensable place in the Czech educational institutions and complement properly the other, generally utilized concepts. However, a reasonable preparation of trainers and educators, who are willing to use this approach, is a necessary condition. Teachers of the Technical and Information Education, wishing to utilize this educational method, may profit from the potential of ICT and robotics, with the aid of which it is easy to realize many activities more easily and with excessively better results.

A preparatory course of such teachers on the use of robotics in the constructivist education has taken place in its pilot run at our working unit. It has offered students a possibility to learn more about the theories of constructivism and constructionism and has inspired them to take advantage of new educational technologies. A successful realization of these practices and the awoken positive response have persuaded us that the goals, course topics and thematic directions, as well as the choice of the students addressed, have been properly executed. An important aspect for a smooth realization of the course was an existing excellent background, based - among others - on the availability of an extensive theoretical base and on a quite satisfactory material outfit; this implication had coherence with the course inclusion in an international project (TERECoP).

Chapter 5

Exemplary projects and examples of learning activities with robotics

5.1 Two projects proposed by French teachers-trainees

Authors: Liliane Aravecchia, Luc Montel

This section presents two exemplary projects produced by our teacher-trainees within the framework of their training within the TERECoP project at IUFM, Marseille: Robotics challenge (trainees: BAUR Matthieu, CLEMENT Gabriel and VINCENZI Guillaume); Automated camera (trainees: Nicolas BOLDRINO and Laetitia CANDIDO)

5.1.1 First example: Robotics challenge

The project is based on the following challenge: A robot has to go from point A to point B either through a labyrinth with colored walls (white when the path turns left and black when it turns right) or following a black line on the floor.

This activity was implemented in a classroom of 28 secondary school pupils, aged 12-13, as part of their technology course treating the topic of “computer-aided piloting”.

The target skills aimed at are part of the French Technology curriculum. After the end of this project, pupils were expected to be able to:

- Identify the different parts of the robot ;
- Identify and justify the sensors and actuators used ;
- Represent the various stages of the movement by observing the robot;
- Modify an existing program according to the specifications given ;
- Adapt the system to a new situation.

The project was designed to be completed in 5 stages:

- Engagement stage: Pupils watch a video on robotics, followed by a discussion. The robotics challenge is then presented. (1h)
- I exploration stage: Pupils analyze the route their robot will have to follow from point A to point B and decide on a strategy to program the robot (30’)
- Creation stage: Pupils modify the existing robot by implementing the sensors and the program chosen according to their defined strategy (2h 30’)

- Evaluation stage: The different projects from each group of pupils are analyzed and compared by the class, and a synthesis is made by the teacher and the pupils (1h)

For the above project each group of pupils was given:

- One basic moving platform (Tribot), but without any sensors (the robot is given partly built due to lack of time to let the pupils do the building)
- A computer with NXT-G software for programming the robot.
- Different Lego sensors and Lego parts to modify their robot.
- One of the arenas the robot has to cross (labyrinth, obstacles or black line).

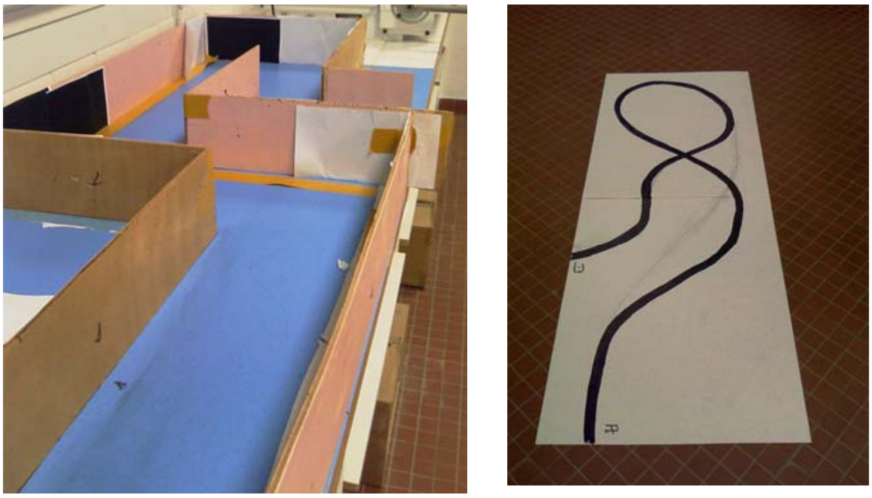


Fig. 5.1.1 *The Labyrinth and the Black Line challenges*

Teacher's guide:

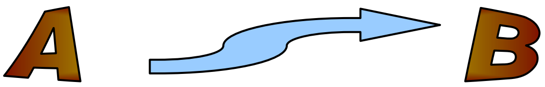
Stage	Activities and Progress	Class Organization	Time
Engagement stage	Introduction	Full class	5'
	Playing the video	Full class	20'
	Discussion on robotics		25'
Exploration stage	Presentation of the robot, the teacher asks "how will it be able to cross the arena?" Each group receives its robot, sensors and arena		10'
	Pupils analyze their arena and define a « way » for the robot to go from A to B They write it down in their own words at the back of the document they were given	Group of 3	20'
Creation stage	The pupils choose the sensors suited to their situation and install them on the robot. The teacher questions the pupils on the reasons of their choice	Group of 3	20'
	Pupils finish the robot and then go to their arena where they discover the necessity of the right program in order to pilot the robot. Then, they go to the computers and use the NXT-G programming software Teacher gives instructions on how to use the software	Group of 3	1h 40'

Evaluation stage	After clearing up , questions are asked by the teacher to see if the knowledge was understood Each group observes the working of the other groups' robots, try to deduce their logic of programming and justify the sensors used. A synthesis is made before a written evaluation in class	In groups	55'

Pupil's worksheet:

You have at your disposal a robot capable of evolving in its environment…

The robot has to go from A to B…



The robot moves along a straight line, collides with the first wall in front of it … and stays stuck

Why??

Some hints….

1. Analyze the arena the robot has to cross: What are its characteristics? What parts of the arena can help to guide the robot?
2. Propose a logical way to go from A to B
   - Write down your proposal here :
   - What are your conclusions?

Experiences and feedback

The main difficulty encountered, when this course was implemented in a classroom, was passing from the description in natural language working with the robot to the programming in formal language.

Engagement stage

In the first stage, the viewing of the video and the presentation of the challenge was aimed at catching the pupils' interest in the project and at bringing to light any lack of knowledge, any need to learn. This stage was successful; it induced a lot of interest and curiosity in the pupils. The concept of a challenge allowed to stimulate the children by an effect of competition. This means is effective, but has to be used with care in order not to value systematically the "winners" at the cost of the "losers".

Exploration stage

This stage consisted in the analysis of the course the robot had to follow and revealed the first difficulties. Most of the pupils, having observed quickly the circuit, tried to find a 'sequential' way of crossing the arena. For example, for the "labyrinth", most of the pupils said: "it is necessary to turn right, then left, then left, then right …" It is then advisable to question them in order to boost their reasoning: "but if I now want to go from B towards A, does it work?" Or still "observe, please, the elements of the course which can help you ".

The reflection thus resulted in a "logic", which allows crossing the arena in both directions, of the type: "if we meet a black wall, we make a quarter of a turn to the right, if we meet a white wall, a quarter of a turn to the left". This is true in particular of the arenas labyrinth and obstacle. The groups working with the "black line" had more difficulty because the logic to be built in this case is more complicated

Creation stage

During the designing phase, the first observation which we can make is that the Lego Mindstorms NXT support federated at once the interest of the pupils. Indeed, after the observation of the arena to be crossed, the implementation of one or several necessary sensors was realized without problem by the vast majority of the pupils. They often used the guide of assembly supplied in the box in a mechanical way. The main difficulty was to choose the adapted sensor or, more exactly, to understand the role of every sensor. Some guiding from the teacher helped them.

The contribution of a document allowing the pupils to choose the sensor independently (by describing the role and function of each of them) could turn out to be useful (fewer interventions on behalf of the teacher). We notice that robots are assembled in a very ill-assorted way, while all the groups used the same guide of assembly. We can thus conclude that the association of the material element with its representation is not without raising problems and calls for apprenticeship over the duration.

Furthermore, certain groups chose different sensors for the same use: for example, the detection of an obstacle can be made by a touch sensor or by an ultrasonic sensor. This is interesting and can be exploited during the synthesis by bringing to light the advantages and the disadvantages of each solution.

Once the sensors were installed, the first groups did not identify at once the necessity for programming the robot. They thus went on the arena, placed the robot down, switched it on and realized that it remained totally immovable. We asked them at this moment the question “why would it move if nobody gives it the order?” The role of the programming was then justified.

During the programming itself, the principle of 'blocks' was easily understood. On the other hand, loops and conditional actions raised more problems. It was thus important to establish a link between the logic in the form of a sentence and the software representation. To do it, the pupil has to formulate in writing the actions in a sequential way, by means of a compulsory syntax, facilitating the transition in the programming language under the shape " action if, until, as long as etc. ". For example "I move until I meet a wall, if it is white I turn to the right etc". This stage in natural language is a necessary step before programming in the language of the robot.

The difficulty for the teachers was here avoiding giving too many solutions to the pupils, while helping them not to remain blocked because of the programming language, which is not the object of learning here. In the case of this activity the logic behind programming is a skill that the pupil has to acquire, not the language itself.

5.1.2 Second example: Automated camera

The problem: A firm sells an automated production line that fills packages and packs flasks of different types. In order to present its machine to future clients, the firm wants to make a video of the course followed by a flask along the production line. In order to follow the progress of the flask, a robot with an onboard camera will be used.

This project was designed for a group of 12 pupils of age 16, in their first year of technical secondary school in the field of “Maintenance of Industrial Plants”.The learning objectives of this activity are linked to kinematics. The aim is to enable the pupils to define basic notions, such as trajectories (indifferent, rectilinear and circular) and movements (linear and rotation).

Progress of the teaching sequence: the project was planned over 4 hours, during one day (2 hours in the morning and 2 in the afternoon).

- Engagement stage: The teacher presents the problem to be solved to the pupils (they have seen the production line in function before), as well as the Lego NXT kit and programming software. The next hour is used by the pupils on building the robot with the help of an assembly guideline.

- Exploration and Creation stage: The pupils have to retrace the course of the production line “ERMAFLEX” with their robot.

- Evaluation stage: The different results from each group of pupils are analyzed and shared by the class and a synthesis is made by the teacher and the pupils.

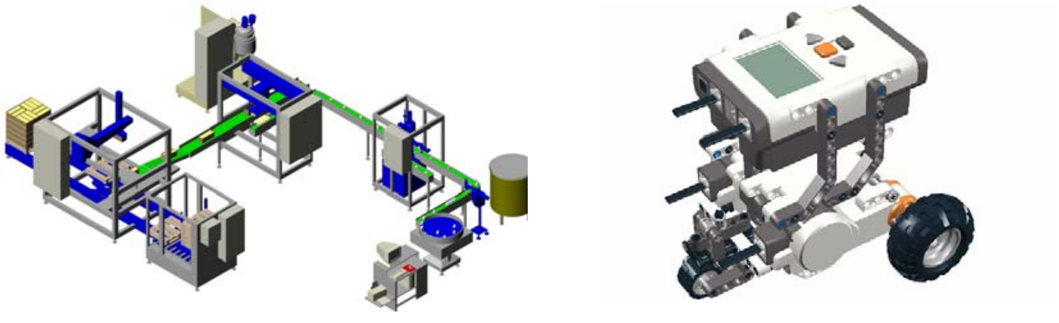


Fig. 5.1.2 *The production line and the robot*

Teacher’s guide:

Stage	Activities and Progress	Class Organization	Time
Engagement stage	Introduction	Full class	5'
	Presentation of the robot and the programming software	Full class	25'
	Teacher divides the class into groups, allocates a robot to each group and gives instructions	Group of 3	15'
Creation stage	Each group constructs its robot		55'
Exploration and creation stage	The pupils analyze the different programming blocks and try them on their robot, they then decide on a program solving the problem of following the path of a bottle on the assembly line	Group of 3	90'
Evaluation stage	Each group observes the work of the other groups’ robots, try to deduce their logic of programming and justify the sensors used. A synthesis is made before a written evaluation in class	Full class	20’
	Each group stores away its equipment, the teacher verifies that nothing is missing	Group of 3	5'

In this example the main goal is to enable the pupils to define the concepts of basic trajectories (rectilinear and circular) with the discovery of linear movements and rotation. In order to really work on the definition of trajectories and movements, it seemed important that the pupils should use blocks already preprogrammed by the teacher with simple movements such as: uniform rectilinear movement, uniform circular movement (left and right), accelerated rectilinear movement, decelerated rectilinear movement

So, by using these preprogrammed blocks to program Lego Mindstorms, the pupils can set up a fast experiment with the aim of recreating the route followed along the production line. The pupils will determine first the function of these various blocks and then, in order to solve the problem, they will create a program. In that way they will not waste time understanding how to program completely the robot and they will concentrate only on the realization of the route by a simple association of these blocks.

Pupil's worksheet:

Short description of the production line: The production line is used in industry, to manufacture, condition, pack and palletize several products in various containers. It makes it possible to fill bottles or flasks while following the process from the distribution of empty bottles to the palletization of packed filled bottles.

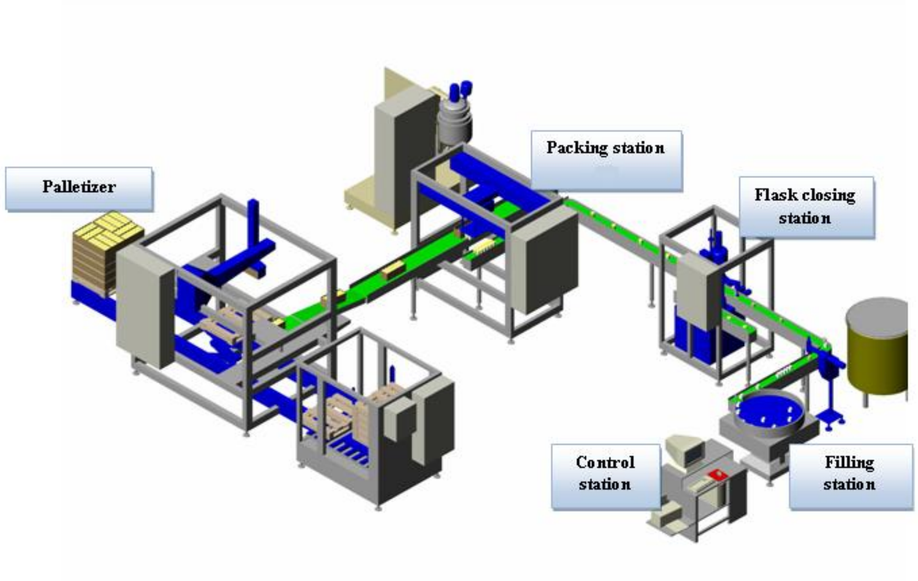


***Fig. 5.1.3** The structure of the production line*

Problem: With the aim of presenting the principle of operation of the production line, the company would like to carry out a video of the course, followed by a bottle on this production line. To follow the path followed by the bottle, a Lego Mindstorms programmable robot with an embarked camera will be placed at your disposal. This robot will allow, thanks to the realization of a program recalling the route followed by the bottle, to film, with the embarked camera, the course of the

ERMAFLEX production line. Before testing the program on the real production line, we will recreate this course in a reduced scale.

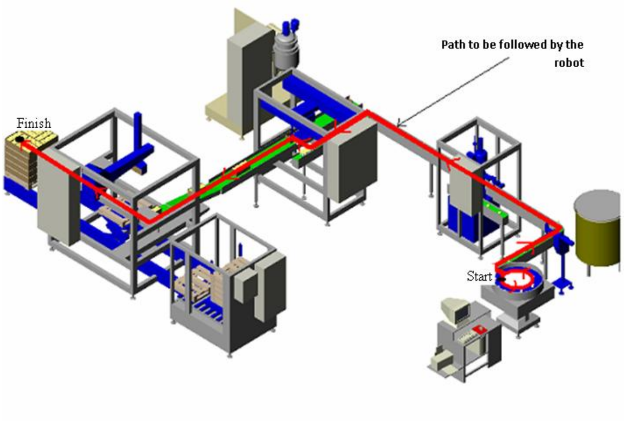


Fig. 5.1.4 *Path to be followed by the robot*

Programming task: To carry out the experimentation, you have:

- 1 Lego robot
- “MINDSTORMS EduNXT” programming software
- 1 USB cable allowing connection of the robot with the computer for programming

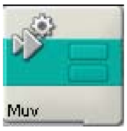
Programming instructions:

1. Launch the software.
2. Click on the icon “start a new program”.
3. Click on the Palette icon personalized.
4. In the icon “My blocks”, select a macro by clicking.
5. Position the macro in the programming window of the software.
6. Repeat stage 4 and 5 as many times as necessary, according to the macros that you want to test or to the program that you want to realize.
7. Once the program is finished, connect the robot with the computer via cable USB.
8. Start the robot by pressing on the orange button.
9. Click on the Download icon in order to load the program in the robot.
10. Await the end of the compilation and disconnect the robot.

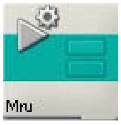
11. Place the robot on the ground and press the orange button of the robot to launch the program.

Experimentation:

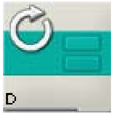
✍ Instructions: Explain for each image of the blocks below the behaviour carried out by the Lego robot.



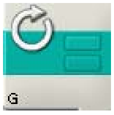
Answer: uniformly accelerated movement
....



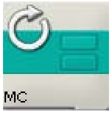
Answer: constant speed movement
....



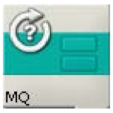
.... *Answer: turn right*



Answer: turn left
....



.... *Answer: half turn right*



.... *Answer: half turn left*

Experience feedback: This project was implemented by two student-teachers in one classroom and was compared with a more typical lesson treating the same subjects. The results of the comparison of the two different teaching methods (with or without the help of educational robotics) was presented by the student-teachers in their professional reports, as part of their evaluation as teachers trainees. In this example, the students had great difficulties in designing a course using a constructivist approach because it had to fit in a curriculum that was built on an approach based on skills, and prevented them to focus on the construction of knowledge in the chosen situation.

5.2 Robotics-based educational tool for an airplane servo-loop model

Author: Silviu Ionita

The proposed educational tool is an example of robotics-enhanced learning project for students, which originated from the TERECoP project. As some other examples approaching “Project Based Learning with Robotics”, this project was developed for the training needs of the pilot training course in the University of Pitesti, Romania, which took place during the three years of the project implementation.

5.2.1 Teacher guide

Title of the activity: Learning on airplane servo-loop control by building a robotic demonstrator.

Introduction: This example addresses a complex topic and is appropriate for several educational purposes on general curricula, as well as for vocational disciplines. The key teaching issue addressed by this learning activity is focused on model and the students build themselves the learning tools.

Goals: The goals of an airplane servo-loop model as educational tool are divided into three classes: cognitive objectives, skills and attitudes as follows:

- *Cognitive objectives* include basic knowledge obtained in the area of airplane engineering focused on specific issues from mechanics and mechanisms. The understanding of the aircraft control problem involves, also, knowledge acquisition on related issues in the field of flow dynamics and aerodynamics. Basic knowledge on sensory (ultrasound transducers) and telemetry (ultrasound-based impulse range finder) are expected to be acquired with the proposed robotic model. The key cognitive goal of this example is the systemic understanding on the control issue for a body with six degree-of-freedom (particularly for an airplane). The strategy of control involves knowledge on cybernetics (error-based automatics, feedback concept, multi-loop control). Finally, this example helps the trainees to perceive the effect of fins and, also, helps the trainer explain the effect of combined rudders’ deflection.

- *The skills* expected to be acquired with the robotic demonstrator for airplane servo-loop system involve both, creativity and ability to implement servo-loop controllers. The proposed application will urge the imagination to build specific mechanisms for aircraft serving with the elements from Lego kit.

- *The attitudes* expected to be developed with this project are also very relevant for trainees. They have to reason in terms of technical correctness, realism and feasibility of the proposed technical solutions. Comparative and critical thinking, as well as optimal reasoning, are the typical mental skills related to the attitudes to be developed with this example. Building a flying machine, either a real one or a model, is a question of responsibility, accuracy and reliability of technical solutions. Finally, this example creates a positive attitude for collaborative work and co-creation.

The example “NXT/Robotics-based educational tool for an airplane servo-loop model“ was created for a large age group of trainees, typically from 15, including adults, too.

Technical details: The activities in this example are organized modularly. Each module is provided in the face to face session of 2 hours. It is recommended to plan the entire course (all modules) during one or two weeks. The prerequisites are depending on the group of trainees-the age and their level of familiarity with Lego Mindstorms educational kit. For groups that are not familiar with the educational toolkit, an optional introductory session on Lego Mindstorms/NXT kit including software development tools is provided. In the following table the basic guidelines of the teaching process are presented.

Name of module (duration of session)	Activities carried out during the session
1. Introduction (2 hours)	1.1. Identifying the *learning needs* and *goals, expectations* and possible learning difficulties. 1.2. *Challenging and motivating* the trainees. 1.3. *Harmonizing the heterogeneous group* of trainees in terms of their background and experience. 1.4. Providing a generic course on *robotics as educational tool*- constructivism, constructionism and project-based learning.
2. Engagement (2 hours)	2.1. Identification and presentation of the scientific problem. 2.2. Trainer defines the problem of *airplane servo-loop control*. (Exposes the principles and gives the explanations referring to real life examples).
3. Getting started with robotics(*) (2 hours)	3.1. Dividing the group of trainees into *smaller work teams*. 3.2. Providing the *introductory course* (*) about Lego Mindstorms/NXT robotic toolkit. 3.3. Trainers challenge the trainees to imagine differ-

	ent constructions and the associated programs that could play the role of demonstrator and *help them in learning* different emerging disciplines.
4. Designing the project with robotic toolkit (2 hours)	4.1. Trainer and trainees identify the technical requirements for specific mechanisms. 4.2. Trainer assigns each specific mechanism to be developed by a group. 4.3. Trainer supervises the running of tasks paying a special attention to the following issues: - How the trainees are dealing with the particularities of the toolkit, encouraging an enthusiastic, uncritical attitude among trainees. No criticism of ideas! - The ideas developed and contributed by every one of the trainees, during the session, taking notes of any such ideas. - Relevant ideas that come out of the session.
5. Demonstration (2 hours)	5.1. The outcomes from the previous module are integrated by the students in a robotic demonstrator tool for the *airplane servo-loop controlling*. 5.2. Trainer performs different scenarios with the demonstrator in order to exemplify how the aircraft guiding systems work. 5.3. Trainer helps the trainees to adjust or to refine the software application on NXT controller in order to realize the effects.
6. Course evaluation (2 hours)	- Trainer makes a synthesis by summarizing trainees' ideas, as they emerged. - Trainer makes a comparative analysis of the ideas that were noted during the session. - *Structured questionnaires* should be applied immediately after the end of the course, simultaneously to all participants. Note. *Regarding participants that worked individually, the answers provided express their personal opinions and views.*

	- Conclusions.
(*)Introductory course on Lego Mindstorms/NXT (2 hours-optionally/ if needed)	Trainer presents the guide lines of the work with the use of slides on the following issues: - Robotic Lego Mindstorms demo (use of Lego Digital Design, pictures of already constructed robots). - Software development tools. a) Trainees work effectively in small groups proceeding *step by step to get familiarized* with the robotic Lego kit and software development tools. b) Trainees explore the *mechanical parts* focused on their function and practical lab exercises on assembly work, (lab activity, groups, 30 minutes). c) Trainees browse the NXT *brick menus*, (lab activity, groups, 30 minutes). d) Trainees explore the Lego Mindstorms software development tools and robot *programming*, (lab activity, groups, 30 minutes). Note. *Each trainee receives a copy of Lego Mindstorms software programming tool to study independently the help section of the functional blocks.*

Rationale of the teaching approach: The overall approach in this example is interdisciplinary, but the activities involved here can be implemented in many disciplines: Computer Science, Technology, Mathematics, Science and Engineering. The specific fields from Engineering are specially addressed in Control engineering, Flight mechanics and Aeronautics. The airplane servo-loop demonstrator is fully developed and is ready for teaching purposes. The aircraft control problem has a significant degree of complexity that requires knowledge from different disciplines.

