Friday, 17 August 2001
Håkan Johansson

Robotic courses based on LEGO™

Abstract

Robotic courses based on LEGO^{TM}

For many years the courses in the subject of robotics have been held at universities all over the world. These courses have ranged from the introduction in the field of robotics to advanced courses digging deep into some special aspects of robotics. The field of robotics is vast and in this thesis I will concentrate as much as possible on courses that use LEGO to build robots, in which the relevant robotics theories are taught. The main aim of this thesis is to analyse some of the courses that have been held and to evaluate what the students have gained in comparison to other methods of teaching students in this area.

Sammanfattning

Robotik kurser baserade på LEGO^{TM}

I flera år nu har kurser i ämnet robotik hållits på universitet runt om i världen. Dessa kurser har varit allt ifrån introduktionskurser till specialkurser i robotik. Området robotik är stort och jag skall i denna uppsats koncentrera så mycket som möjligt på kurser som använder LEGO för uppbyggnaden av robotar, till vilken väsentlig teori lärs ut. Huvudmålet för denna uppsats är att analysera några kurser som har hållits och utreda huruvida studenterna har vunnit något på att använda sig av LEGO jämfört med andra undervisningsmaterial och tillvägagångssätt.

Contents

Preface

I would like to thank my mentor at the University of Lund, Jacek Malec, for all support and help he has given me in the working process of this master thesis.

I also would like to thank my family and friends for their help and support.

Håkan Johansson
Lund, July 2001

1 Introduction

In working life today, we have started to see more and more tools that with more or less intelligence help us in our daily work. A well-known area is the use of robots in the car industry, which use robots in the construction of the cars in the factories. Here Japan has been the leader for many years in exploring the ability to use robots in the industry, but the rest of the world is catching up and today none of the large car companies would survive without some kind of robots working in their factories. This use of robots has spread throughout the rest of industry and they are doing things like moving chocolate bars in a candy factory or moving dangerous nuclear waste into stores. One realises that this widespread use of robotic equipment requires well-educated people in the field of robotics as the demands on the robotic equipment increases every year.

A single person can on her own explore a wide area of robotics on the surface or go deep into it thanks to the equipment and knowledge that exists today. For more complicated tasks or if time is of importance, e.g., for a company, building a robot demands teamwork. The teamwork is not only needed in the respective disciplines but also between disciplines in order to fulfil the demands of the market. Universities and companies almost always create as the first point a project that is divided, if needed, into milestones that shall be passed after a certain time and decide who shall do what. In longer tasks one often comes to the conclusion that there is not enough available people, so people from outside the organisation have to be brought in, to fill empty places, so that a project can start.

Courses in physics, computer science, biology, etc., have been held for many years all around the world. Often these courses are specific for their respective fields and do not go outside the boundaries of the subject. Some courses held at universities have been of the broader type that makes use of the knowledge from several different fields; some of these have been in the interdisciplinary area of robotics.

Through the development of robotic technology, it has become easier to hold introductory courses in robotics at a reasonable level. This allowed the students to look at basic ideas without having to risk getting stuck in hardware faults, and not being able to test their ideas or do a special task set for them by their teacher.

A major breakthrough has been done by LEGO[1]-based MIT 6.270 courses and the result has been the LEGO creation, Mindstorms, that for only a couple of thousand Swedish kronor allows people from all age to play and learn basic robotic skills. This makes it possible for schools and universities to afford to hold courses in robotics using LEGO without the need to have personal knowledge in all sub-fields of robotics.

There have been robotics courses held using LEGO and some special hardware like infrared sensors, etc., in the Western world, for about a decade. These courses are what I am interested to investigate in this paper. My aim is to look at some courses held with LEGO at the universities and to analyse the following three elements:

- Whether such LEGO-based courses should be held instead of other robotics courses?
- If yes, how such LEGO-based courses should be held?

[1] LEGO is a trademark of the LEGO Group.

- What do the students really learn by taking these kinds of courses?