Working in small groups, the trainees build different robots, making their own programs in order to demonstrate several rules, principles and techniques from physics, mathematics and informatics as follows:

- The Newtonian principles applied to a body with six degree-of-freedom;
- The rule of *forces composition* and the couples of forces working;
- Geometry of triangle and elementary trigonometry;

- Acoustic waves reflection and propagation;
- The principle of *closed-loop control* and feedback;
- Basics of *programming*, for instance: relational operators, conditional instruction, loop control, etc.

Other topics are also highlighted, for instance: *levers law*, *kinematics*, *reduction gear*, and so on.

The issue of airplane guiding by automatic control of its rudders can be understood in the context of preliminary knowledge on aerodynamics. Some possible students' difficulties can be avoided by using the robotics based educational tool for the airplane servo-loop demonstrator. We propose an innovative teaching approach with the use of the robotic construction to emulate an aircraft. Building a mechanical structure with feasible mechanisms to actuate the airplane's rudders challenges the students to find innovative solutions, other than those usually used when they build wheel-driven mobile robots.

The proposed scenario takes into consideration the key teaching issue addressed here: *learning by demonstration through building the demonstrator*. So, the experimental environment is defined by the idea of learning precisely by building the learning tools. Building the learning tools, in this case by using robotics, is performed by the following tasks in an experimental environment, as depicted in Fig. 5.2.1.

According to the constructivist and constructionist approach, the activities with students are organized in the following methodological: Engagement stage, Exploration, Investigation, Creation and Evaluation. As a matter of fact, to organize a project in robotics, the content of the individual stages are highlighted as follows:

- In the *engagement stage* students are provided with the open-ended problem and get involved in defining the project. This stage requires the identification and presentation of the scientific problem. Students work as a class putting their ideas into a question format as follows:
    - If I am piloting a plane, which command or commands should I actuate in order to: lift, dive or turn the aircraft?
    - Which are the elements that command effectively the airplane?
    - How do the forces that move effectively the aircraft work?

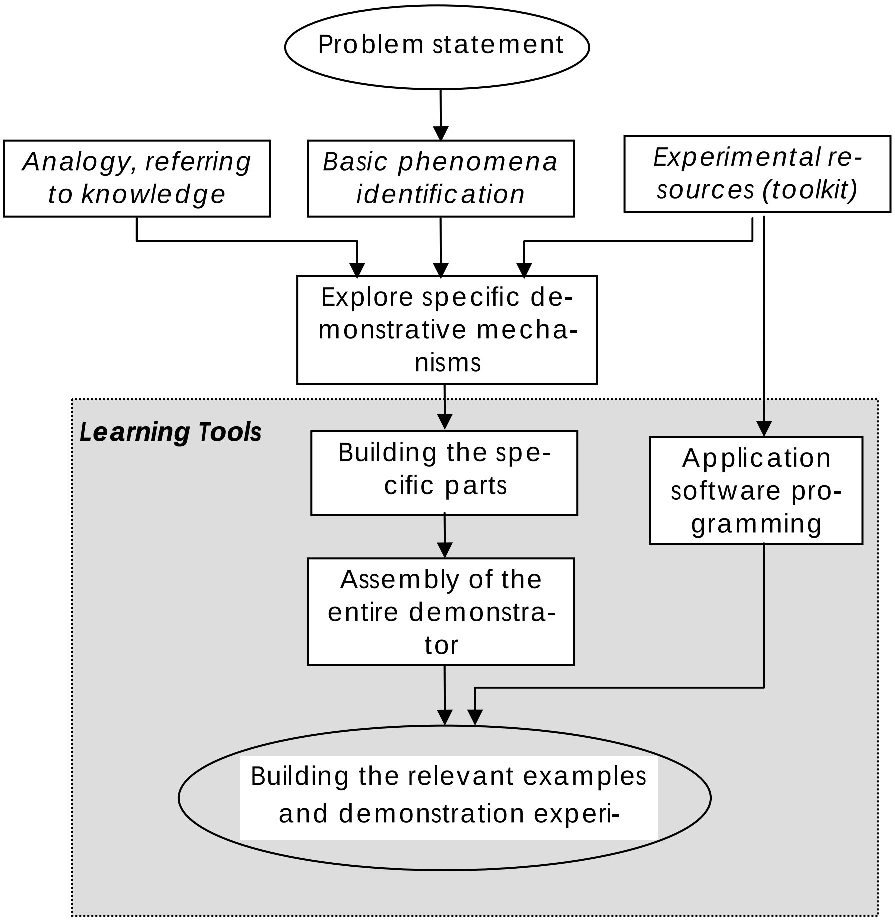

Problem statement
Analogy, referring to knowledge
Basic phenomena identification
Experimental resources (toolkit)
Explore specific demonstrative mechanisms
Learning Tools
Building the specific parts
Application software programming
Assembly of the entire demonstrator
Building the relevant examples and demonstration experi-

Fig. 5.2.1*-The learning environment*

- In the *exploration stage*, students get familiar with educational Lego Mindstorms kit, controlling devices and software, make hypotheses and test their validity in real conditions, provide initial ideas and preliminary solutions. This stage is performed in a flexible manner, depending on the level of students' familiarity with the educational toolkit. If the students are well introduced in the technology of Lego Mindstorms robotic toolkit, this stage will be adjusted adequately. On the contrary, if the students see for the first time the Lego Mindstorms kit, a special introductory session will be provided. In this case, the trainees should work effectively in small groups proceeding step by step to get familiarized with the robotic Lego kit and software development tools.

- In the *investigation stage,* students search for resources and investigate alternative solutions. Students reconsider the problem and the different issues arisen during the engagement stage, based on their experience gained through the exploration stage. In addition, the students involve their preliminary knowledge, their beliefs and own experiences via analogous reasoning and intuition to imagine specific demonstrative mechanisms. At this stage, students, in collaboration with the teacher, formulate the driving questions which link with the learning goals of the project, as they were defined: cognitive goals, skills and attitudes.
- In the *creation stage,* students share and combine their artifacts, synthesize ‘solutions’ to the project, reflect on their initial ideas. Students present their work in class and then work on the synthesis of a final product, including the artifact and the software, and reach a common proposal for the project. The result is the assembly of the final demonstrator. This work may lead to similar solutions, but, also, to innovative proposals. In this stage, the relevant part of the learning tools is achieved. Moreover, the demonstrator for airplane servo-loop control becomes itself a learning tool supporting related experiments and future developments.
- In the *evaluation stage,* the trainer and the students make together a synthesis by summarizing trainees' ideas, as they emerged from the preceding stages. Now, students share their ideas, products at class level, argue about their final proposals and evaluate them. Alternative solutions are also presented at class level and evaluated based on the driving questions raised in previous stages of the project (stages of engagement, investigation). All the solutions noted during the course are subjected to a comparative analysis on strengths and weaknesses. At this stage, students should critically judge their work, express their opinions and compare their works. The evaluation stage is considered a conclusive lesson rather than an assignment. In order to obtain a measure of the learning efficiency, with this example, some *structured questionnaires* could be applied immediately after the end of the course, distributed simultaneously to all participants.

5.2.2 Fully developed example

The fully developed example, based on an educational tool for an airplane servo-loop demonstrator, is presented here in terms of technical and methodological principles.

The educational tool is briefly described with main emphasis on the function of the proposed robotic construction and the teaching contribution: the questions, the problems, the experimentations and the investigations that students can perform with the suggested demonstrator. The entire educational purpose is defined by questions such as: “How is an airplane piloted?”, and “How the aircraft can be automatically controlled?”

The problem statement starts with the presentation of the basic notions on the airplane’s movement according to three-degrees of free rotations, which are the subject of control, as depicted in Fig. 5.2.2.

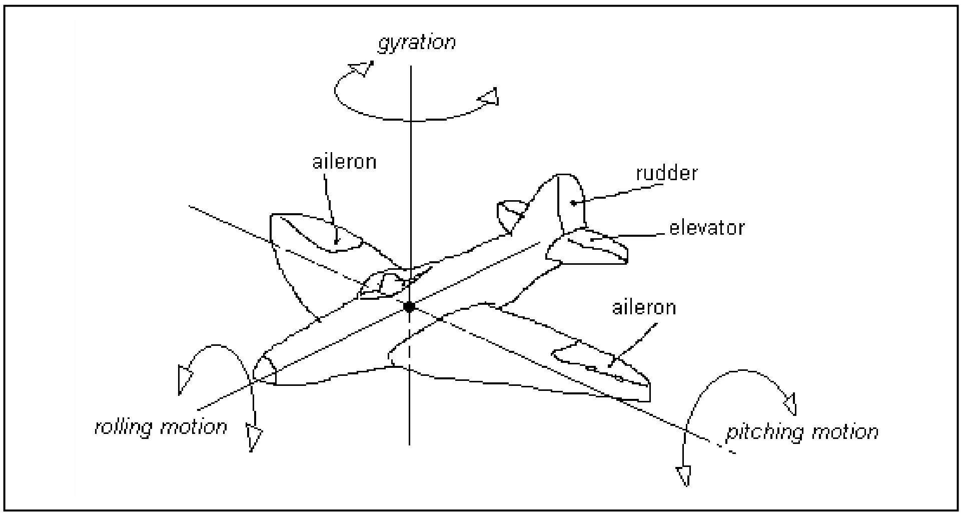


Fig. 5.2.2 *The airplane and basic controlled movements*

The next step refers to the connection between rotations and the three-dimensional displacement of the aircraft. The dependency between the three controlled rotations around the aircraft’s center of gravity and the translations along the three axes are explained in terms of aerodynamic effect on the fin-rudders, as depicted in Fig.5.2.3.

Under these circumstances, the trainees should have clearly in their minds the cause-effect relationships that govern the evolutions of the airplane. The main relationship is between the *propeller thrust* and the *longitudinal driving* that defines the airplane’s lifting and all the related aerodynamic effects. The propeller thrust is controlled by the pilot via the gas-throttle.

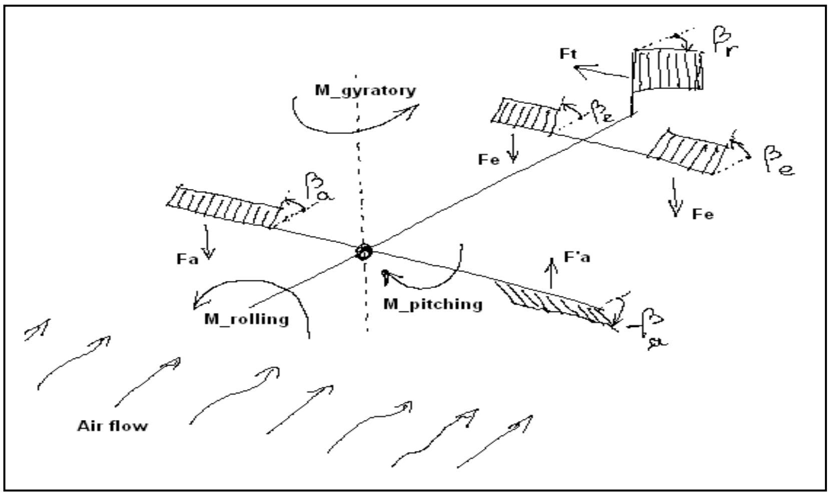


Fig. 5.2.3 *An intuitive draft on rudders' effects*

Table 5.2.1. *Synthesis of the cause-effect relationships governing the airplanes evolution*

Cause		Effect			
Pilot's command	Rudders' state	Translation	Rotation	Aircraft's evolution	
Handle (stick) right	Aileron right: up-deflection, Aileron left: down-deflection	Lateral	Rolling	Lean over on the right side	Banking to right
Rudder bar (pedal) right	Tail right deflection	Lateral	Gyration	Turning to right	
Handle (stick) left	Aileron right: down-deflection, Aileron left: up-deflection	Lateral	Rolling	Lean over on the left side	Banking to left
Rudder bar (pedal) left	Tail left deflection	Lateral	Gyration	Turning to left	
Handle (stick) forward	Elevator down deflection	Vertical	Pitching	Diving	
Handle (stick) backward	Elevator up deflection	Vertical	Pitching	Climbing	

Based on the rationale described above, in the next step, the airplane servo-loop control is introduced, which, in fact, is a basis of an automatic pilot. The task of the autopilot is to maintain a prescribed flight altitude and, possibly, to avoid air collisions of the aircraft. In this example, the servo-loop control system includes sensors, servo actuators, feeds, comparators and signal references. There are two separate main control channels in this case, as depicted in Fig. 5.2.4.

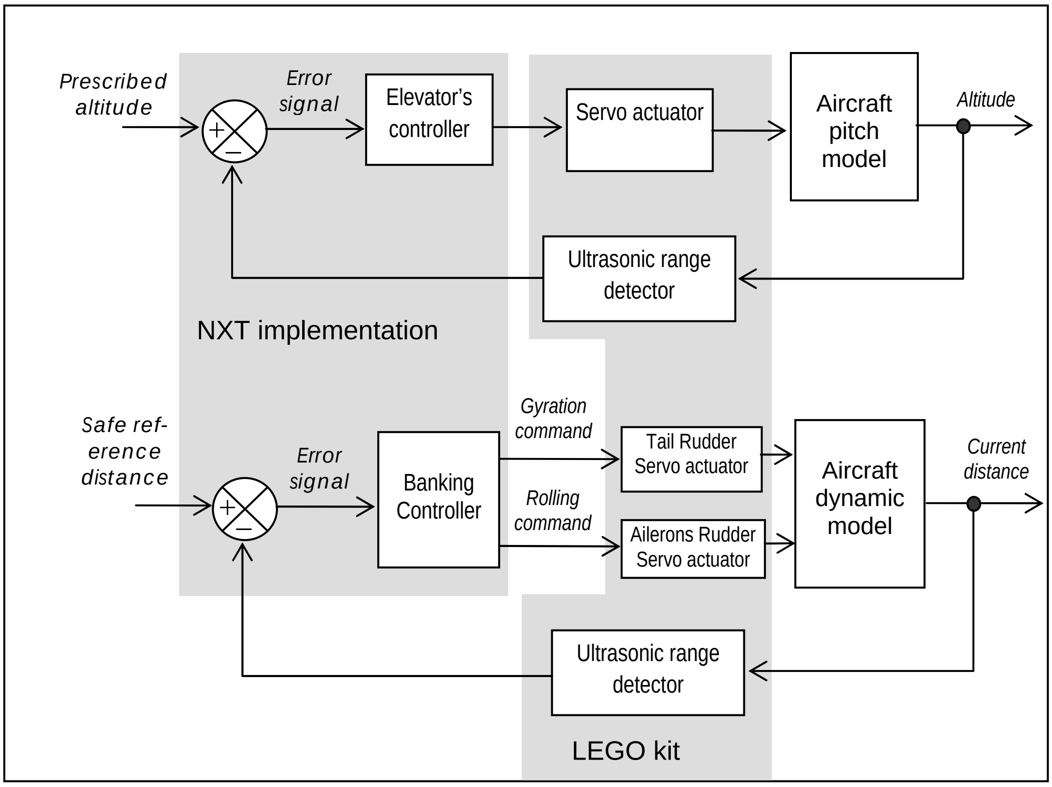
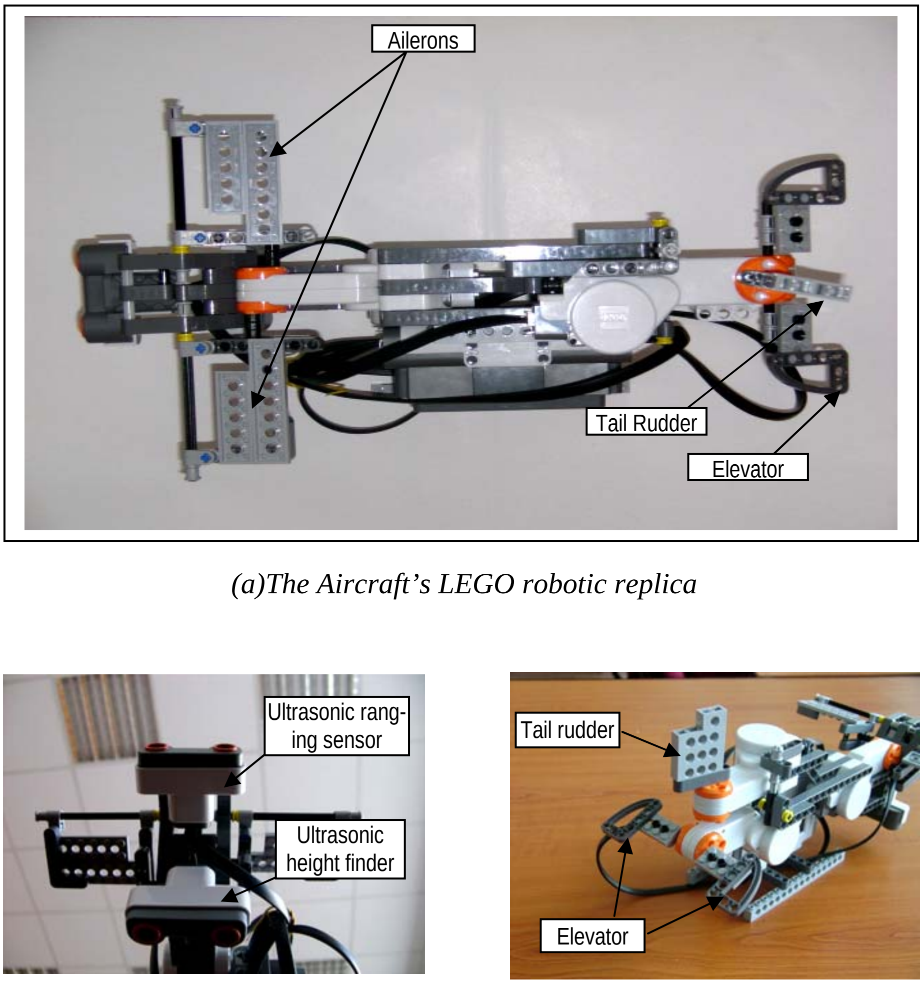


Fig. 5.2.4 *Block diagram of the control and the parts that can be implemented with Lego Mindstorms toolkit*

The following paragraphs are dedicated to the constructive details of the NXT Lego kit, based on the educational tool named demonstrator. Figures 5.2.5 (a) to (d) provide some annotated pictures of the demonstrator's mechanical structure, including sensors and the NXT brick.

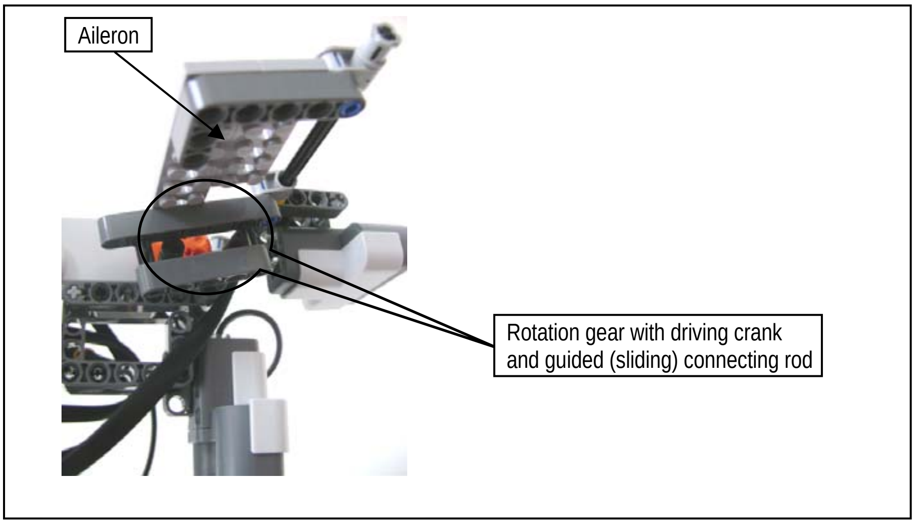
The demonstrator is programmed to emulate the control of an airplane by highlighting the cause-effect relationships during the following scenario: keeping a safe altitude and avoiding any obstacle that is heaved in sight, in a certain prescribed range. The NXT brick was programmed according to the above mentioned tasks. The control algorithm reflects the principle of closed-loop adjustment



(a)The Aircraft's LEGO robotic replica

(b) A front side bottom-up view of the demonstrator

(c) The back side view of the demonstrator


Aileron
Rotation gear with driving crank
and guided (sliding) connecting rod

(d) The details of the aileron driving mechanism

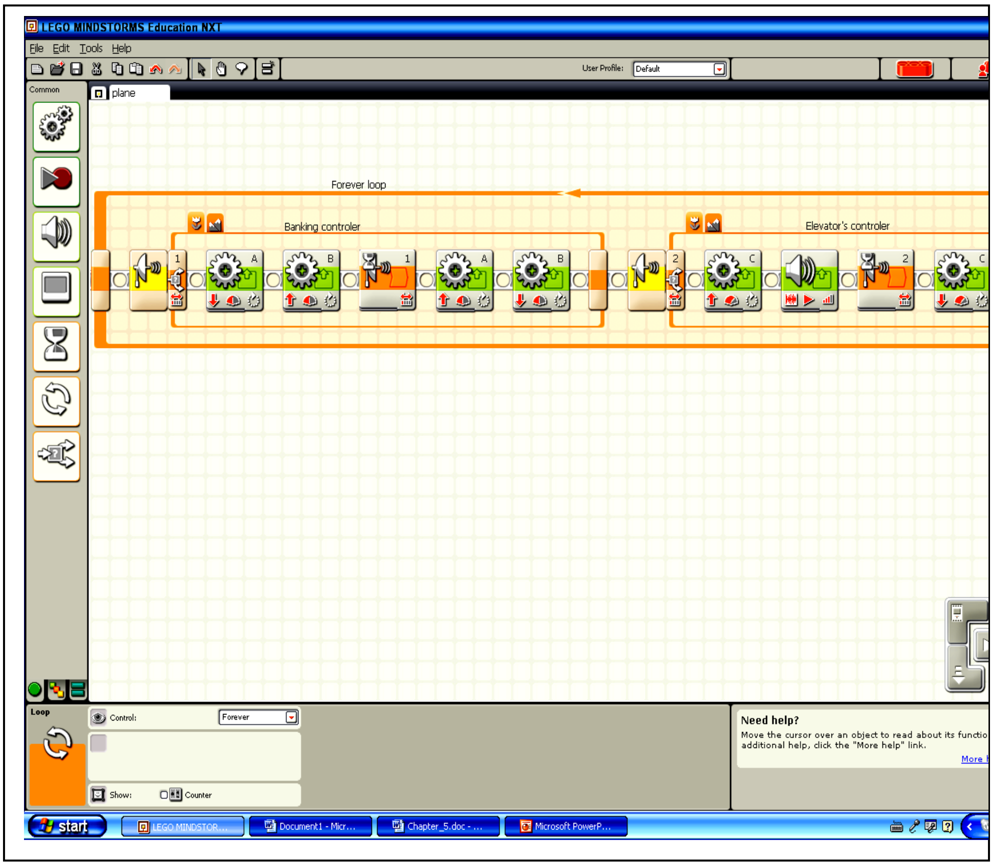
Fig. 5.2.5 (a, b, c, d) *The demonstrator*

An effective software implementation of the demonstrator is presented in Figure 5.2.6. The two main control channels introduced in Fig. 5.2.4 are implemented using LEGO Mindstorms education NXT software involving the common palette of programming blocks. The result is the introduction of three distinct controllers into a single program. The altitude controller is designed as an independent one, but the banking controller consists in two coupled controllers for both: the gyration and the rolling. This is because, as well known, the correct banking evolution imposes the correlation of the aircraft's gyration with its rolling.

An important part of the automatic servo-loop control system is the ultrasonic sensors that play the role of range detectors (for the danger altitude and the frontal collision). For practical reasons, the ultrasonic sensors are set up for ranges less than 0.8 meters. The motors of the servo-actuators are programmed to execute rotations up to 30 degrees in both senses.

Let us note that the demonstrator presented here is not a mobile robot. It was designated as an educational tool to be maneuvered by the trainer during the demonstrative scenario according to the learning purposes. Figure 5.2.7 presents an instance demonstrating the reaction of the model in case where safe limits for both distance and truth altitude are violated.

A typical scenario for demonstration can be described as follows: The trainer puts the demonstrator on a table in such a position that the ultrasonic height finder sensor is beyond the table's edge catching a signal reflection from the floor.



User Profile: Default

Control: Forever

Show: □ Counter

Need help?
Move the cursor over an object to read about its functio
additional help, click the "More help" link.
More I

***Fig. 5.2.6** An example of control implementation*

In this position, the NXT program is started and the servo systems should not react. If any sensor detects an obstacle (for example, hand presence) in front of it, the demonstrator exhibits the appropriate reactions by deflecting its own rudders._The same effects can be highlighted if the trainer takes the demonstrator in his hand simulating a flight-path and approaching different obstacles in the class (walls, floor, etc.).

5.2.3 Final comments

The demonstrator proved its usefulness as: a *learning object* and a *learning tool*. From the constructivist approach, building the demonstrator was a challenge. The result is an advisable learning tool in the area of aircraft. Some constructive limits have been discovered in terms of learning object regarding the certain possibilities to joint the LEGO parts.

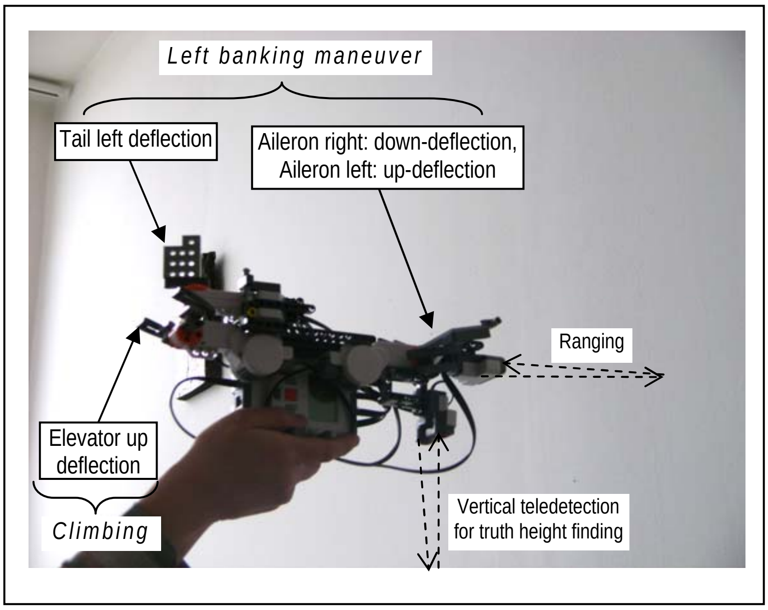

Left banking maneuver
Tail left deflection
Aileron right: down-deflection,
Aileron left: up-deflection
Ranging
Elevator up deflection
Climbing
Vertical teledetection for truth height finding

Fig. 5.2.7 *An instance of demonstration with annotated effects*

Other limits concern conceptual issues, such as processing capabilities of the NXT unit. In a real aircraft's system of control, the different tasks are concurrently executed, whereas our demonstrator is able to run three control-loops just sequentially on a single processor. An improved version of the demonstrator can be developed using, for instance, three NXT units.

5.3 The Planet

Author: Michele Moro

5.3.1 Teacher's guide

- Title: the Planet
- Introduction: Simulation of the motion of a planet (or another object) around the Sun based on the gravitational force; study of the conic curves.
- Goals:
  - To improve the knowledge of some basic physics concepts, such as space, speed, acceleration, time;
  - To study the theory of the planet motion (Kepler's and Newton's laws, conics);
  - To study some aspects of the analytical representation of conics;
  - To get some experience of what a simulation is and to what extension it can give a measurable representation of a real phenomenon;
  - To combine appropriately integer operation with loss of precision (division and square root) in an expression in order to minimize the total error.
- Age group: 16-19 years old.
- Rationale of the teaching approach: physics is better taught and learned when theory is presented together with some experimental activities. The notion of gravitational field is one of the most fascinating and difficult ones at the same time. The essence of the Kepler's and Newton's laws is not immediately intuitive and, therefore, the robotic simulation makes them more acceptable. The simulation, made with a relatively simple robot, is also the occasion to make some accessory reasoning about the interesting properties of conics and other geometrical aspects.

5.3.2 The problem

When Isaac Newton tried to determine what force justified the planetary motion in accordance with the Kepler's laws, he reached the conclusion that this force must be a mutual attraction, proportional to the product of the masses involved (the sun and the planet) and inversely proportional to the square of the distance from their centers of mass. In formula, the form of this force, the gravitational force (fig. 5.3.1), is given by:

$$F_G = G\ M\ m\ /\ r^2 \qquad (5.3.1)$$

with G the universal constant equal to:

$$G = 6.670\ 10^{-11}\ \mathrm{Nm^2kg^{-2}} \qquad (5.3.2)$$

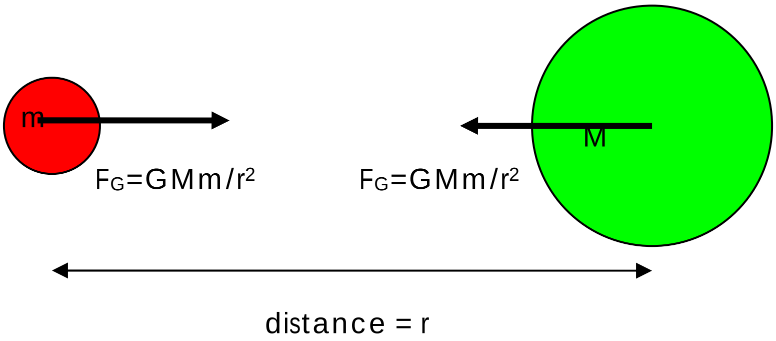


***Fig. 5.3.1** – The gravitational force*

Limiting the analysis to the motion of a single body around the sun, making the usual simplifying assumption that *M* (mass of the sun) >> *m* (mass of the body), it is possible to consider that the gravitational effect is limited to the only moving body, forced in its orbit by a centripetal acceleration in the form of:

$$a_G = F_G / m = G\ M / r^2 = \mu / r^2 \qquad (5.3.3)$$

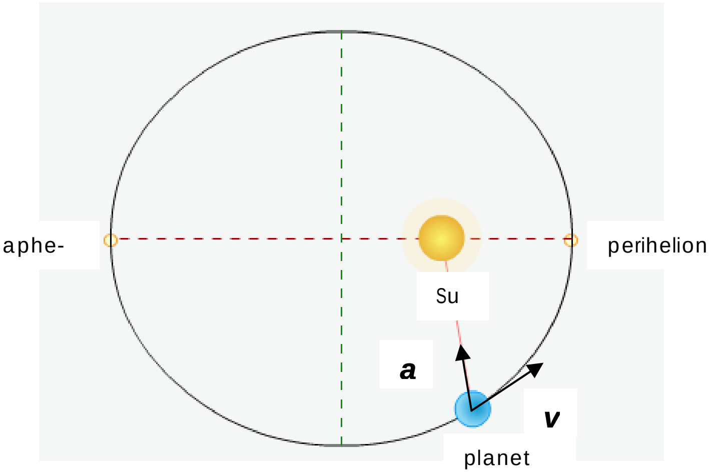


***Fig. 5.3.2** – The orbit of a body around the sun*

Under these conditions, the orbit is a conic which has the sun in one of its focus (fig. 5.4.2). The speed vector is always tangent to the orbit, oriented in the same direction of the motion, whereas the acceleration with modulus a_G is always directed from the body to the sun and, therefore, has an effect to accelerate (positive sign of the projection of the vector acceleration onto the speed vector) on the speed modulus, when the body is approaching and to decelerate (negative sign of the projection of the vector acceleration onto the speed vector), when the body is distancing. At the point closest to the sun (called *perihelion*), the acceleration is orthogonal to the speed and it is the greatest. In the case of the elliptical orbit, in the farthest point (the *aphelion*), the acceleration is still orthogonal but the speed is the smallest.

Unfortunately, the analytical study of a gravitational field based on the force of (5.3.1) is very hard and out of competence of a normal secondary level student. Therefore, a practical experience, which can give at least a qualitative evidence of the kind of orbit a body is forced to follow in a gravitational field, is of great interest.

Conics can be analytically defined as the locus of points *p*, for which the ratio between the distance of *p* from a point *F* (focus) and a line *D* (Directrix) is constant. This ratio is called *eccentricity* (*e*): when $0 \le e < 1$ the locus is an ellipse, with $e = 1$ a parabola, with $e > 1$ a hyperbola. In the cases of hyperbola and ellipse, these properties remain substantially the same for the second focus. Said *a* the semi-major axis of the ellipse (or the distance between the center of the focuses and the cusp of the hyperbola), *d* and *f* the distance between the center and respectively D and F, it also holds:

$$f = e/a \qquad e = f/a \qquad (5.3.4)$$

$$d = a/e \qquad (5.3.5)$$

Thus, for $e = 0$ the two focuses coincide, while the directrix is at infinity, and you get the special case of ellipse of the circumference.

The equation of an ellipse with the main focus placed on the origin and the other on its left on the x-axis is given by:

$$(x+f)^2/a^2 + y^2/b^2 = 1 \qquad (5.3.6)$$

where *a* and *f* are the semi-major axis and the focal distance already defined above and *b* the semi-minor axis. An alternative characterization is given in polar coordinates (fig. 5.3.3), placing the ellipses in the same position as in (5.4.6):

$$r = l / (1 + e \cos \theta) \qquad (5.3.7)$$

where *l* is the distance of the main focus from its vertical projection on the ellipse, *r* e *θ* are the polar coordinates of a generic point on the ellipse. For the perihelion and aphelion it holds:

$$r_{PER} = r_{MIN} = a - f = [\theta = 0] = l / (1+e) \qquad (5.3.8)$$

$$r_{APH} = r_{MAX} = a + f = [\theta = \pi] = l / (1-e) \qquad (5.3.9)$$

$$r_{PER} + r_{APH} = (l / (1+e)) + (l / (1-e)) = 2a \; a=l/1-e^2 l=a (1-e^2) \qquad (5.3.10)$$

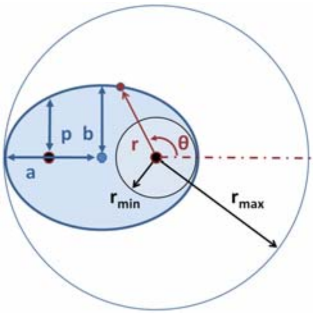


Fig. 5.3.3 – *Ellipse in polar coordinates*

Finally one can easily verify that:

b = a sqrt(1-e^2) $b^2 = a^2 – a^2e^2 = a^2 – f^2$ $b^2 + f^2 = a^2$ (5.3.11)

from which it derives that the distance of focus from the topmost point of the ellipse coincides with the semi-major axis.

Generally speaking, there is a known relation between speed and distance from the sun that, in the cases shown above, is given by:

Elliptical trajectory: v = sqrt(μ ((2/r)-(1/a))) (5.3.12)

Parabolic trajectory : v = sqrt(μ (2/r)) (5.3.13)

Hyperbolic trajectory: v = sqrt(μ ((2/r)+(1/a))) (5.3.14)

The speed of (5.3.13) is also known as the ‘escape speed’ (or ‘escape velocity’) because it is necessary that the initial speed of the body is greater than or equal to that of the escape speed for the body to escape the close orbit.

In the elliptical case, the maximum (at perihelion) and minimum (at aphelion) speeds are given by:

v_{PER} = sqrt(μ ((2/(a-f))-(1/a))) (5.3.15)

v_{APH} = sqrt(μ ((2/(a+f))-(1/a))) (5.3.16)

whereas for the period T we have:

T = (2π/sqrt(μ)) sqrt(a^3) $T^2 = (4\pi^2/ \mu)\ a^3$ (5.3.17)

in harmony with the Kepler’s third law.

5.3.3 Our simulation with NXT

In preparing the experience, one needs to be aware of the limitations, both physical and programming, of the NXT robot: inaccuracies in the control of motors and motion, integer arithmetic only (in the standard firmware, there is no floating point support), basic operations (lacking in particular of trigonometric functions). The results of this experience have a rather qualitative than quantitative value.

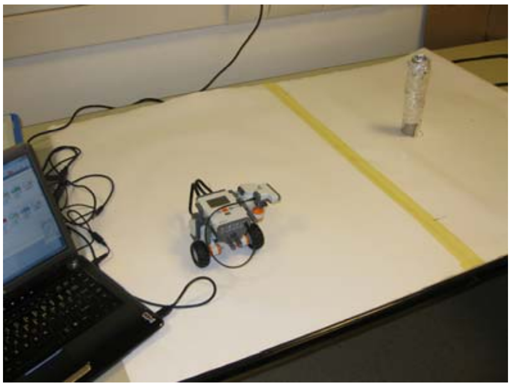


Fig. 5.3.4 – *The 'orbiting body'*

The robot, which is the orbiting body, is a *tribot* (fig. 5.3.4), with two independently driven wheels and a third one free, and equipped with a sonar head mounted on a third motor in order to adjust the angle of its 'vision' during the motion. The sun is represented by a fixed object 'visible' to the sonar so that the distance *r* is measured by this sensor.

The basic idea is to choose two reasonable values for μ and *a* in order to obtain practical values for the other parameters, specifically speed and acceleration. You have to choose also an appropriate (small) time interval Δt on which the calculation of each motion step will be based. During the simulation, acceleration and speed will always be calculated using formulas (5.3.3) and (5.3.12) (assuming you have installed the square root block in your NXT-G environment).

To simplify the simulation, we also assume that, initially, the robot is put on the aphelion, at a distance of $a+f$ from the focus, so that its axis can be orthogonal as to the major axis of the expected elliptical trajectory. The simulation does not maintain any information about the position and the orientation of the robot: we always assume that, at any motion step, the starting position and orientation are correct as the cumulative effect of the previous steps.

In the simulation, a step at time *t* is formed by small straight-line motion, followed by a rotation around the robot's axis middle point. The first motion corresponds to the contribution of the tangential speed vector, and, therefore, it is calculated as $v*\Delta t$; the rotation corresponds to the contribution of the acceleration vector that rotates the speed vector: fig. 5.3.5 shows that the vector composition of speed *v* at time *t* and its variation given by the vector $\Delta v = a_G * \Delta t$.

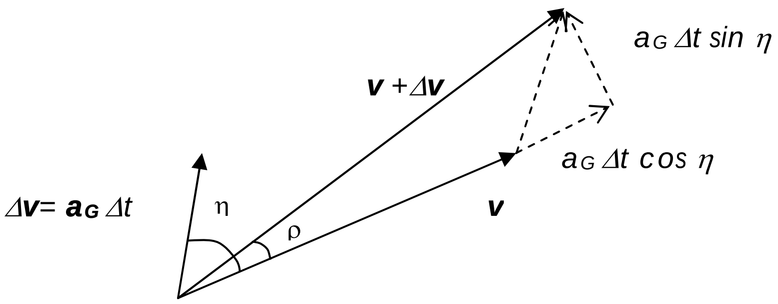

$a_G\,\Delta t \sin \eta$
$\boldsymbol{v} + \Delta\boldsymbol{v}$
$a_G\,\Delta t \cos \eta$
$\Delta\boldsymbol{v} = \boldsymbol{a_G}\,\Delta t$
η
ρ
$\boldsymbol{v}$

***Fig. 5.3.5** – Vector composition*

To ensure that the orientation of the robot at the instant $t+\Delta t$ is the vector $v+\Delta v$, as shown by the figure, we need to rotate the robot of an angle ρ equal to:

$$\rho = \arctan((a_G\,\Delta t \sin \eta) / (v + a_G\,\Delta t \cos \eta)) \qquad (5.3.18)$$

where η is the angle between vectors v and a_G.

A first rough approximation is to consider the acceleration vector always substantially orthogonal to the speed vector ($\eta = \pi/2$). With this simplification it holds:

$$\rho_{APPROX} = \arctan(a_G\,\Delta t / v) \approx a_G\,\Delta t / v \qquad (5.3.19)$$

Further (5.3.19) approximation of considering the angle so small as to have the arctan value and the angle (in radiants) coincident has been applied. This produces a first approximated calculation of the rotation to be applied to the robot very simple.

Plotting the theoretical ellipse and that of the simulation, obtained by applying the approximate method described above, by using the following values of characteristic parameters (lengths scaled in centimeters)

$$a=40 \qquad f=20 \qquad e=0.5 \qquad b = a\,\mathrm{sqrt}(1-e^2) = 20\sqrt{3}$$

$$\mu=6000\,\Delta t=0.5 \qquad (5.3.20)$$

and starting, as already assumed, with the robot placed on the hypothetical aphelion and oriented parallel to the minor axis of the ellipse, we obtained the plotting values of fig. 5.3.6. As you can see the approximation is not good after the first quadrant. Thus, we decided to add a correction to this first approximation trying to maintain the calculation simple.

From an analysis of the variation of the angle η, while the point moves along the bottom semi-ellipse, you observe that, both at aphelion ($r_{AF} = a+f$) and at perihelion ($r_{PER} = a-f$) $\eta=\pi/2$, but along the path, the angle decreases until it reaches a minimum, which depends on the value of e, when $r = a$. With the data given above, the minimum is equal to $\eta_{MIN} = \pi/3$ with $\sin(\eta_{MIN}) = \sqrt{}\ 3/2 \approx 0.87$, $\cos(\eta_{MIN}) = 0.5$, while $\eta_{MAX} = \pi/2$ with $\sin(\eta_{MAX}) = 1$ and $\cos(\eta_{MAX}) = 0$. Therefore, we decided to apply a correction to the angle in the range where sin(η) is more different from 1, which has been empirically determined as *a*-*f*/2 ≤ *r* ≤ *a*+*f*/2: in this interval, the rotation to be applied is calculated as:

$$\rho_{\text{APPROX}} \approx ((a_G\ \Delta t \sin \eta_m) / (v + a_G\ \Delta t \cos \eta_m)) = ((a_G\ \Delta t\ 0.9) / (v + a_G\ \Delta t\ 0.5) \qquad (5.3.21)$$

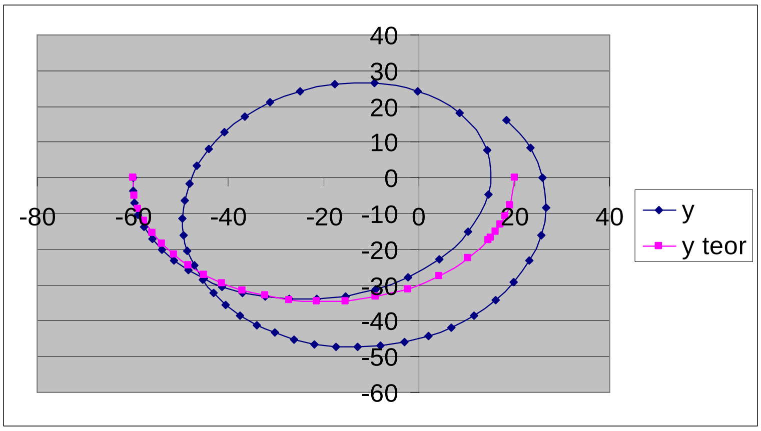


Fig. 5.3.6 *– A first approximation*

With this correction, the simulated evolution is plotted in Fig. 5.3.7 and the improvement is evident.

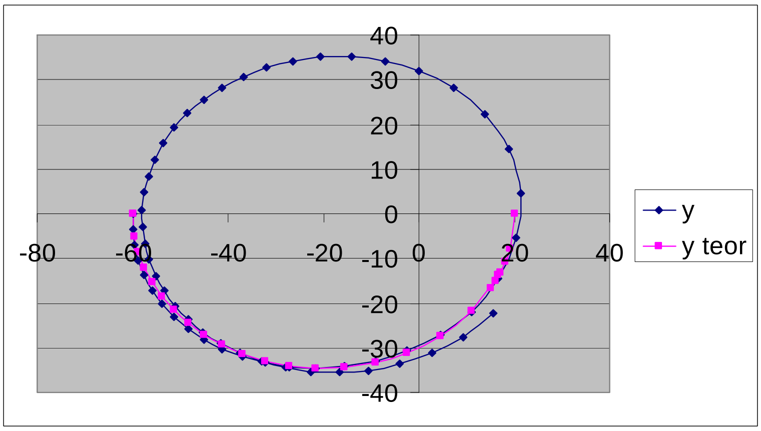


Fig. 5.3.7 *– A second approximation*

Another straightforward correction is necessary in order to compensate the measure of the distance *r* performed by the sonar: in fact, the measure is less than the actual distance of the focus from the rotation point of the robot, which we consider as the application point of the speed and acceleration vectors, both because the object used as the attracting body has a non-null radius and because the sonar is at a certain distance from the axis of the robot, where the rotation point lies. Thus, to obtain the correct value of *r*, the measure must be incremented of the sum of these two distances (the sun's radius and the offset of the sonar).

In the practical realization with NXT, we must also take into account the limitations of the integer calculation: besides delaying as much as possible all the divisions in just one division as the final operation, in our case we must also deal with the imprecision introduced by the square root. It results more precise to execute the integer square root as the very last operation. But you must also take into account the relative error of the square root in cases of small and large numbers. This could suggest to move inside the square root external multiplicative factors (elevating them to their square), whereas divisional factors should be moved inside the square root only if the inner division maintains a quotient not too small (in order to maintain small the error of the integer square root). In doing these passages, you must not generate overflow of the 32 bit integer capacity ($+2^{31} \approx 2\ 10^9$).

Therefore, assuming that

$$\Delta t = I / j \qquad R_w = p / q \qquad D_w = w / z \qquad (5.3.22)$$

with R_w the radius of the wheels, D_w the distance between the wheels and *i*, *j*, *p*, *q*, *w*, *z* integer values, for the straight-line motion step it follows:

$$v \cdot \Delta t = \theta_w\ R_w = \theta_{wd}\ (2\pi/360) \cdot R_w \qquad (5.3.23)$$

where θ_w and θ_{wd} are the angles which the wheels must rotate of, respectively measured in radiants and degrees. From (5.3.23) you obtain the angle to be set as a parameter of the motor control block, expressed in degrees:

$$\theta_{wd} = 360 \cdot v \cdot \Delta t / (2\pi \cdot R_w) \qquad (5.3.24)$$

To perform this calculation accurately, we now apply the recommendations suggested above:

$$\begin{aligned}\theta_{wd} &= 360 \cdot \mathrm{sqrt}(\mu((2/r)-(1/a))) \cdot (i/j) / (2(314/100) \cdot (p/q)) = \\ &= (\ (9000 \cdot i \cdot q) / (157 \cdot p \cdot j)\ )\ \mathrm{sqrt}((2 \cdot \mu \cdot a - \mu \cdot r) / (a \cdot r)) \qquad (5.3.25)\end{aligned}$$

Assuming that the first fraction (the terms are all constant) is reduced to the lowest terms as *pp / qq* , we could finally obtain:

$$(9000 \cdot i \cdot q) / (157 \cdot p \cdot j) = pp/qq \qquad \theta_{wd} = \mathrm{sqrt}(pp^2 \cdot (2 \cdot \mu \cdot a - \mu \cdot r) / (qq^2 \cdot a \cdot r)) \qquad (5.3.26)$$

The convenience to move pp/qq under square root depends on their size. In the case of the values given by (5.3.20) and with standard wheels with a diameter of 56 mm, it holds (linear measures in centimetres):

$$\Delta t = 1/2 \qquad R_w = 28/10 \qquad (9000 \cdot 1 \cdot 10) / (157 \cdot 2 \cdot 28) = 11250 / 1099 \qquad (5.3.27)$$

pp and qq are too big to be moved under square root. Let us try to find a simpler approximation:

$$11250 / 1099 \approx 10.2365 \approx 1024/100 = 256/25 \qquad (5.3.28)$$

Say isqrt and idiv respectively our available integer version of the square root and division, we obtain:

$$\begin{aligned}\theta_{wd} &= \text{isqrt}(\ \text{idiv}(pp^2\cdot(2\cdot\mu\cdot a - \mu\cdot r),\ (qq^2\cdot a\cdot r))\ ) = \\ &= \text{isqrt}(\ \text{idiv}(256^2\cdot(2\cdot 6000\cdot 40 - 6000\cdot r),\ (25^2\cdot 40\cdot r))\ ) = \\ &= \text{isqrt}(\ \text{idiv}(65536\cdot(480 - 6\cdot r),\ (25\cdot r))\ ) = \qquad (5.3.29)\end{aligned}$$

The minimum value of θ_{wd} is reached at the aphelion when $r = a+f = 60$:

$$\begin{aligned}\theta_{wdMIN} &= \text{isqrt}(\ \text{idiv}(65536\cdot(480 - 6\cdot 60),\ (25\cdot 60))\ ) = \\ &= \text{isqrt}(\ \text{idiv}(\ 7864320,\ 1500)\ ) = \text{isqrt}(5242) = 72° \qquad (5.3.30)\end{aligned}$$

corresponding to approximately 3.52 cm. The result is good because the precise value is 72.34. The maximum appears at the perihelion when r = a-f = 20:

$$\begin{aligned}\theta_{wdMAX} &= \text{isqrt}(\ \text{idiv}(65536\cdot(480 - 6\cdot 20),\ (25\cdot 20))\ ) = \\ &= \text{isqrt}(\ \text{idiv}(23592960,\ 500)\ ) = \text{isqrt}(47185) = 217° \qquad (5.3.31)\end{aligned}$$

Even here the result, which corresponds to a move of about 10.6 cm, is good because its precise value is 217.04.

Now, we consider the elementary rotation of the robot: to turn the robot of an angle ρ, we need to set to its maximum the steering parameter of a *move* block (100% with the correct direction) and make the motors to rotate of an angle equal to:

$$\theta_{wd} = 360\cdot D_w\ \rho\ /(2\pi\cdot 2\cdot R_w) = (90\cdot D_w\ /(\pi\cdot R_w))\ \rho \qquad (5.3.32)$$

The approximated values of ρ are given by (5.3.19) and (5.3.21), respectively for each one of the two identified parts of the orbit. Using the first formula:

$$\begin{aligned}\rho &= ((\mu/r^2)\cdot\Delta t) / (\text{sqrt}(\mu\cdot((2/r)-(1/a)))) = \text{sqrt}(\mu\cdot a\cdot\Delta t^2 / (r^3\cdot(2a-r))) = \\ &= \text{sqrt}(\mu\cdot a\cdot i^2 / (r^3\cdot(2a-r)\cdot j^2)) \qquad (5.3.33)\end{aligned}$$

$$\begin{aligned}\theta_{wd} &= (90\cdot(w/z)\ /((314/100)\cdot(p/q)))\ \text{sqrt}(\mu\cdot a\cdot i^2 / (r^3\cdot(2a-r)\cdot j^2)) = \\ &= (4500\cdot w\cdot q / (157\cdot z\cdot p))\ \text{sqrt}(\mu\cdot a\cdot i^2 / (r^3\cdot(2a-r)\cdot j^2)) \qquad (5.3.34)\end{aligned}$$

Using the already used data and with $D_w = w/z = 10$, we obtain:

$$\begin{aligned}\theta_{wd} &= (450000 / (157\cdot 28))\ \text{sqrt}(6000\cdot 40 / (r^3\cdot(80-r)\cdot 4)) = \\ &= (112500 / 1099)\ \text{sqrt}(60000 / (r^3\cdot(80-r))) \qquad (5.3.35)\end{aligned}$$

A 100 factor can be moved under square root, and we obtain:

$$\theta_{wd} = (1125 / (1099\cdot r))\ \text{sqrt}(600000000 / (r\cdot(80-r))) \qquad (5.3.36)$$

For the acceleration increases faster than the speed when *r* varies, the maximum angle occurs when the acceleration is at its maximum and, thus, having the minimum *r* (=20):

$$\begin{aligned}\theta_{wdMAX} = {} & \text{idiv}(\ (1125\cdot\text{isqrt}(\ \text{idiv}(600000000,\ (20\cdot(80\text{-}20))\ ))\ ),\\ & (1099\cdot 20)) = \\ & \text{idiv}(\ (1125\cdot\text{isqrt}(\ \text{idiv}(600000000,\ 1200))\ ),\\ & (21980)) = \\ & \text{idiv}(\ (1125\cdot\text{isqrt}(500000)\ ),\ 21980) = \\ & \text{idiv}((1125\cdot 707),\ 21980) = \text{idiv}(795375,\ 21980) = 36° \quad (5.3.37)\end{aligned}$$

The result is good because the correct value is 36.17.

Now considering the second approximation of the formula (5.4.21), in the range of interest 30≤*r*≤50, it approximately follows 10≤*v*≤15, 1.25≤$a_G\cdot\Delta t$≤3.2. With these values it is advisable to scale up of 100 in order to make some decimals significative for the integer calculations:

$$\rho\ = ((a_G\ \Delta t\ 90) / (100\cdot v + a_G\ \Delta t\ 50) \qquad (5.3.38)$$

$$\begin{aligned}\theta_{wd} &= (90\cdot(w/z)\ /((314/100)\cdot(p/q)))\cdot(a_G\cdot(i/j)\cdot 90\ /\ (100\cdot v + a_G\cdot(i/j)\cdot 50)) = \\ &= (8100\cdot 10\cdot 6000\cdot 1000)\ /\ (314\cdot 28\cdot 2\cdot r^2\cdot(\text{sqrt}(1500000\cdot(80\text{-}r)/r) + (6000\cdot 25/r^2)\ )) = \\ &= 27638762\ /\ (r\cdot 10\cdot\text{sqrt}(15000\cdot r\cdot(80\text{-}r)) + 6000\cdot 25) \qquad (5.3.39)\end{aligned}$$

With an intermediate value (*r*=*a*=40) and the usual truncations you obtain 13 degrees versus a precise 12.22.

What about the power to be applied to the motors during the two types of motions? As known, in the absence of excessive load, the power control is actually a speed control. Considering that the speed of the second motion, the rotation, is not so important to have the feeling of the simulated speed, which is originally continuous, in every straight-line motion step we would impose a speed able to make the robot move exactly in the step time Δt of the simulation. The angular speed to be set follows from (5.3.24):

$$\omega_{wd} = \theta_{wd}\ /\ \Delta t = 360\cdot v\cdot/\ (2\pi\cdot R_w)\ \text{degrees/s} \qquad (5.3.40)$$

with Δt = 0.5, $\omega_{wd} = 2\cdot\theta_{wd}$. For with the given experimental data we have estimated 72≤θ_{wd} ≤217, it would follow 144≤ω_{wd} ≤434. Assuming true the already estimated relation:

$$\text{P(ower)} = (1/8.15)\cdot\omega \qquad (5.3.41)$$

we obtain 17.66≤*P*≤53.25. Thus, if this range of powers is problematic, we could scale up or down the power of a given factor to maintain the same feeling of the motion, especially the increase of speed from aphelion to perihelion.

Finally, as regards the position of the sonar, we decided not to estimate step by step the viewing angle of the sonar in respect of the mutual orbiting body-sun position, because we didn't want to maintain a state variable describing the position of the robot. Instead, a more robust solution is to initially put the sonar with its axis orthogonal to the robot, and at the beginning of every step make the sonar sweep an angle of sufficient amplitude in the range of $\pm\alpha$, performing a certain number of readings and taking the minimum as a measure of the distance.

5.3.4 The program

We are describing the core of the program, which is an infinite loop every execution of which corresponding to a single step. One step is formed by three stages: the measurement of the distance, the straight-line motion, the rotation.

The first stage is represented by the code of fig. 5.3.8. Assuming that the sonar is mounted on a motor connected to port B, the scanning of the head is made as follows: an initial rotation of 60 degrees in one direction; 5 steps with one reading at the beginning of every step, an updating of the variable d to represent the minimum distance measured, a rotation of 30 degrees in the opposite direction at the end of the step. After this evaluation, the head is repositioned to its original direction with a final rotation of 90 degrees.

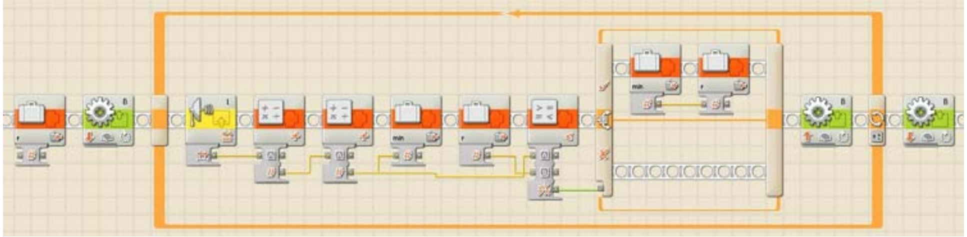


Fig. 5.3.8 – *Part I: measuring the distance*

```
VarDecl(Name=delta, Type=NUM) -- the angle to be performed

VarDecl(Name=min, Type=NUM) -- the minimum distance

VarDecl(Name=r, Type=NUM) -- the distance for the calculations

Var(Name=r.NUM, Act=WR, Val=255)  -- d=initial minimum

Motor(Port=B, Dir=BK, Act=CONST, Pwr=20, PwrCtrl=ON,
  Dur=60.DEG, Wait=ON, Next=BRK)

Loop1: Loop(Ctrl=FOREVER, Dis=OFF) [

          Loop2: Loop(Ctrl=COUNT, Until=5, ShowCnt=OFF) [
    So: SonarSens(Port=1, Cmp=??, Show=CM)
    Ad1:MathOp(Type=ADD, A=So.Dist, B=<SonarOffset>)
    Ad2:MathOp(Type=ADD, A=Ad1.Res, B=<SunRadius>)
    Var(Name=min.NUM, Act=WR, Val=Ad2.Res)
    Vd1: Var(Name=r.NUM, Act=RD)
    Cm1: CmpOp(Type=LT, A=Ad2.Res, B=Vd1.Val)
    Sw1: Switch(Ctrl=VAL, Type=LOGIC, Dis=ON,
      CondUp=TRUE, Val=Cm1.Res)
    [Sw1.IF
            Vm1: Var(Name=min.NUM, Act=RD)
            Var(Name=r.NUM, Act=WR, Val=Vm1.Val)
    Sw1.IF]
    [Sw1.ELSE
    Sw1.ELSE]
    Motor(Port=B, Dir=FD, Act=CONST, Pwr=20,
      PwrCtrl=ON, Dur=30.DEG, Wait=ON, Next=BRK)
Loop2]
Motor(Port=B, Dir=BK, Act=CONST, Pwr=20, PwrCtrl=ON,
  Dur=90.DEG, Wait=ON, Next=BRK)
```

The second stage (see fig. 5.3.9) is the calculation of the angle to be performed for the straight motion based on the (5.3.29) formula. This piece of code is rather straightforward.

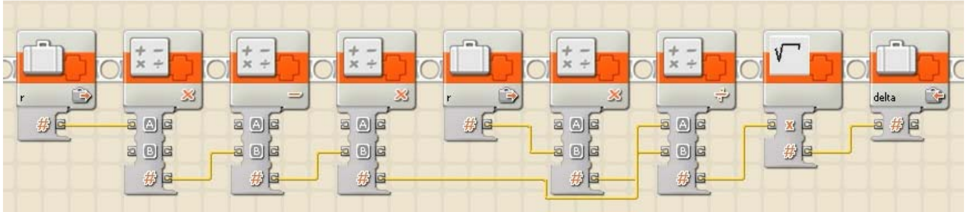


Fig. 5.3.9 – *Part II: calculating the angle for straight motion*

```
Vd2:Var(Name=r.NUM, Act=RD)
Mu1:MathOp(Type=MUL, A=Vd2.Val, B=6)
Su1:MathOp(Type=SUB, A=480, B=Mu1.Res)
```

```
Mu2:MathOp(Type=MUL, A=65536, B=Su1.Res)
Vd3:Var(Name=r.NUM, Act=RD)
Mu3:MathOp(Type=MUL, A=25, B=Vd3.Val)
Di1:MathOp(Type=DIV, A=Mu2, B=Mu3.Res)
Sq1:Sqrt(x1=Di1.Res)
Var(Name=delta.NUM, Act=WR, Val=Sq1.Res)
```

In order to give the impression of the variation of speed during the orbit, we apply a varying motor power in the range of 30÷60 (30 when the angle is minimum, i.e. 72°, 60 when it is maximum, i.e. 217°), linearly scaling the angle value (fig. 5.3.10):

$30 = \alpha \cdot 72+\beta \qquad 60=\alpha \cdot 217+\beta \quad \Rightarrow$
(subtracting the first from the second) $30=\alpha \cdot 145 \quad \Rightarrow \quad \alpha=30/145$
$\beta= 30 - 30 \cdot 72/145$
$\text{pot} = (30/145) \cdot \theta + 30 - 30 \cdot 72/145 = (4350 + 30 \cdot (\theta-72)) / 145 \qquad (5.4.42)$

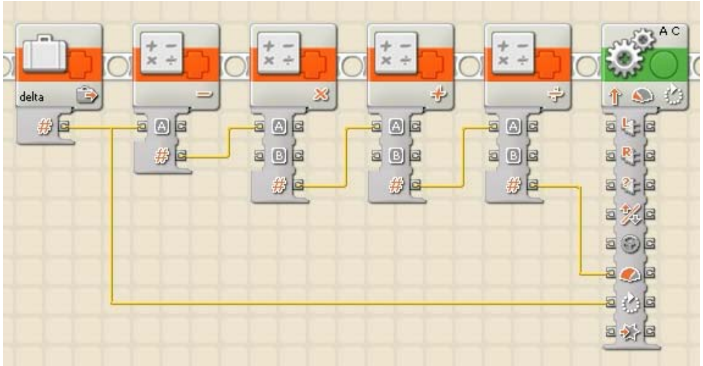


Fig. 5.3.10 – *Part III: simulating the variation of speed*

```
Vd4:Var(Name=delta.NUM, Act=RD)
Su2:MathOp(Type=SUB, A=Vd4.Val, B=72)
Mu4:MathOp(Type=MUL, A=Su2.Res, B=30)
Ad3:MathOp(Type=ADD, A=Mu4.Res, B=4350)
Di2:MathOp(Type=DIV, A=Ad3.Res, B=145)
```

After the straight motion, a short rotation is applied. We must distinguish the section in which we apply the simpler approximation of (5.3.19) from the section where we apply the more complex one (5.3.21). The choice is based on the value of the measured distance from the focus: the simpler approximation is the case when such distance is out of the range of 30÷50 (fig. 5.3.11).

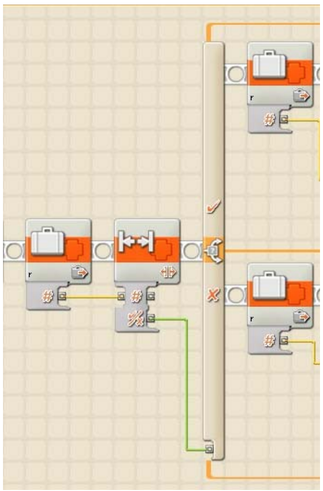


Fig. 5.3.11 – *Part IV: to distinguish the two approximations*

```
Vd5: Var(Name=r.NUM, Act=RD)
Ra1:Range(Type=OUT, A=30, B=50, Val=Vd5.Val)
```

Now, for the ‘then part’ we must apply (5.3.36) (fig. 5.3.12):

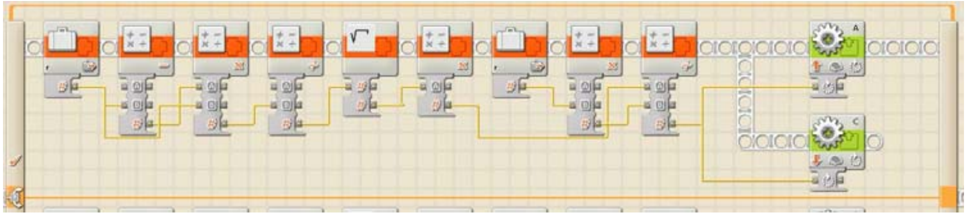


Fig. 5.3.12 – Part V: simpler approximation

```
Sw2: Switch(Ctrl=VAL, Type=LOGIC, Dis=ON,
  CondUp=TRUE, Val=Ra1.Res)
[Sw2.IF
        Vd6: Var(Name=r.NUM, Act=RD)
        Su4:MathOp(Type=SUB, A=80, B=Vd6.Val)
        Mu5:MathOp(Type=MUL, A=Su4.Res, B=Vd6.Val)
        Di3:MathOp(Type=DIV, A=600000000, B=Mu5.Res)
        Sq2:Sqrt(x1=Di3.Res)
        Mu6:MathOp(Type=MUL, A=Sq2.Res, B=1125)
        Vd7: Var(Name=r.NUM, Act=RD)
        Mu7:MathOp(Type=MUL, A=2156, B=Vd7.Val)
        Di4:MathOp(Type=DIV, A=Di3.Res, B=Mu7.Res)
        {1
                Motor(Port=A, Dir=FD, Act=CONST, Pwr=15,
                  PwrCtrl=ON, Dur=Di4.Res.DEG, Wait=ON,
                  Next=BRK)
        1}
        {2
                Motor(Port=C, Dir=BK, Act=CONST, Pwr=15,
                  PwrCtrl=ON, Dur=Di4.Res.DEG, Wait=ON,
                  Next=BRK)
        2}
Sw2.IF]
```

The more complex approximation follows (fig. 5.3.13).

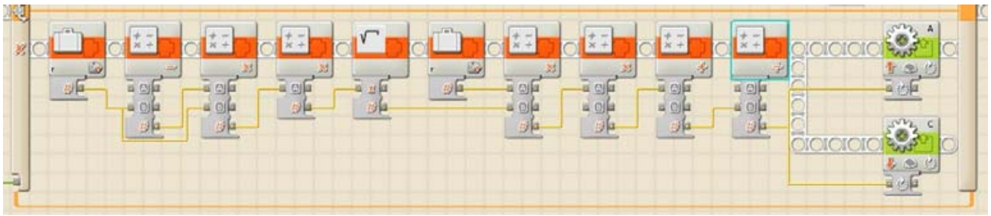


Fig. 5.3.13 – Part VI: more complex approximation

```
[Sw2.ELSE
        Vd7: Var(Name=r.NUM, Act=RD)
        Su5:MathOp(Type=SUB, A=80, B=Vd7.Val)
        Mu6:MathOp(Type=MUL, A=Su5.Res, B=Vd7.Val)
        Mu7:MathOp(Type=MUL, A=Mu6.Res, B=15000)
        Sq3:Sqrt(x1=Mu7.Res)
        Vd8: Var(Name=r.NUM, Act=RD)
        Mu8:MathOp(Type=MUL, A=Vd8.Val, B=Sq3.Res)
        Mu9:MathOp(Type=MUL, A=Mu8.Res, B=10)

        Ad4:MathOp(Type=ADD, A=Mu9.Res, B=150000)

        Di5:MathOp(Type=DIV, A=27638762, B=Ad4.Res)

        {1

            Motor(Port=A, Dir=FD, Act=CONST, Pwr=15,
              PwrCtrl=ON, Dur=Di5.Res.DEG, Wait=ON,
              Next=BRK)

        1}

        {2

            Motor(Port=C, Dir=BK, Act=CONST, Pwr=15,
              PwrCtrl=ON, Dur=Di5.Res.DEG, Wait=ON,
              Next=BRK)

        2}

    Sw2.ELSE]

Loop1]
```

5.4 The Lunar Lander

Author: Michele Moro

5.4.1 Teacher's guide

- Title: the Lunar Lander
- Introduction: Simulation of the landing of a lunar module, assumed to compensate partly the lunar gravitation with a reaction engine.
- Goals:
    - To improve the knowledge of some basic physical items like space, speed, acceleration, time
    - To study the theory of basic motions, like the uniformly accelerated (decelerated) motion;
    - To solve first grade equations and systems;
    - To afford the difficulties of the landing problem with a simple and robust sensor-based robotic solution.
- Age group: 16-19 years old.
- Rationale of the teaching approach: physics is better taught and learned when theory is presented together with some experimental activities. There are several well known approaches to present basic items like space, time etc., but some suffer of the lack of an easy reproducibility in normal conditions or have no clear relation to normal life experiences. The lunar landing problem can be easily presented and understood in its essence even though its solution is related to a theory with evident difficulties. The presented robotic simulation can help the students in understanding such a solution and can make them more comfortable with other more general similar theories.

5.4.2 The problem

From the Kepler and Newton laws we know that the motion of an object subjected to the gravitational force of a planet or a satellite is in general a conic, but as a special case of a body that has a null initial speed or whose initial velocity vector points towards the center of the mass of the planet, the motion tends to the degenerate case of a motion following a line, and the body sooner or later will fall on the surface of the planet. If a lunar module should 'land' on the surface of our satellite coming from the Earth, to prevent the destructive impact, it is necessary to impress, through a jet engine, a force in the direction opposite to the attraction, that is, tending to push it out of the Moon, that not only compensates for the acceleration, but makes the objects reduce their speed near the surface to very small values, essentially under a negligible threshold (fig. 5.4.1).

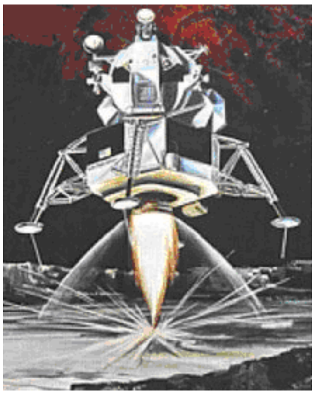


Fig. 5.4.1 – *A lunar module*

To give an idea of the speed you can reach close to the surface of a planet or a satellite, we assume that a mass m is ‘dropped’ with speed zero 1 km away from the lunar surface, and that there are no other forces different from the lunar gravitation (no friction, no influence of other bodies). The acceleration, which in this case is continuously parallel to the motion and oriented towards the moon, has the effect of increasing the speed, and its modulus is equal to:

$$a_G = F_G / m = G M / r^2 = \mu / r^2 \qquad (5.4.1)$$

where G is the universal gravitational constant, M the mass of the Moon, and r the distance from the center of the planet/satellite, which serves as a center of mass. In the case of the Moon we have:

$$\mu_{MOON} = G M_{MOON} = 6.670\ 10^{-11} \cdot 7.34\ 10^{22} = 4.895\ 10^{12}\ Nm^2kg^{-1} \qquad (5.4.2)$$

With the data of the problem, the initial acceleration is:

$$a_{G0} = \mu_{MOON} / ((R_{MOON} + 10^3)^2) \qquad (5.4.3)$$

Using the average radius of the Moon, corresponding to 1.74 10^6 m, as the distance of the surface from the center of mass, we see that in the considered range of distances, the acceleration increases slightly approaching to the lunar surface. In fact, at the beginning and end of motion it holds:

$$a_{G0} = 4.895\ 10^{12} / ((1.74\ 10^6 + 10^3)^2) = 1.6149$$
$$a_{Gf} = 4.895\ 10^{12} / 1.74\ 10^{12} = 1.6167 \qquad (5.4.4)$$

Because for an elliptical orbit it holds:

$$v = sqrt(\mu\ ((2/r)-(1/a))\ ) \qquad (5.4.5)$$

giving the speed value when the distance varies in the elliptical (closed) motion, we obtain that in order to justify the initial null speed it must hold:

$$(2/r_0)=(1/a) \qquad a = r_0/2 \qquad (5.4.6)$$

that is, it is actually a degenerate motion in which we can consider the eccentricity of the ellipse tending to 1, with the body leaving at null speed from the secondary focus and moving to reach the attractive focus. With our data, (5.4.5) and (5.4.6), we give the speed at the impact on the surface:

$$v = \text{sqrt}(\mu_{MOON} ((2/R_{MOON})-(2/(R_{MOON} +10^3)))) = 56.84 \text{ ms}^{-1} = 204.65 \text{ km h}^{-1} \qquad (5.4.7)$$

a considerable speed, certainly destructive.

5.4.3 The theory

Let us assume that in our simulation we can easily control the speed of the robot: this is true because the settable power of the motors of the NXT is actually an angular speed control, at least until such control can compensate for a possible resistant torque. So, wanting to provide an analytical solution of the problem, a first approach is to set ‘a priori’ to a specific speed time profile (v = v (t)) that has the desired characteristics, in particular a desired initial speed and a final speed tending to zero.

The profiles of this kind are numerous: we will conduct our analysis in a simple case, a uniformly decelerated motion (fig. 5.4.2).

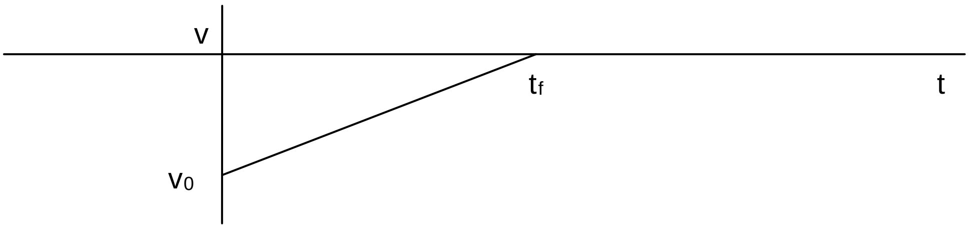


***Fig. 5.4.2 –** Uniformly decelerated motion*

In this case it holds:

$$v(t) = v_0 - (v_0 / t_f)\cdot t \qquad (5.4.8)$$

v_0 is negative and acceleration is constant (positive) and equal to $-v_0/t_f$, and $v(t_f)=0$. But how much is t_f, at the moment when the motion is completed and the body is on the surface? This parameter depends on the spatial motion and it is the moment where $r(t_f) = R$ (radius of the planet/satellite). This motion is given by:

$$r(t) = r_0 + {}_0\int^t v(t)\cdot dt \qquad (5.4.9)$$

A primitive function of (5.4.8) is:

$$\rho(t) = v_0\cdot t - (v_0 / (2\cdot t_f))\cdot t^2 \qquad (5.4.10)$$

so from (5.4.9) we obtain:

$$r(t_f) = R = r_0 + [v_0\cdot t - (v_0 / (2\cdot t_f))\cdot t^2]_0^{tf} \quad = r_0 + v_0\cdot t_f - (v_0 / 2)\cdot t_f \qquad (5.4.11)$$

$$t_f = 2\cdot(R-r_0) / v_0 = -2\cdot(r_0-R) / v_0 \qquad (5.4.12)$$

(remember that v_0 is negative). On the numerator there is the initial distance from the surface. With this value of the t_f the (5.4.8) and (5.4.9) become:

$$v(t) = v_0 \; + (v_0^2 / (2(r_0-R)))\cdot t \qquad (5.4.13)$$

$$r(t) = r_0 + v_0\cdot t + (v_0^2 / (4\cdot(r_0-R)))\cdot t^2 \qquad (5.4.14)$$

If, for example, we complete the approaching phase in 1 minute, starting at 1 km far from the surface, it should be:

$$v_0 = -2\cdot(r_0-R) / t_f = -2\cdot10^3/ 60 = -100/3 \text{ ms}^{-1} = -120 \text{ kmh}^{-1} \qquad (5.4.15)$$

from which it derives as a (positive) deceleration:

$$a = v_0 / t_f = 100 / (3\cdot60) = 0.555 \text{ ms}^{-2} \qquad (5.4.16)$$

a very limited value, when compared with the terrestrial acceleration of gravity (9.78).

Assuming that the students have the knowledge to derive the mathematical description of the motion based on the chosen speed profile, the approach presented here can in principle be adopted by the simulation using the NXT robot, but the accuracy in the implementation of the speed profile is essentially tied on maintaining the accuracy of the time going by, which maybe somewhat problematic for the NXT. Moreover, no sensorial capability of the robot is exploited that could make it possible to realize a 'robust' control that adapts itself to the inevitable inaccuracies.

The NXT kit includes a sensor of distance that gives us the opportunity to adopt an alternative approach, where the user defines a speed/space profile ($v = v(s)$) instead of a speed/time one. Now we conduct the analysis again on a seemingly simple case, a linear profile (fig. 5.4.3)

The profile is reasonable because in the initial position (r_0) the initial velocity is v_0 and when the body reaches the surface, the speed is zero. Now how can we derive the temporal functions of space, velocity and acceleration?

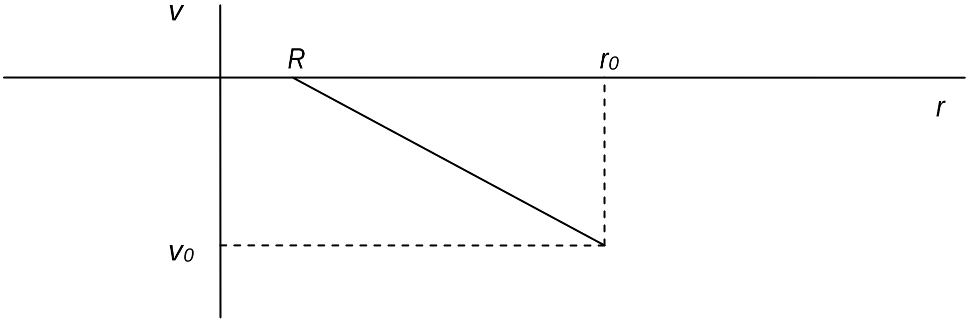


***Fig. 5.4.3** – Linear speed/space profile*

For the given profile it holds:

$$v(r) = (v_0 / (r_0\text{-}R))\cdot(r\text{-}R) = dr/dt \qquad (5.4.17)$$

(5.4.17) is a first order differential equation, a mathematical relation usually affordable at the university level, and usually too harder for a student of secondary education (though some secondary schools present it during the last year). Fortunately, as we shall see in the next section, making the robot to follow a speed profile does not require to solve analytically such a relation (or a similar one), but for the sake of completeness we recall that such kind of equation as a general solution the family of exponential functions. The solution, applying the constraints of our case, is:

$$r(t) = R + (r_0\text{-}R)\ e^{(v0\ /\ (r0\text{-}R))\cdot t\cdot} \qquad (5.4.18)$$

The function in (5.4.18) tends to *R* when *t*→∞: therefore, the motion is absolutely a not uniformly decelerated motion, as a superficial observer could believe in, and in theory the surface is really reached only asymptotically. This is not a problem for the NXT because the motor has a minimum speed to be taken into account, as well as the precision of the distance sensor. Calculating the derivative to obtain the the temporal profile of velocity, we obtain:

$$v(t) = (r_0\text{-}R)\ (v_0 / (r_0\text{-}R))\ e^{(v0\ /\ (r0\text{-}R))\cdot t\cdot} \qquad (5.4.19)$$

Thus, even the speed decreases exponentially.

Also, in theory, we try now to calculate what speed/space profile would produce a uniformly decelerated motion, through a reasoning inverse of that made above. From (5.4.8), assuming $a_0 = -v_0/t_f = v_0^2 / (2\ (R\text{-}r_0))$, we obtain:

$$t(v) = (v\text{-}v_0) / a_0 \qquad (5.4.20)$$

Substituting now in (5.4.14) it holds:

$$r(v) = r(t(v)) = r_0 + v_0\cdot(v\text{-}v_0) / a_0 + (a_0 / 2)\ ((v\text{-}v_0)/a_0)^2 \qquad (5.4.21)$$

$r(v) = r_0 + (v_0 \cdot / a_0) \cdot v - v_0^2 \cdot / a_0 + (1 / (2 \cdot a_0)) (v^2 - 2v_0 v + v_0^2)$ (5.4.22)

Moving *r* to the second member and multiplying by *$2a_0$* we obtain the equation of second degree:

$$v^2 - v_0^2 + 2a_0r_0 - 2a_0r = 0 \qquad (5.4.23)$$

Once solved, getting the negative solution, as the speed must be, the (5.5.23) gives the following profile (always taking into account that v_0 is negative):

$$v(r) = -\mathrm{sqrt}(2a_0r + v_0^2 - 2a_0r_0) = -\mathrm{sqrt}((v_0^2 / (r_0-R)) \cdot r + v_0^2 - v_0^2r_0 / (r_0-R)) = -\mathrm{sqrt}((v_0^2 / (r_0-R)) \cdot r - v_0^2R / (r_0-R)) = (v_0 / \mathrm{sqrt}(r_0-R)) \cdot \mathrm{sqrt}(r - R) \quad (5.4.24)$$

To check the result, it is known that *$v(r_0) = v_0$* and *$v(R) = 0$*. Thus, concluding, the speed/space profile that produces a uniformly decelerated motion is a scaled and shifted square root (fig. 5.4.4).

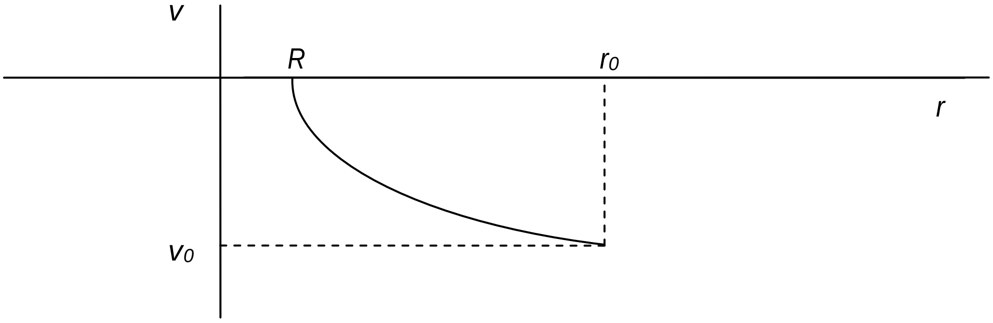


Fig. 5.4.4 – *Speed/space profile for a uniformly decelerated motion*

5.4.4 Our simulation with NXT

Taking the second of the two approaches presented, always within the precision that qualifies NXT, the simulation is pretty simple: we just mount the sonar sensor on the robot and make the applied power depend on the measured distance according to the chosen speed/space profile. To represent the descent of a lunar module, we decided to mount a single motor on the robot that, appropriately de-multiplied, unrolls from a top spool a tape to which the robot is hung to a fixed point (fig. 5.4.5). The speed is controlled as usual by the 'power' parameter of the motor. We have empirically determined the minimum power under which the robot is still: that threshold depends also on the action of the resistant torque caused by the weight of the robot. This parameter must follow the speed profile determined by the measure of the distance from the ground provided by the sonar, which is oriented downwards.

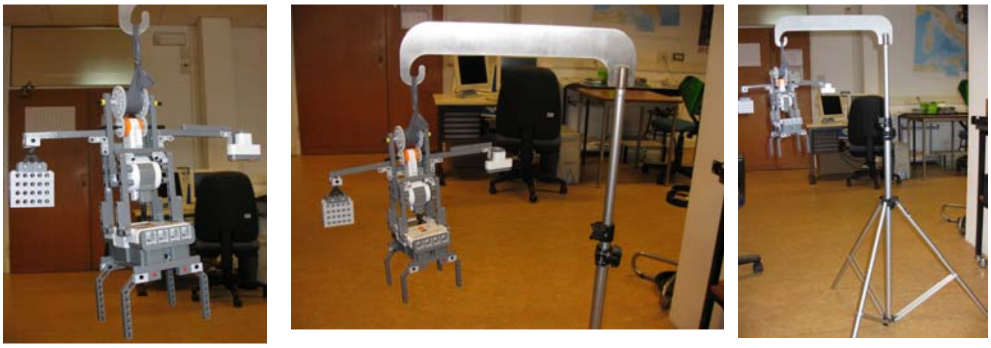


***Fig. 5.4.5** – The lunar module*

Another inaccuracy, especially if the spool has a small radius, is that the radius of unrolling is not constant and, thus, the speed of the robot is not exactly proportional to the angular speed of the motor. Again, this imprecision is self-compensated thanks to the measure of the distance from the ground repeated at the maximum possible frequency, and, in particular, it is certain that, when the robot is very close to the ground, the speed is very low and the impact smooth.

The code is divided into an initial phase and a loop. The initial phase settles the initial speed and distance (variables v_0 and x_0) and checks if the initial position is too small (less than 10), in which case it stops the program. Space is measured in cm and speed in cm/s (fig. 5.4.6).

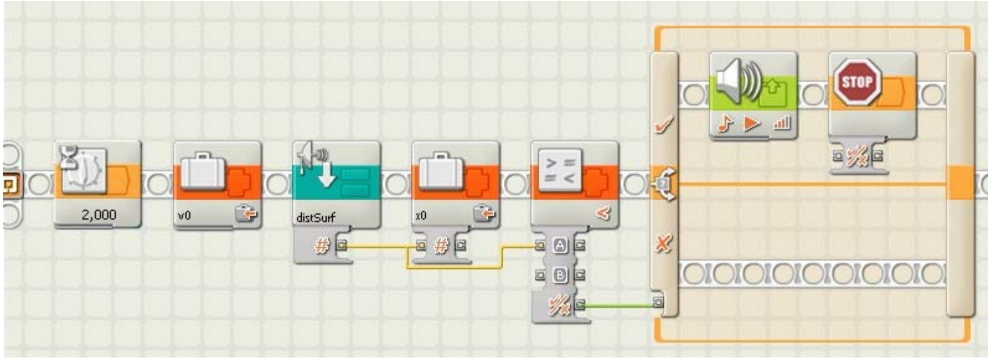


***Fig. 5.4.6** – The Initial phase*

```
VarDecl(Name=v0, Type=NUM) -- the initial speed

VarDecl(Name=x0, Type=NUM) -- the initial position

VarDecl(Name=v, Type=NUM) -- speed
VarDecl(Name=x, Type=NUM) -- position

Wait(Ctrl=TIME, Until=2)

Var(Name=v0.NUM, Act=WR, Val=50)

Ds1:distSurf()

Var(Name=x0.NUM, Act=WR, Val=Ds1.Result)

Cm1:CmpOp(Type=LT, A=Ds1.Result, B=10)

Sw1: Switch(Ctrl=VAL, Type=LOGIC, Dis=ON, CondUp=TRUE,
           Val=Cm1.Res)

[Sw1.IF

          Sound(Act=TONE, Ctrl=PLAY, Vol=75, Rep=OFF, Note=C2,
            Dur=1, Wait=ON)

          Stop()

Sw1.IF]

[Sw1.ELSE

Sw1.ELSE]
```

The *distSurf* block returns the corrected distance from the landing surface, which takes into account the offset of the sonar in respect of the 'feet' of the landing module (fig. 5.4.7).

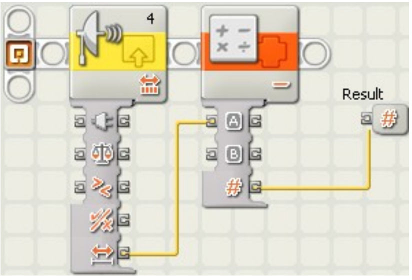


Fig. 5.4.7 – *The distSurf MyBlock*

```
MyBlock(Name=distSurf, InParams=(), OutParams=(Result.NUM)) {

        Ss1:SonarSens(Port=4, Cmp=??, Show=CM)

        Su1:MathOp(Type=SUB, A=Ss1.Dist, B=<offset>)

        SetOut(Name=Result.NUM, Val=Su1.Res)

distSurf}
```

The second part is the main loop, which is responsible for calculating the current speed to be applied to the motor. After having taken the measure of the distance, the code within the loop calculates the corresponding speed in accordance with the speed/space profile, linear in this case. Because the distance is corrected so that it is equal to 0 when the landing module reaches the lunar surface, the profile is simply:

$$v(x) = (v_0 / x_0)\cdot x \qquad (5.4.25)$$

Current distance and speed are assigned respectively to variables x and v. We obtain the code of fig. 5.4.8.

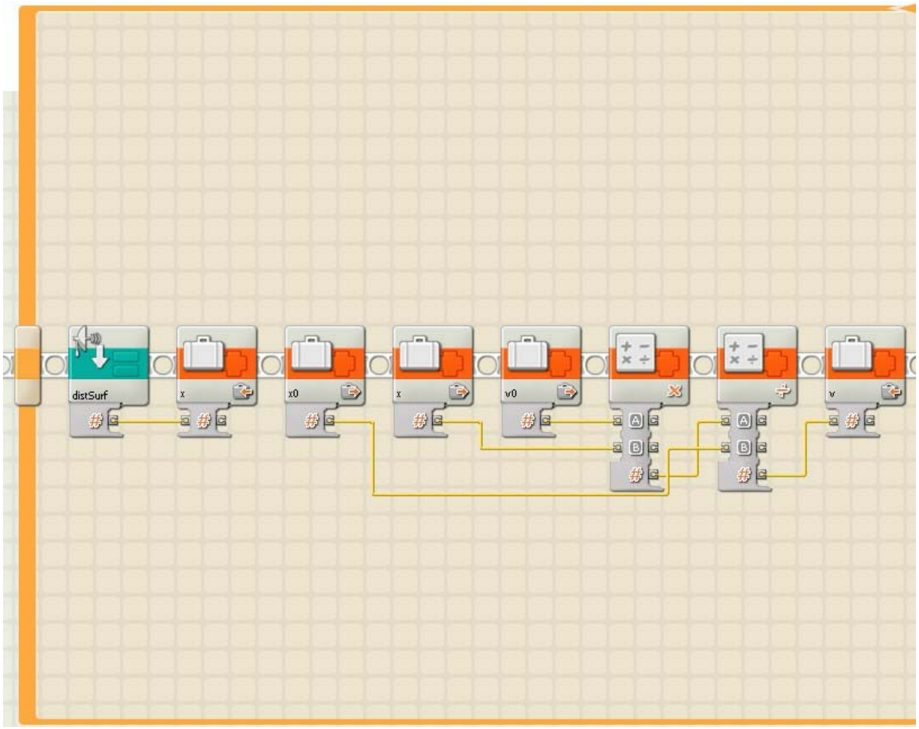


Fig. 5.4.8 – *The main loop (part I)*

```
Loop1: Loop(Ctrl=FOREVER, Dis=OFF) [
        Ds2: distSurf()
        Var(Name=x.NUM, Act=WR, Val=Ds2.Result)
        Vd1:Var(Name=x0.NUM, Act=RD)
        Vd2:Var(Name=x.NUM, Act=RD)
        Vd3:Var(Name=v0.NUM, Act=RD)
        Mu1:MathOp(Type=MUL, A=Vd3.Val, B=Vd2.Val)
        Di1:MathOp(Type=DIV, A=Mu1.Res, B=Vd1.Val)
        Var(Name=v.NUM, Act=WR, Val=Di1.Res)
```

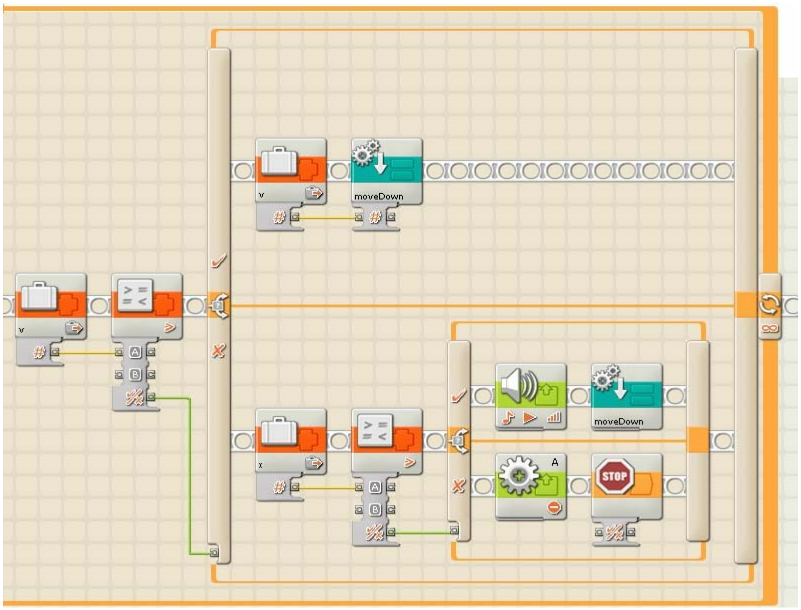


Fig. 5.4.9 – *The main loop (part II)*

Now, we distinguish two cases, one when the vehicle is still far from the surface and makes a normal motion step with the calculated speed; the second is necessary, when the vehicle is very close to surface. Due to the integer calculation limit, the speed could result in 0 even though the vehicle is still going down very slowly (remember the exponential motion). Thus, for practical reasons, we manage this situation by applying a constant speed of 1 until the distance becomes 0: only at that moment we stop the motion (fig. 5.4.9).

```
      Vd4:Var(Name=v.NUM, Act=RD)

      Cm2:CmpOp(Type=GT, A=Vd4.Val, B=1)

      Sw2: Switch(Ctrl=VAL, Type=LOGIC, Dis=ON,
        CondUp=TRUE, Val=Cm2.Res)

      [Sw2.IF

            Vd5:Var(Name=v.NUM, Act=RD)

            moveDown(v=Vd5.Val)

      Sw2.IF]

      [Sw2.ELSE

            Vd6:Var(Name=x.NUM, Act=RD)

            Cm3:CmpOp(Type=GT, A=Vd6.Val, B=0)

            Sw3: Switch(Ctrl=VAL, Type=LOGIC, Dis=ON,
              CondUp=TRUE, Val=Cm3.Res)

            [Sw3.IF

                  Sound(Act=TONE, Ctrl=PLAY, Vol=75,
                    Rep=OFF, Note=B4, Dur=0,5, Wait=OFF)

                  moveDown(v=1)

            Sw3.IF]

            [Sw3.ELSE

                  Motor(Port=A, Dir=STOP, Next=BRK)

                  Stop()

            Sw3.ELSE]

      Sw2.ELSE]

Loop1]
```

The *moveDown* block scales the imposed speed so that the range of speeds (1÷50) corresponds to the range of powers (20÷100). The calculation of the parameters α and β of the linear function *power* = α·*speed* + β is a good exercise, being the resolution of a system of two first grade equations:

$$\begin{cases} 20 = \alpha\cdot 1 + \beta \\ 100 = \alpha\cdot 50 + \beta \end{cases} \qquad (5.4.26)$$

Subtracting the first from the second you obtain:

$$\begin{cases} 80 = \alpha\cdot 49 \quad \Rightarrow \quad \alpha = 80/49 \\ \beta = 20 - \alpha = (980 - 80)/49 = 900/49 \end{cases} \qquad (5.4.27)$$

$$\text{power} = (\text{speed}\cdot 80 + 900) / 49 \qquad (5.4.28)$$

The imposed speed is the *v* formal parameter of the *moveDown* block (fig. 5.4.10).

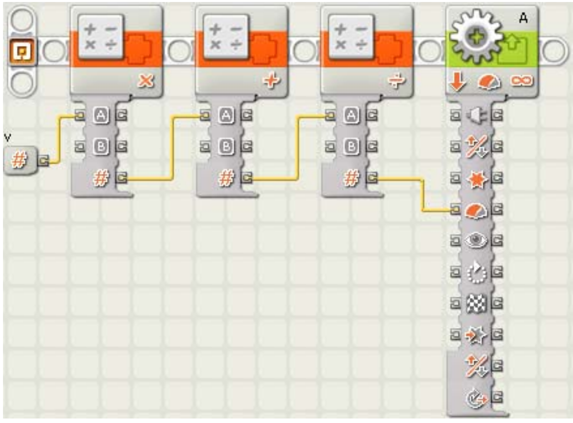


Fig. 5.4.10 – *The moveDown Myblock*

```
MyBlock(Name=moveDown, InParams=(v.NUM), OutParams=()) {

        Mu1:MathOp(Type=MUL, A=v.NUM, B=80)

        Ad1:MathOp(Type=ADD, A=Mu1.Res, B=900)

        Di1:MathOp(Type=DIV, A=Ad1.Res, B=49)

        Motor(Port=A, Dir=BK, Pwr=Di1.Res, PwrCtrl=ON,
          Dur=FOREVER)

moveDown}
```

To reproduce a different speed/space profile, it suffices to adapt the section of the controlling program that calculates the motor power. For the relation that implies a uniformly decreasing speed, you need to use the square root block. In the x_0 variable, the square root of the initial distance (fig. 5.4.11) is stored directly: therefore the first comparison allows the program to go on if the initial distance has a square root greater than or equal to 4. The other necessary modifications are rather straightforward and detailed in the NXT-GTD code (fig. 5.4.12).

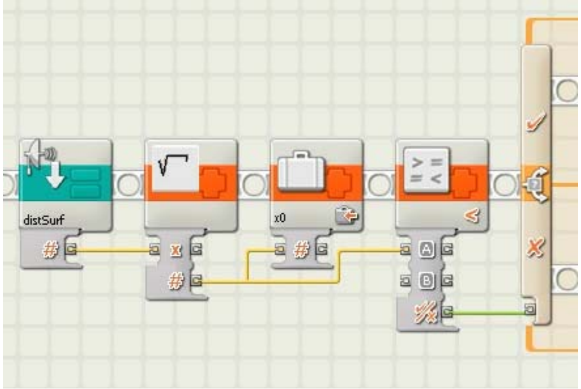


Fig. 5.4.11 – *The initial phase*

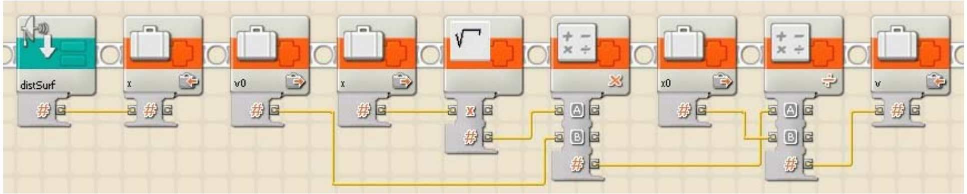


Fig. 5.4.12 – *The speed calculation*

```
Ds1:distSurf()

Sq1:SquareRoot(x1=Ds1.Result)

Var(Name=x0.NUM, Act=WR, Val=Sq1.Sqrt)

Cm1:CmpOp(Type=LT, A=Sq1.Sqrt, B=4)

Sw1: Switch(Ctrl=VAL, Type=LOGIC, Dis=ON, CondUp=TRUE,
          Val=Cm1.Res)

        Ds2: distSurf()
```

```
Var(Name=x.NUM, Act=WR, Val=Ds2.Result)

Vd1:Var(Name=v0.NUM, Act=RD)

Vd2:Var(Name=x.NUM, Act=RD)

Sq2:SquareRoot(x1=Vd2.Res)

Mu1:MathOp(Type=MUL, A=Sq2.Sqrt, B=Vd1.Val)

Vd3:Var(Name=x0.NUM, Act=RD)

Di1:MathOp(Type=DIV, A=Mu1.Res, B=Vd3.Val)

Var(Name=v.NUM, Act=WR, Val=Di1.Res)
```

This kind of experience can be repeated in another form with a simple bi-motorized vehicle (the classic *tribot*) and the sonar put in direction of the motion (fig. 5.4.13). The implementation of a speed/space profile will result as usual in a power-space profile that will produce a straight line motion, which follows the evolution deriving from the chosen profile, such as exponential for a linear profile and uniformly decelerated for a profile of a square root.

Fig. 5.4.13 – *The tribot*

Chapter 6

Useful Resourses for Teachers and Students

Authors: Kyparisia.Papanikolaou, Stassini. Frangou

In this final chapter some useful resources on educational robotics including books, articles and web recourses are recommended for teachers and students to support them in using robotics or designing activities for educational purposes.

6.1 TERECoP Publications (in chronological order)

1. Alimisis, D., Moro, M., Arlegui, J., Pina, A., Frangou, S., Papanikolaou, K. (2007). *Robotics & Constructivism in Education: the TERECoP project*, *Eurologo 2007,* EuroLogo 2007,40 Years of Infuence on Education,Ivan Kalas (ed.),Proceedings of the 11th European Logo Conference, 19-24 August 2007,Bratislava,Slovakia, Faculty of Mathematics, Physics and Informatics, Comenius University, Bratislava, ISBN pp. 978-80-8918.

2. Papanikolaou, K. Frangou, S., Alimisis, D. (2007). *On the development of a framework for the design and implementation of robotic-enhanced activities: the TERECoP project*, In Proceedings of the 4th National Teacher Conference on ICT in Education, Island of Syros, May 2007 (in Greek language).

3. Frangou, S., Papanikolaou, K., Alimisis, D. (2007). *Using RoboLab of Lego Dacta to support programmable robotic constructions*, In Proceedings of the 4th National Teacher Conference on ICT in Education, Island of Syros, May 2007 (in Greek language).

4. Ionita, S., (2007). *Robotics between «object» and «educational tool». Steps towards a constructionist methodological approach*. In Proceedings of Eco-Media International Conference, 23-24 Nov. 2007, Pitesti, Romania, pp.119-125, ISBN978-973-690-701-2.

5. Tocháček D. (2008). *Course TERECoP at KITTV PedF UK – the preparation of future teachers of technical and information education on the use of robotics in the wider of constructructivist education.* In proceedings of the Modern technology in education conference, Faculty of Education of Masaryk University, Brno, Czech Republic; Available at; http:/ /boss.ped .muni.cz/hrbacek/ktivkonf2008/; ISBN: 978-80-7392-091-3 (in Czech language)

6. Arlegui, J., Menegatti, E., Moro, M., Pina, A. (2008). *Robotics, Computer Science curricula and Interdisciplinary activities.* In proceedings of the TERECoP Workshop "Teaching with robotics: didactic approaches and experiences" or-

ganised in the context of SIMPAR 2008 conference, University of Padova, 2008. Available at http://www.simpar-conference.org/

7. Frangou, S., Papanikolaou, K., Aravecchia, L., Montel, L., Ionita, S., Arlegui, J., Pina,A., Menegatti, E., Moro, M., Fava, N., Monfalcon, S., Pagello, I. (2008). *Representative examples of implementing educational robotics in school based on the constructivist approach*. In proceedings of the TERECoP Workshop "Teaching with robotics: didactic approaches and experiences" organised in the context of SIMPAR 2008 conference, University of Padova, 2008. Available at http://www.simpar-conference.org/

8. Papanikolaou, K.m Frangou, S., Alimisis, D. (2008). *Teachers as designers of robotics-enhanced projects: the TERECoP course in Greece,* In proceedings of the TERECoP Workshop "Teaching with robotics: didactic approaches and experiences" organised in the context of SIMPAR 2008 conference, University of Padova, 2008. Available at http://www.simpar-conference.org/

9. Alimisis, D. (2008). *Designing Robotics-enhanced constructivist training for science & technology teachers: the TERECoP project*. In proceedings of ED-MEDIA 2008-World Conference on Educational Multimedia, Hypermedia & Telecommunications, Vienna, Austria. Published by Association for the Advancement of Computing in Education (AACE), USA, p. 288-293.

10. Alimisis, D., Frangou, S., Papanikolaou, K. (2009). *A Constructivist Methodology for Teacher Training in Educational Robotics: the TERECoP Course in Greece through Trainees' Eyes*. In proceedings of the 9th IEEE International Conference on Advanced Learning Technologies, Latvia, July 2009.

11. Frangou, S., Papanikolaou, K. Alimisis, D., Kynigos. Ch. (2009). *Teachers in the role of designers of robotic-enhanced learning activities: the case of the TERECoP training course in Greece*. In Proceedings of the 5th National Teacher Conference on ICT in Education, Island of Syros, May 2009 (in Greek language).

12. Fava, N., Monfalcon, S., Moro, M., Menegatti, E., Arlegui, J., Pina, A. (2009). *Teacher Training In The Scientific Field Through Robotics Activities: Some Experiences From Italy & Spain*. In proceedings of the INTED2009 Conference, Valencia, Spain, March 9-11, 2009.

13. Comite, M., Moro, M. (2009). *How introducing artificial intelligent behaviours in educational robotics*. In proceedings of the INTED2009 Conference, Valencia, Spain, March 9-11, 2009.

14. Moro, M., Alimisis D. (2009). From the Logo Turtle to the Tiny Robot Turtle: practical and pedagogical issues. In *Proceedings of the 5th National Teacher Conference on ICT in Education,* Island of Syros, Greece, May 2009.

15. Alimisis, D. (2009). Robotic technologies as vehicles of new ways of thinking about constructivist teaching and learning: The TERECoP Project in : Robotics & Automation Magazine, IEEE, Sept. 2009, Volume:16, Issue: 3, On page(s): 21-21,23

6.2 Proposed Literature

6.2.1 Books

1. Brooks, J. and Brooks M. (1993). *The case for constructivist classrooms.* Alexandria, Virginia: Association for Supervision and Curriculum Development.
2. Druin, A., & Hendler, J. (2000). *Robots for kids: Exploring new technologies for learning*. San Diego, CA: Academic Press.
3. Harel, I. and Papert S. (1991). *Constructionism.* Norwood, New Jersey: Ablex Publishing Corporation.
4. Maeda, J. (2000). *Maeda@Media*. Rizzoli Publications. New York.
5. Martin, F., Mikhak, B., Resnick, M., Silverman, B., and Berg, R. 2000. To Mindstorms and beyond: Evolution of a construction kit for magical machines. *Robots for kids: Exploring new technologies for learning.* Morgan Kaufmann.
6. Papert S. (1980) Mindstorms – Children, Computers and Powerful Ideas, New York , Basic Books 1980
7. Papert, S. (1991) *Situating Constructionism*. In S.Papert and I.Harel (eds.) Constructionism, Norwood, NJ, Ablex Publishing Corporation.
8. Perkins, D. 1986. *Knowledge as design.* Hillsdale, New Jersey: Lawrence Erlbaum Associates.
9. Resnick, M. 1995. New paradigms for computing, new paradigms for thinking. *Computers and Exploratory Learning.* New York, New York: Springer-Verlag.
10. Thomas, J. W., Mergendoller, J.R., & Michaelson, A. (1999). *Project-based learning: A handbook for middle and high school teachers*. Novato, CA: The Buck Institute for Education.

6.2.2 Papers

1. Ackermann E., (2001) Piaget's constructivism, Papert's constructionism: What's the difference? Future of Learning Group Publication, Available at http://learning.media.mit.edu/content/publications/EA.Piaget%20_%20Papert.pdf

2. Alimisis, D., Karatrantou, A., Tachos, N. (2005), Technical school students design and develop robotic gear-based constructions for the transmission of motion, Eurologo 2005, *Digital Tools for Lifelong Learning*, Proceedings, Warsaw, Poland, 76-86.

3. Carbonaro, M., Rex, M., Chambers, J. (2004), Using LEGO Robotics in a Project-Based Learning Environment. The Interactive Multimedia Electronic *Journal of Computer-Enhanced Learning*, Vol. 6, No 1. Available at http://imej.wfu.edu/articles/2004/1/02/printver.asp Cerezo, E., Baldassarri, S., Pina, A. and Huizi, L. (2005), Learning robotics via web: remote experiment systems for distance training, IADAT-e2005 *2nd IADAT International Conference on Education*, Biarritz, France

5. Denis, B. & Hubert, S. (1999), A conceptual framework of educational robotics, In 9th International conference on Artificial Intelligence in Education, AI-ED 99, Workshop on Educational Robotics, Le Mans, 45 – 54

6. Denis, B. & Hubert, S. (2001), Collaborative learning in an educational robotics environment\, Computers in Human Behaviour, 17, 465 – 480

7. Dimitriou A. and Xatzikraniotis E (2003) Educational robotics as a tool for skills development, In Proceedings of the 2nd national teachers' conference on "ICT in Education", Syros, May 2003, pp. 146-157 (in Greek).

8. Erstad, O. (2002), Norwegian students using digital artifacts in project-based learning, Journal of Computer Assisted Learning 18, 427-437.

9. Han, S. & Bhattacharya, K. (2001). Constructionism, Learning by Design, and Project-based Learning. In M. Orey (Ed.), Emerging perspectives on learning, teaching, and technology. Available Website: http://www.coe.uga.edu/epltt/LearningbyDesign.htm

10. Järvinen Esa-Matti (1998), The Lego/Logo Learning Environment in Technology Education: An Experiment in a Finnish Context; Spring; Journal of Technology Education; Vol. 9, No.2. Available at http://scholar.lib.vt.edu/ejournals/JTE/v9n2/jrvinen.html

11. Kagkani K., Dagdilelis V., Satratzemi M., Evangelidis G., (2005) A case study of teaching programming in secondary education with Lego Mindstorms. In on- line Proceedings of the 3d National Conference "Teaching Computer Science", University of Peloponnese, Korinthos, 7-9 October 2005, http://www.etpe.gr/uploads1/paper_s53.pdf (in Greek).

12. Karatrantou A, Panagiotakopoulos X., Pierri E. (2006) Robotic constructions of Lego Mindstorms and science understanding in primary education: a case study. In Proceedings of the 5th National Conference on ICT in Education, University of Thessaloniki, 5-8 October 2006, pp. 310-317 (in Greek).

13. Karatrantou A, Tachos N., Alimisis D., (2005), Introduction in basic principles and programming structures using the robotic constructions LEGO Mindstorms. In on- line Proceedings of the 3d National Conference "Teaching Computer Science", University of Peloponnese, Korinthos, 7-9 October 2005 http://www.etpe.gr/uploads1/paper_s81 (in Greek).

14. Kynigos C. and Frangou S. (2000), Aspects of pedagogical use of control technology in school class, In Proceedings of the 2nd national conference on "ICT in education", University of Patras, pp. 83-91 (in Greek).

15. Martin, F. (1996), Kids Learning Engineering Science Using LEGO and the Programmable Brick, presented at the American Educational Research Association Annual Meeting

16. Martin, F., Butler, D., and Gleason, W. (2000), Design, story-telling, and robots in Irish primary education, Proceedings of the 2000 IEEE International Conference on Systems, Man, and Cybernetics, Nashville, TN. IEEE Systems, Man, and Cybernetics Society; http://www.cs.uml.edu/~fredm/papers/martin-smc2000.pdf

17. Mataric´, M.J. (2004), Robotics Education for All Ages, Accessible Hands-on AI and Robotics Education. In Proceedings of 2004 AAAI Spring Symposium "Accessible Hands-on AI and Robotics Education". Available at http://www.aaai.org/Library/Symposia/Spring/ss04-01.php

18. McCartney, R. (1996), Introduction to robotics in computer science and engineering education, Computer Science Education, 7(2), 135 – 137

19. Parsons, S. and Sklar. E. (2004). Teaching AI using LEGO Mindstorms. In Proceedings of 2004 AAAI Spring Symposium "Accessible Hands-on AI and Robotics Education". Available at http://www.aaai.org/Library/Symposia/Spring/ss04-01.php

20. Portsmore, M. (1999), RoboLab: Intuitive robotic programming software to support lifelong learning, *Learning Technology Review*, Spring/Summer, 26-39

21. Resnick, M. (1993), Behavior Construction Kits, Communications of the ACM, 36(7), 64 – 71

22. Resnick, M. and Silverman, B. (2005), Some Reflections on Designing Construction Kits for Kids. Proceedings of Interaction Design and Children conference, Boulder, CO. Available at http://llk.media.mit.edu/papers.php

23. Resnick, M., Martin, F., Berg, R., Borovoy, R., Colella, V., Kramer, K. & Silverman, B. (1998), Digital manipulatives: new toys to think with, In Karat, C., Lund, A., Coutaz, J. & Karat, J. (eds.), Proceedings of the SIGCHI Conference on Human Factors in Computing Systems, ACM Press/Addison-Wesley Publishing Co., New York, NY, 281 – 287

24. Resnick, M., Martin, F., Sargent, R., and Silverman, B. (1996). 'Programmable Bricks: Toys to Think With' IBM Systems Journal, vol. 35, no. 3-4, pp.443-452

25. Resnick, M. & Silverman, B. (2005), Some reflections on designing construction kits for kids, In Proceeding of the 2005 conference on Interaction design and children, Boulder, Colorado, 117 – 122

26. Rusk, Z. N., Resnick, M., Berg, R. & Pezalla-Granlund, M. (2008), New Pathways into Robotics: Strategies for Broadening Participation, Journal of Science Education Technology, 17, 59 – 69

27. Turbak, F. & Berg, R. (2002), Robotic Design Studio: Exploring the big ideas of engineering, Liberal Arts Enviroment, Journal of Science Education and Technology, 11(3), 237 – 253.

6.2.3 Web Recourses

1. Educational Robotics Repository: A shared space for collecting curricular materials on the use of educational robotics, currently in undergraduate classes http://www.sci.brooklyn.cuny.edu/~sklar/er/er.html

2. LEGO Engineering – Mindstorms education. http://www.legoengineering.com Site dedicated to providing educators with resources for teaching through engineering with LEGO materials.

3. Lifelong Kindergarten http://llk.media.mit.edu/mission.php The Lifelong Kindergarten group is located within the MIT Media Lab, a hotbed of creative activity.

4. Houghton Mifflin's Project Based Learning Space: A site that supports course instructors, novice teachers, practicing teachers to a) do sustained inquiry on extended problems and projects b) get background knowledge on its theory and use in classrooms, and c) revisit generic teaching concepts. http://www.college.hmco.com/education/pbl/background.html

5. Why do project-based learning? http://pblmm.k12.ca.us/PBLGuide/WhyPBL.html

6. LCSI http://www.microworlds.com/. The site of LCSI where you can find information about the logo like environment for controlling NXT.

Software

7. Lego Mindstorms Web site http://www.legomindstorms.com

8. Robolab http://www.ceeo.tufts.edu/

9. NXT-G iconic language http://www.ni.com/academic/mindstorms/

10. NBC language http://bricxcc.sourceforge.net/nbc/
11. Microsoft Robotic Studio http://msdn.microsoft.com/robotics
12. LCSI Microworlds EX Robotics http://www.microworlds.com/
13. Lego Digital Designer http://ldd.lego.com/

Ideas for robotic-enhanced projects

14. Lego Mindstrorms Projects http://mindstorms.lego.com/nxtlog/ProjectList.aspx
15. The Bee-bot robot http://www.bee-bot.co.uk
16. The Scribbler robot http://www.parallax.com/html_pages/robotics
17. The Pico Cricket robot http://www.picocricket.com
18. Robotis humanoid http://www.robotis.com

Projects on robotics

19. TERECoP project www.terecop.eu
20. IRRE Piemonte http://robotica.irrepiemonte.it/robotica/index.htm
21. Robot@Scuola http://www.scuoladirobotica.it/retemiur/
22. Amico Robot http://www.amicorobot.net /
23. Robotics in Spain http://www.robocity2030.org/
    http://complubot.educa.madrid.org/
24. Robot competitions in Spain http://www.roboteca.org
25. Logo in Spain http://roble.cnice.mecd.es/~apantoja
26. RESS project http://www.cnice.mec.es/pamc/pamc_2003/2003_proyecto_ress/
27. Koldo Olaskoaga http://www.euskalnet.net/kolaskoaga/es/

Appendix1: LegoMindstorms NXT (hardware and software)

Author: Michele Moro

The NXT brick is the brain of a MINDSTORMS® robot. It is a computer-controlled LEGO® brick that can read MINDSTORMS's sensors and command MINDSTORM's motors. The NXT brick is much powerful and flexible with respect to previous LEGO MINDSTORMS programmable bricks (see Table 1 – Technical specifications).

Now, the NXT brick has two microcontrollers one for running the main applications and one for controlling the motors. Both microcontrollers are fitted with FLASH memory so they will not loose your programs nor the firmware if one removes the batteries (differently from the previous RCX brick). The NXT brick can be programmed via USB or Bluetooth. In fact, the LEGO MINDSTORMS NXT includes a wireless Bluetooth node that enables the NXT brick to communicate with other Bluetooth devices. Actually, both of these connections are bidirectional so one can also upload data from the robot to the computer.

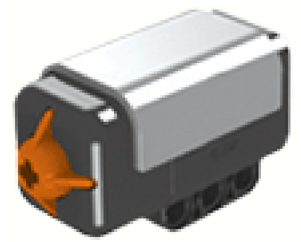
The NXT brick has a LCD display large enough to also play simple video-games. Also, the sound output has been improved, while the old RCX brick could utter only sine wave tones, the new NXT brick, in addition to generate these tones, can also play any pre-recorded audio file in PCM and ADPCM formats.

Table 1-Technical specifications

Main CPU:	32-bit ARM7 microcontroller, 256 Kbytes FLASH, 64 Kbytes RAM
Motor controller:	8-bit AVR microcontroller, 4 Kbytes FLASH, 512 Byte RAM
Input /Output:	Bluetooth wireless communication (Bluetooth Class II V2.0 compliant 460.8 Kbit/s)
	USB full speed port (12 Mbit/s)
	4 input ports for attaching sensors - Ports 1, 2, 3 and 4. The ports uses the 6-wire cable digital platform (One port includes a IEC 61158 Type 4/EN 50 170 compliant expansion port for future use)
	3 output ports for attaching motors - Ports A, B and C with 6-wire cable digital platform
Display:	100 x 64 pixels LCD graphical display
Sound:	Loudspeaker - 8 kHz sound quality. Sound channel with 8-bit resolution and 2-16 KHz sample rate.
Power source:	6 AA batteries

TOUCH SENSOR

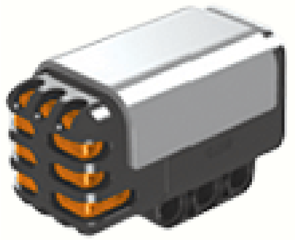
The Touch Sensor gives the robot the sense of touch. The Touch Sensor detects when it is being pressed by something and when it is released again. It has a cross-axle hole at the front of the sensor to attach cross-axle for expanding the sensor range or use it as animals use wiskers.



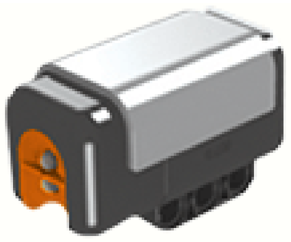
SOUND SENSOR

The Sound Sensor is a microphone that picks-up sounds from the environment. It can work both in decibels [dB] mode and in adjusted decibel [dBA] mode. In **dB mode** the sensor is detecting standard [unadjusted] decibels, all sounds are measured with equal sensitivity. Thus, these sounds may include some that are too high or too low for the human ear to hear. In **dBA mode** the sensor is detecting adjusted decibels, the sensitivity of the sensor is adapted to the sensitivity of the human ear and is limited at the range of frequencies that the humans can ear.

The Sound Sensor can measure sound pressure levels up to 90 dB – about the level of a lawnmower. Sound pressure levels are extremely complicated, so the Sound Sensor readings on the MINDSTORMS NXT are displayed in percent [%]. The lower the percent the quieter the sound. For example:



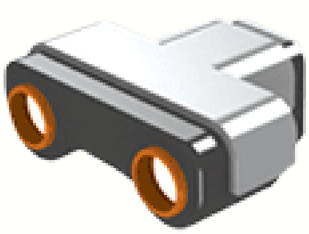
- 4-5% is like a silent living room
- 5-10% would be someone talking some distance away
- 10-30% is normal conversation close to the sensor or music played at a normal level
- 30-100% are people shouting or music being played at a high volume



LIGHT SENSOR

The Light Sensor is one of the two sensors that give your robot vision [The Ultrasonic Sensor is the other]. This sensor is made of a light emitter (a LED) and a light meter (a photocell). It has to be noticed that the new NXT sensor has a small plastic barrier between the emitter and the sensor solving the problem of emitter light going directly into the sensor. The Light Sensor can measure the light intensity reflected by colored surfaces at the short range. Alternatively, the emitter can be disabled and the sensor can measure the ambient light, so the robot can distinguish between light and dark or measure the light level in different locations of the room.

ULTRASONIC SENSOR



The Ultrasonic Sensor is one of the two sensors that give your robot vision [The Light Sensor is the other]. The Ultrasonic Sensor enables your robot to detect objects and to measure the distance to them. You can also use it to make your robot avoid obstacles and detect movement.

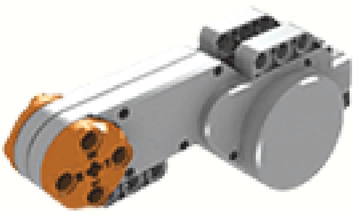
The Ultrasonic Sensor measures distance in centimeters and in inches. It is able to measure distances from 0 to 255 centimeters with a precision of +/- 3 cm.

The Ultrasonic Sensor uses the same scientific principle as bats: it measures distance by calculating the time it takes for an ultrasound wave to hit an object and return (this is called time of flight). Again like in the light sensor, there is an emitter (i.e., an ultrasound emitter) and a receiver (i.e., a microphone) picking up the echo waves reflected by the obstacles in front of the sensor.

Large sized objects with hard surfaces return the best readings. Objects made of soft fabric or that are curved [like a ball] or are very thin or small can be difficult for the sensor to detect. This sensor can give false reading also with hard surfaces oriented at small angles with respect to the sensor, because this might cause the ultrasound wave to be specularly reflected in a different direction and no echo is coming back to be picked up by the sensor receiver. Another situation that can disturb the sensor readings is the presence of other ultrasonic sensors (maybe mounted on other robots). One sensor could pick up the ultrasound wave emitted by a second sensor, instead of the echo of its own emitted waves and thus calculating an incorrect time of flight and then calculating an incorrect distance to the obstacles. Fortunately, the ultrasonic sensors have a special operation mode that help solving such a situation.

The three Servo Motors give the robot the ability to move. The NXT motors are larger and less efficient than the RXC motors, but have several improvements. They have a gear train that provides less speed and more torque, so most of the times complex gear-down trains are no longer needed. They provide an integrated encoder or Rotation Sensor to read the exact motor position. The Rotation Sensor measures motor rotations in degrees [accuracy of +/- one degree]. One rotation is equal to 360 degrees. The Rotation Sensors combined with the new capability of the NXT firmware to control motors in pairs to synchronise them enables precise control of robot's movements. For instance, using the Move block in the LEGO MINDSTORMS NXT-G software to program the motors, the two motors will automatically synchronise, so that the robot will move in a straight line. Moreover, the new motors have two different connection possibilities either the central hole to fix cross-axles of different lengths or four off-center holes to fix the standard Technic connetors.

SERVO MOTORS

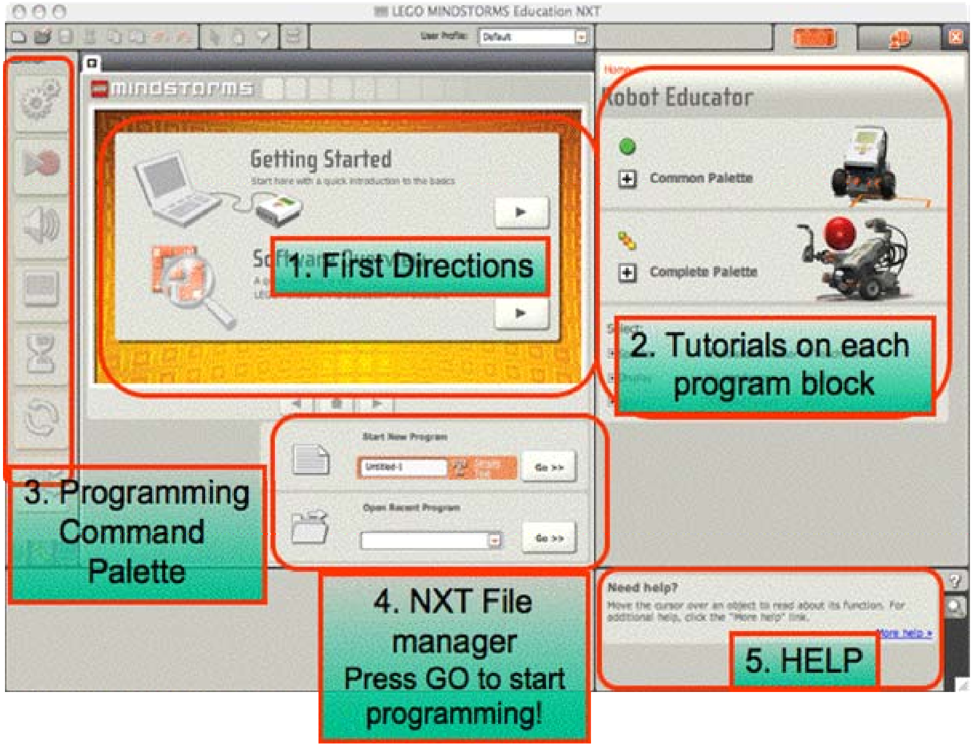


Third party sensors.

The four input ports for attaching sensors (namely Port 1, 2, 3, and 4) are compatible with the industrial standard I^{2}C bus. This allow everyone to develop a new sensor that comply with the I^{2}C standard and to read it through the NXT sensor ports. Companies like HiTechnic and Mindsensors produce several sensors that use the I^{2}C bus.

MINDSTORM NXT-G Software

LEGO® MINDSTORMS® NXT-G software enables to simply program the NXT even without knowing any programming language. NXT-G is a what is called a visual programming environment, this means one will use a graphical interface to develop code (the G in NXT-G stands for graphical). With NXT-G one can upload his/her programs to the NXT via USB or Bluetooth connectivity. The intuitive drag and drop software, powered by National Instruments LabVIEW, comes with building instructions and programming guides to easily begin constructing and programming with MINDSTORMS NXT. Users create a program by selecting and dragging functional blocks to a canvas area from the palettes. In the palettes, one can find blocks implementing all basic functionalities of the NXT kit. In other words, each block is a programming method. The user/programmer can set different values for the parameters of the method by setting the block's sliders, drop-dwon menu, radio button, text boxes, etc.



LEGO MINDSTORMS Education NXT
User Profile: Default
MINDSTORMS
Getting Started
Start here with a quick introduction to the basics
1. First Directions
Robot Educator
Common Palette
Complete Palette
2. Tutorials on each program block
3. Programming Command Palette
Start New Program
Untitled-1
Go >>
Open Recent Program
Go >>
4. NXT File manager Press GO to start programming!
5. HELP
More help »

Need help?
Move the cursor over an object to read about its function. For additional help, click the "More help" link.

Fig. 1 MINDSTORMS NXT-G Software

The Figure 1 depicts the first window appearing when opening NXT-G. One can notice five major areas, as highlighted in Figure 1. Area 1: the first directions for novices on how to get start with NXT-G. Area 2: a tutorial on each programming block. Area 3: the area of the programming palettes containing the programming blocks. Area 4: the file manager to open an existing file or to give a name of your new project. Area 5: the detailed help in which one can find all he/she needs. As soon as a new block is input for the new project a new window appear. Here, one can find again on the right the help and tutorials section.

In Fig. 2, we highlighted three areas. Area 1: the programming canvas where program blocks have to be dropped and ordered to obtain the desired robot behaviour. Area 2: the NXT brick controller, and Area 3: the palette selector used to switch between the three block palettes.

Once, a NXT-G program is created it has to be transferred to the robot and it has to be executed by the NXT brick. This process is governed using the controller (see Fig. 3). The controller has five buttons. Top left button connects NXT-G with the NXT brick and once connected information status of the NXT brick will be displayed (e.g., battery level, internal memory usage, etc.). Bottom left button downloads the open program from the computer to the NXT brick. The central button downloads and runs the open program on the NXT brick. The top right button downloads and runs a different program to be selected on the NXT brick. The bottom right button stops the current program running on the NXT brick.

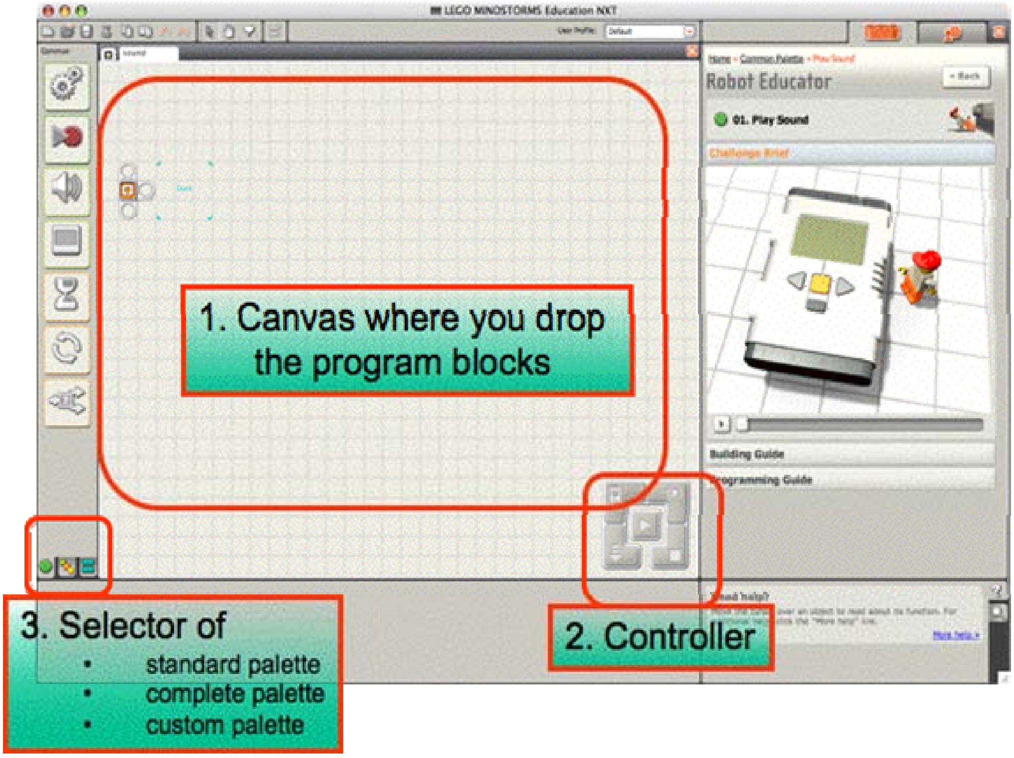


Fig. 2 MINDSTORMS NXT-G Software

The programming environment is very simple for novice, but is more powerful that one could expect. You can create custom blocks that contains lot of basic blocks

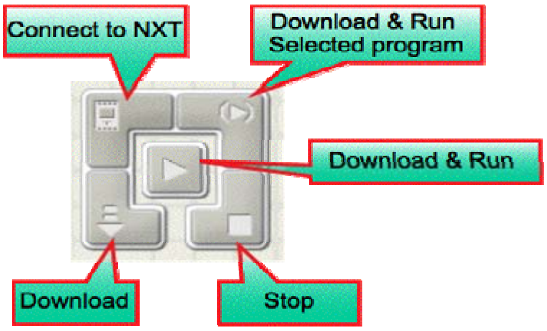
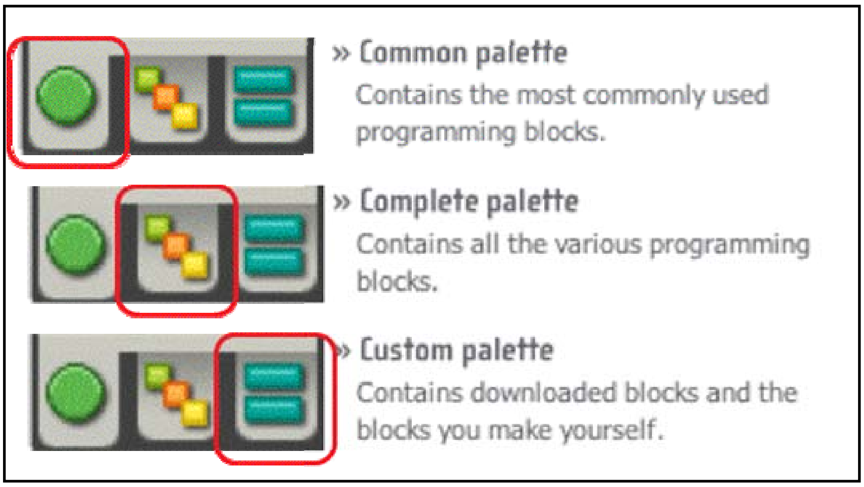


Fig. 3 MINDSTORMS NXT-G Software

(i.e. lot of code) that connected together realise a specific task. This custom block can be saved to the custom palette and used later as a part of a larger program. This makes easy to fit a lot of code into the limited graphic canvas, There are three palettes (see Fig. 4): the common palette, the complete palette, and the custom palette. In the custom palette one can import blocks created bu other users or for new devices as they become available for the NXT.



» Common palette
Contains the most commonly used programming blocks.

» Complete palette
Contains all the various programming blocks.

» Custom palette
Contains downloaded blocks and the blocks you make yourself.

Fig. 4 MINDSTORMS NXT-G Software

Although the iconic programming environment of the graphical programming interface is in general very easy to learn for users with no programming experience, sometimes it might be misleading. First of all, it is not always clear what a block or a diagram of block is doing. Second, sometimes the iconic language primitives might be much more ambiguous than a textual programming language primitive. In the end, when the experience of the user grows over a certain threshold and thus his/her programs grow over a certain complexity and size it is hard to manage the code just with the graphical interface both because of space and because of NXT-G software instability. Once one reach a certain degree of confidence and skill in controlling the NXT kit, he/she should move to a textual programming language like Java or C. For using Java, LeJOS NXJ is the choice. For using C, NXC is the choice.

Appendix 2: A textual description language equivalent to the NXT-G graphical language

Author: Michele Moro

Objective

Definition of a textual language exactly equivalent to NXT-G to be used in presenting NXT programming code in substitution of large snapshots of the NXT-G GUI. It is defined using English acronyms but several national versions can be easily provided.

General requirements

- A name can be enclosed within “” when spaces are present in the name
- Blocks are functionally represented with a recognizable name, with an optional label for forward/backward references
- Any block parameter is specified as a couple <paramname>=<value> where value can be a predefined constant (e.g. ON, OFF, COUNT), a numerical or logical value (e.g. 123, TRUE), or the output of a previous block connected through a data wire
- An output of a block is used as a qualifying name of the label representing such a block when such an output must be connected to another block input: thus the qualified label is a way to represent an input data wire coming out from the labelled block (e.g. SUM1:MathOp(Type=ADD, A=37, B=56) . . . MathOp(Type=MUL, A=SUM1.Res, B=-23) SUM1.Res represents a data wire connecting the Res(ult) output of the SUM1 adding block with the input A of the multiplier block)
- The list of possible input/output connections is given at the end of the description of each command block. When a input connection has its corresponding output, the two connections share the same name: for example, in the Motor block labelled with M1, the Port parameter represents also an input/output connection, that is its may be defined as the output of one other block, as explained above, but also M1.Port could be the definition of a parameter of another block.

Structure

- Variable declaration is an off-line activity

VarDecl (Name=<varName>, Type=LOGIC|NUM|TXT)

- MyBlock definition

```
MyBlock (Name=<BlockName>, InParams=(<InputParamList>),
  OutParams=(<OutputParamList>)) {
        -- code for the command block
<BlockName>}
```

where inputParamList and outputParamList are lists of qualified names:
<ParamName>.[LOGIC|NUM|TXT]

In the block body the reading references to the input parameters (i.e. the ‘abstract’ input data wires of the block) are made with the name of the parameter, optionally qualified with its type indicator (e.g. Par1.NUM) for better readability. ‘Abstract’ output data wired, used to set the output parameters, are expressed by the block :

```
SetOut(
        Name=<OutputParamName>.[LOGIC|NUM|TXT]
        Val = TRUE | FALSE – if .Type==LOGIC
                <IntVal> -- if .Type==NUM
                <Text> -- if .Type==TXT
        )
```

- Multitasking is equivalent to a fork in the flow control; it can be nested in more than one level

```
Fork
{1 <commands> 1}            {1 . . .{1.1 . . . 1.1} 1}
{2 <commands> 2}
```

- Control structures define blocks of code, delimited by square bracket, generally nestable.

Loop

```
<lab>:Loop(. . .) [
        -- loop body
<lab>]
```

Switch

1. Two choices (equivalent to an if-then-else)

```
<lab> Switch(. . .) [<lab>.IF
        -- true part
<lab>.IF]
[<lab>.ELSE
        -- else part
<lab>.ELSE]
```

2. Multi-choice (equivalent to a switch-case)

```
<lab> Switch(. . .) [<lab>.case1
        -- case 1 part
<lab>.case.1]
```

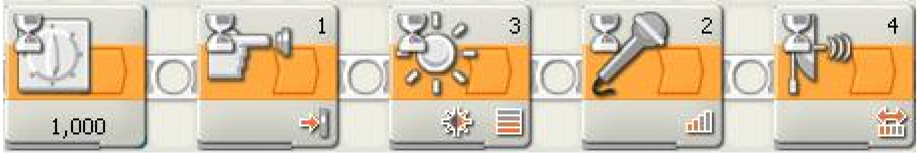
```
[<lab>.case.2
        -- case 2 part
<lab>.case.2]
. . .
[<lab>.case.n
        -- case n part
<lab>.case.n]
```

Common palette

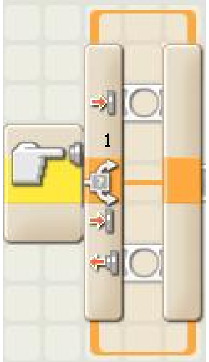
```
Move (
        Ports = A | B | C | AB | AC | BC | ABC
        Dir = FD | BK | STOP
        StLt = A | B | C
        StRt = A | B | C
        Steer = <-100 .. 100>
        Pwr = <0 .. 100>
        Dur = FOREVER | <IntVal>.[DEG | ROT | SEC]
        Next = BRK|COA
        )
        In=MotorL(=1|2|3), MotorR(=1|2|3), MotorO(=1|2|3), Dir(=TRUE|FALSE), Steer, Pwr,
        Dur, Next (=TRUE|FALSE)
        Out=see In
```

```
Record (
        Act=REC | PLAY
        Name = <Name>
        Ports = A | B | C | AB | AC | BC | ABC
        Time = <NatVal> -- if Act==REC
        )
        In=Act (=0|1), Name, PortA(=TRUE|FALSE), PortB(=TRUE|FALSE),
        PortC(=TRUE|FALSE), Time, Rate(=<0..255> Hz)
        Out=see In
```

```
Sound (
        Act = FILE | TONE
        Ctrl = PLAY | STOP
        Vol = <0..100>
        Rep = ON | OFF
        File = <Text> -- if Act==FILE
        Note = <C1..B3> -- if Act==TONE
        Dur = <Val> -- if Action==TONE
        Wait = ON | OFF – if Rep==OFF
        )
        In=Act(=0|1), File, Freq(=<NatVal> Hz), Ctrl (=0|1), Vol,
        Dur (=<NatVal> in ms)
        Out=see In
Display (
        Act = IMG | TXT | DRAW | RESET
        Clr = ON | OFF
        File = <Name> -- if Act==IMAGE
        Txt = <Text> -- if Act== TEXT
        Type = POINT | LINE | CIRCLE -- if Act=DRAW
        PosX = <0..99> -- if Act!=RESET
        PosY = <0..63> -- if Act!=RESET
        LastX = <0..99> -- if Act==DRAW && Type=LINE
        LastY = <0..63> -- if Act==DRAW && Type=LINE
        Line = <1..8> -- if Act==TEXT
        Rad = <0..120> -- if Act==DRAW && Type=CIRCLE
        )
        In=Act(0=IMG, 1=TXT, 2=POINT, 3=LINE, 4=CIRCLE, 5=RESET),
        Clr(=TRUE|FALSE), File, Text, PosX, PosY, LastX,
        LastY, Rad
        Out=see In
```



```
Wait (
        Ctrl = SENSOR | TIME
        Sensor= SOUND | TOUCH | LIGHT | BUTTONS | RCVMSG | ROT | TIMER
                        SONAR | TOUCH* | LIGHT* | ROT* | TEMP* -- if Ctrl== SENSOR
        Port = 1 | 2 | 3 | 4 -- if Ctrl==SENSOR && Sensor==SOUND, TOUCH, LIGHT,
                                -- SONAR, TOUCH*, LIGHT*, ROT*, TEMP*
        Button = OK | LEFT | RIGHT -- if Ctrl==SENSOR && Sensor==BUTTONS
        Msg = TEXT | NUMBER | LOGIC -- if Ctrl==SENSOR && Sensor = RCVMSG
        Port = A | B | C -- if Ctrl==SENSOR && Sensor==ROT
        Timer = 1 | 2 | 3 -- if Ctrl==SENSOR && Sensor == TIMER
        Until = [GT | LT] <0 .. 100> -- if Ctrl==SENSOR &&
                                    -- Sensor==SOUND, LIGHT, LIGHT*
                [FD | BK] [GT LT] <IntVal>.[DEG | ROTS]
                        -- if Ctrl==SENSOR && Sensor == ROT
                [GT | LT] <IntVal> -- if Ctrl==SENSOR && Sensor==TIMER
                [GT | LT] <0 .. 250> -- if Ctrl==SENSOR &&
                                        -- Sensor==SONAR && Show == CM
                [GT | LT] <0 .. 100> -- if Ctrl==SENSOR &&
                                        -- Sensor==SONAR && Show == IN
                [FD | BK] [GT | LT] <IntVal> -- if Ctrl==SENSOR &&
Sensor==ROT*
                [GT | LT] <-20 .. 70> -- if Ctrl==SENSOR &&
                                        -- Sensor==TEMP* && Show = CEL
                [GT | LT] <-4 .. 158> -- if Ctrl==SENSOR &&
                                        -- Sensor==TEMP* && Show = FAR
                 <NatVal> -- if Ctrl==TIME
        Act = PRESS | REL | BUMP -- if Ctrl==SENSOR &&
                                                -- Sensor = TOUCH, BUTTONS,
                                        TOUCH*
        Mailbox = <1 .. 10> -- if Ctrl==SENSOR && Sensor = RCVMSG
        CmpWith = <Text> -- if Ctrl==SENSOR &&
                                --  Sensor==RCVMSG && Msg == TEXT
                        <IntVal> -- if Ctrl==SENSOR &&
                                --  Sensor==RCVMSG && Msg == NUMBER
                        TRUE | FALSE -- if Ctrl==SENSOR &&
                                        --    Sensor==RCVMSG && Msg ==
                                        LOGIC
        Show = CM | IN -- if Ctrl==SENSOR && Sensor == SONAR
                CEL | FAR -- if Ctrl==SENSOR && Sensor == TEMP*
        Light = ON | OFF -- if Ctrl==SENSOR && Sensor == LIGHT
        )
        In=//
        Out=//
Loop (
        Ctrl = FOREVER | SENSOR | TIME | COUNT | LOGIC
        -- if Ctrl==SENSOR || Ctrl==TIME see the options in Wait block
        Until = <NatVal> -- if Ctrl==COUNT
                TRUE | FALSE -- if Ctrl=LOGIC
        Dis = ON | OFF
        )
        In=Until (=TRUE | FALSE if Ctrl=LOGIC)
        Out=Cnt
Switch (
        Ctrl = SENSOR | VAL
        -- if Ctrl==SENSOR see the options in Wait block
        -- for fields Sensor, Cmp and CondUp
        Type = LOGIC | NUMBER | TXT -- if Ctrl==VAL
```



```
        Dis=ON|OFF
        CondUp=TRUE|FALSE -- if Ctrl==VAL && Type==LOGIC
        Cond<1..n>=<IntVal>.DFLT-- if Ctrl==VAL && Type==NUMBER && Dis==OFF
        Cond<1..n>=<Text>.DFLT-- if Ctrl==VAL && Type==TXT && Dis==OFF
        )
        In=Val (if Ctrl==VAL)
        Out=//
```

Action palette

```
Motor (
        Port = A | B | C
        Dir = FD | BK | STOP
        Act = CONST | RAMPUP | RAMPDW -- if Dur.Type==DEG || Dur.Type==ROT
        Pwr = <0 .. 100>
        PwrCtrl = ON | OFF
        Dur = FOREVER | <IntVal>.[DEG | ROT | SEC]
        Wait = ON | OFF – if Dur.Type==DEG || Dur.Type==ROT
        Next = BRK|COA -- if Wait==ON || Dur.Type=SEC
        )
        In=Port, Dir(=TRUE|FALSE), Act(=0|1|2), Pwr, PwrCtrl(=TRUE|FALSE),
        Dur, Wait (=TRUE|FALSE),Next (=TRUE|FALSE)
        Out=see In, DirOut(=TRUE|FALSE), DegOut(=<NatVal>)
SendMsg (
        Conn = 0 | 1 | 2 | 3
        MsgType = TXT | NUM | LOGIC
        Msg = <Text> -- if MsgType==TXT
        Msg = <IntVal> -- if MsgType==NUM
        Msg = TRUE | FALSE -- if MsgType==LOGIC
        Mbx = <1..10>
        )
        In=Conn, Mbx, Txt (if MsgType==TXT), Val (if MsgType==NUM),
        Bool (if MsgType==LOGIC)
        Out=see In
Motor* (
        Port = A | B | C
        Dir = FD | BK | STOP
        Pwr = <0 .. 100>
        )
        In=Port, Dir(=TRUE|FALSE), Pwr
        Out=see In
```

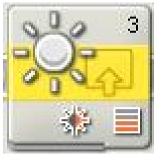
```
Lamp* (
        Port = A | B | C
        Act = ON | OFF
        Int = <0 .. 100>
        )
        In=Port, Act (=TRUE|FALSE), Int
        Out=see In
```

Sensors palette

```
TouchSens (
        Port = 1 | 2 | 3 | 4
        Act = PRESS | REL | BUMP
        )
        In=Port, Act(=0|1|2)
        Out=see In, Res(=TRUE|FALSE), Raw (=<0..1024>)
```

```
SoundSens (
        Port = 1 | 2 | 3 | 4
        Cmp = [GT | LT] <0 .. 100>
        )
        In=Port, Cmp, Gt(=TRUE|FALSE), dbA (=TRUE|FALSE)
        Out=see In, Res(=TRUE|FALSE), Lev (=<0..100>), Raw (=<0..1024>)
```

```
LightSens (
        Port = 1 | 2 | 3 | 4
        Cmp = [GT | LT] <0 .. 100>
        Light = ON | OFF
        )
        In=Port, Cmp, Gt(=TRUE|FALSE), Light
        Out=see In, Res, (=TRUE|FALSE), Int (=<0..100>), Raw (=<0..1024>)
```



```
SonarSens (
        Port = 1 | 2 | 3 | 4
        Cmp = [GT | LT] <0 .. 250> -- if Show == CM
        Cmp = [GT | LT] <0 .. 100> -- if Show == IN
        Show=CM | IN
        )
        In=Port, Cmp, Gt(=TRUE|FALSE)
        Out=see In, Res, Dist
```

```
Buttons (
        Button = OK | LEFT | RIGHT
        Act = PRESS | REL | BUMP
        )
        In=Button(=1|2|3), Act(=0|1|2)
        Out=see In, Res
```

```
RotSens (
        Port = A | B | C
        Act = RD | RST
        Cmp = [FD | BK] [GT LT] <IntVal>.[DEG | ROTS]
        )
        In=Port, Tdir(=TRUE|FALSE), Cmp, Gt(=TRUE|FALSE),
          Rst(=TRUE|FALSE)
        Out=see In, Res(=TRUE|FALSE), Dir(=TRUE|FALSE),
          Deg(=<NatVal>)
```

```
Timer (
        Num = 1|2|3
        Act = RD | RST
        Cmp = [GT LT] <0..100>
        )
        In=Num, Cmp, Gt(=TRUE|FALSE), Rst(=TRUE|FALSE)
        Out=see In, Res(=TRUE|FALSE), Val(=<NatVal>)
```

```
RcvMsg (
        MsgType = TXT | NUM | LOGIC
        Cmp = <Text> -- if MsgType==TXT
        Cmp= <IntVal> -- if MsgType==NUM
        Cmp = TRUE | FALSE -- if MsgType==LOGIC
        Mbx = <1..10>
        )
        In= Mbx, Txt (if MsgType==TXT), Val (if MsgType==NUM),
        Bool (if MsgType==LOGIC)
        Out=see In, Rcv(=TRUE|FALSE), Cmp(=TRUE|FALSE), TxtOut(if
        MsgType==TXT),
        NumOut(if MsgType==NUM), BoolOut(if MsgType==LOGIC)
```

```
TouchSens* (
        Port = 1 | 2 | 3 | 4
        Act = PRESS | REL | BUMP
        )
        In=Port, Act(=0|1|2)
        Out=see In, Res(=TRUE|FALSE), Raw (=<0..1024>)
```

```
RotSens* (
        Port = 1 | 2 | 3 | 4
        Act = READ | RESET
        Cmp = [FD | BK] [GT LT] <NatVal>
        )
        In=Port, Tdir(=TRUE|FALSE), Cmp, Gt(=TRUE|FALSE),
          Rst(=TRUE|FALSE)
        Out=see In, Res(=TRUE|FALSE), Dir(=TRUE|FALSE),
          Tick(=<NatVal>)
```

```
LightSens* (
        Port = 1 | 2 | 3 | 4
        Cmp = [GT | LT] <0 .. 100>
        )
        In=Port, Cmp, Gt(=TRUE|FALSE)
        Out=see In, Res, (=TRUE|FALSE), Int (=<0..100>), Raw (=<0..1024>)
TempSens* (
        Port = 1 | 2 | 3 | 4
        Cmp = [GT | LT] <-20 .. 70> -- if Show = CEL
                [GT | LT] <-4 .. 158> -- if Show = FAR
        )
        In=Port, Cmp, Gt(=TRUE|FALSE)
        Out=see In, Res, (=TRUE|FALSE),
          Temp (=<-20..70> if Show=CEL, <-4 .. 158> if Show = FAR),
          Raw (=<0..1024>)
```

Control palette

```
Stop (
        //
        )
        In=Cond(=TRUE|FALSE)
        Out=see In
```

Data palette

```
LogicOp (
        Type = AND | OR | XOR | NOT
        A=TRUE | FALSE
        B=TRUE | FALSE
        )
        In=A, B
        Out=see In, Res(=TRUE|FALSE)
MathOp (
        Type = ADD | SUB | MUL | DIV
        A=<IntVal>
        B=<IntVal>
        )
        In=A, B
        Out=see In, Res(=<IntVal>)
CmpOp (
        Type = LT | GT | EQ
        A=<IntVal>
        B=<IntVal>
        )
```

```
        In=A, B
        Out=see In, Res(=TRUE|FALSE)
```

```
Range   (
        Type = IN | OUT
        A = <0..B>
        B = <A..100>
        Val = <IntVal>
        )
        In=A, B, Val
        Out=see In, Res(=TRUE|FALSE)
```

```
Rand (
        A = <0..B>
        B = <A..100>
        )
        In=A, B
        Out=see In, Val(=<A..B>)
```

```
Var (
        Name = <Name>.[LOGIC | NUM | TXT]
        Act = RD | WR
        Val = TRUE | FALSE -- if .Type==LOGIC && Act==WR
                <IntVal> -- if .Type==NUM && Act==WR
                <Text> -- if .Type==TXT && Act==WR
        )
        In=Val (if Act==WR)
        Out=Val
```

Advanced palette

```
Text (
        A = <Text>
        B = <Text>
        C = <Text>
        )
        In=A, B, C
        Out=see In, Val(=<Text>)
```

```
N2Txt (
        Num = <IntVal>
        )
        In=Num
        Out=see In, Txt(=<Text>)
```

```
KeepAl (
        //
        )
        Out=Time(=<intVal> in ms)
```

```
File (
        Act = RD | WR | CL | DL
        Name = <Filename>
        Type = TXT | NUM -- if Act==RD || Act==WR
        Txt = <Text> -- if Act==WR && Type==TXT
        Val = <IntVal> -- if Act==WR && Type==NUM
        )
        In=Name, Dim(=<NatVal> if Act==WR && file doesn’t exist),
        Txt(if Act==WR && Type==TXT), Val(if Act==WR && Type==NUM)
        Out=see In, Err(=TRUE|FALSE), TxtOut(if Act==RD && Type==TXT),
        NumOut(if Act==RD && Type==NUM)
```

```
Calib (
        Port = 1 | 2 | 3 | 4
        Sens = SOUND | LIGHT | LIGHT*
        Act = CLB | ERA
        Val = MAX | MIN -- if Act==CLB
        )
        In=Port, Max(=TRUE|FALSE), Era(=TRUE|FALSE)
        Out=see In
```

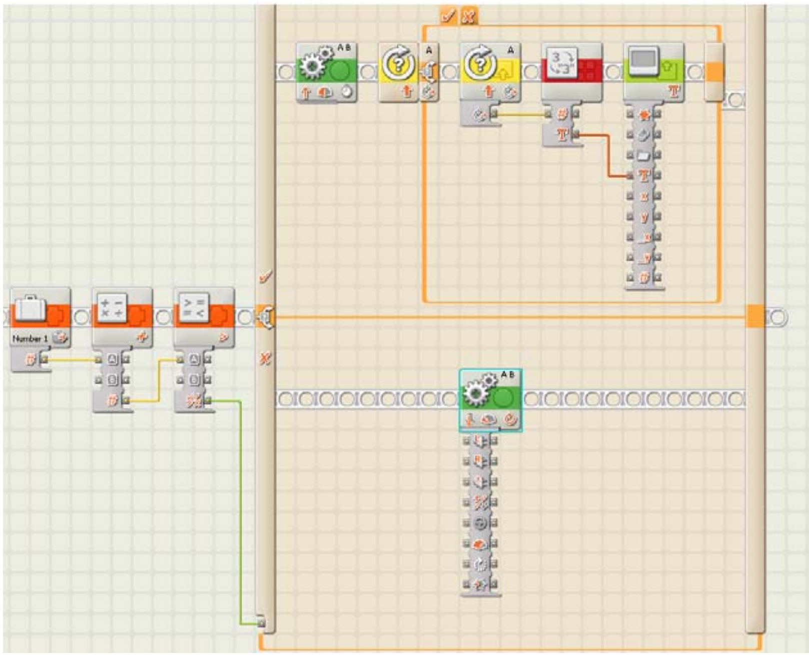
```
Reset (
        A = ON | OFF
        B = ON | OFF
        C = ON | OFF
        )
        In=A, B, C
        Out=see In
```

Example

```
V1:     Var(Name=”Number 1”.NUM, Act=RD)
A1:     MathOp(Type=ADD, A=V1.Val, B=20)
C1:     CmpOP(Type=GT, A=A1.Res, B=30)
Sw1:    Switch(Ctrl=VAL, Type=LOGIC, Val=C1.Res, Dis=ON, CondUp=TRUE)
[Sw1.IF
                Move(Ports=AB, Dir=FD, StLt = A, StRt = B,
                  Steer=0, Pwr=60, Dur=2.SEC, Next=BRK)
        Sw2:    Switch(Ctrl=SENSOR, Sensor=ROT, Dis=OFF, Port=A, Act=RD,
                  CondUp=FD GT 360.DEG) [Sw2.IF
                Rs1:    RotSens(Port = A, Act=READ, Cmp=GT 0.DEG)
                Nt1:    N2Txt(Num=Rs1.Sound)
                        Display(Act=TXT,     Clr=ON,     Txt=Nt1.Txt,     PosX=8
        PosY=32, Line=4)
                Sw2.IF]
```

```
            [Sw2.ELSE       Sw2.ELSE]
        Sw1.IF]
        [Sw1.ELSE
            Move(Ports=AB, Dir=BK, StLt = A, StRt = B,
             Steer=0, Pwr=30, Dur=1.ROT, Next=BRK)
        Sw1.ELSE]
```

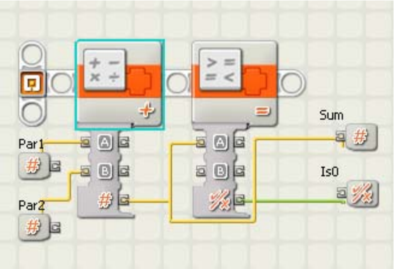
It corresponds to:



The My Block example:

```
MyBlock(Name=sumis0, InParams=(Par1.NUM, Par2.NUM),
  OutParams=(Sum.NUM, Is0.LOGIC)) {
        S01: MathOp(Type=ADD, A=Par1.NUM, B=Par2.NUM)
        S02: CmpOp (Type=EQ, A=S01.Res, B=0)
        SetOut(Name=Sum.NUM; Val=S01.Res)
        SetOut(Name=Is0.LOGIC; Val=S02.Res)
sumis0}
```

it corresponds to the following:


Par1
Par2
A
B
A
B
Sum
Is0