This report does not pretend to be exhaustive, although within its specific area (i.e., the courses making use of LEGO) tries to refer to all relevant material. For you who are more interrested in reading more about using robotics in education could start by reading some articles that has been published in a special number of IEEE Intelligent Systems Magazine. I recommend the following articles in this magazine; "Robots and Education" (Murphy [23]), "A Laboratory Course in Behavior-Based Robotics" (Horswill [24]), "Integrating Robotics Research with Undergraduate Education" (Maxwell [25]), "Undergraduate Robotics on a Shoestring" (Sutherland [26]), "Designing and Implementing Hands-On Robotics Labs" (Rosenblatt [27]) and "Using Robot Competitions to Promote Intellectual Development" (Murphy [22]).

2 LEGO-based courses

In this chapter I will present two robotic courses that have been held in the USA and I will then concentrate my discussion on the topic of courses in robotics using LEGO. These short summaries are not in any way to be regarded as giving a complete picture of what has been taught at all these universities but only as an overview of two courses in robotics.

2.1 LEGO-based courses at Massachusetts Institute of Technology

At Massachusetts Institute of Technology (MIT) the design courses 6.270 (Martin [8]), based on LEGO have been held. They have been studied for research purpose by Fred Martin, (Martin [1]), from 1986 to 1992. The courses had a contest at the end to show the different robots that have been created in action. The main task has changed over the years from mainly simulated computer program, started 1986, to a totally LEGO-based, beginning in 1988.

The first course was called “Battle of C-Robots” and here the goal was to find one’s opponent and shoot it down. This later evolved into “XTank” that was of the same style but was much better simulated.

In 1988 something totally different appeared, called “King of the Mountain”. This was the first hardware-based contest that consisted of building a robot out of LEGO that was able to climb to the top of a large paper mountain. In 1990 “Robo-Puck” was created, this was the first contest where three robots at a time competed to gain possession of a physical hockey puck. “Robo-Pong” came the following year, where the robots should transport ping-pong balls onto the other robots’ side of the table. In 1992 “Robo-Cup” was introduced, also based on a ping-pong ball manipulation. Each of the two contesting robots had to extract balls from a feeder and deposit them into their respective miniature soccer goal to get points.

The motivation of this contest-based course was to create a rich learning and innovative environment that would enrich the students in different areas of robotics design.
Beginning with XTank contest, MIT started to use a contest that allowed for multiple solution strategies and a diversity of approaches. Students competing in the XTank contest were allowed to spend a certain amount of money on parts to build their tank. Here they could choose from a variety of things, e.g., size of engine, different kind of armour and weapon, etc. This allowed the students to deploy different strategies based on different purchase pattern.
The following contests were based on physical LEGO and electronics robots, which gave the students much more freedom considering the range of possibilities in which the robots could be configured. Students were allowed to assemble the robots in any way they wanted, only the given robot kit put constraints on this. The contests grew substantially in complexity. The increase in complexity was motivated by development of successively more versatile robot-building materials which were what the students had to play with.

2.1.1 The King of the Mountain Contest

In the first year of using physical materials the MIT gave a simple contest called King of the Mountain. The students had to build a robot that could climb to the top of a hill-like surface. The contest was simple because the given robot-kit that they had created contained of primitive technology and the results reflected more the problems with the given robot-kit than students' abilities and knowledge in robotics. The robots created were not much different from each other.

2.1.2 The Robo-Puck Contest

The Robo-Puck contest was also simple, but the technology used for the robot-kit was of much better quality and richness which gave the students more development possibilities than the previous year's robot-kit had done. In Robo-Puck, three robots played on a nearly two meter in diameter circular rink where they had to gain control of a puck which was initially placed in the centre of the rink. The students managed to invent many different approaches to solve the problem of the contest. The most common design was named puck fetching. Here the design was centred on the possibility to move the robot towards the puck and attempt to capture it. In order to do so several different techniques were developed. One way was to grab the puck by using robot arms that could close around the puck and another way was to bring the puck inside the robot's body and hold it there. A second not so common design was named dual robot puck fetching design. Here the team created a robot that contained two sub-robots, one smaller that was used to fetch the puck as fast as possible while the other large and slower robot went out and captured the smaller robot. Yet another design was named shotgun; it fired towards the puck a claw which it later could retract. This design was able to gain control over the puck much faster than any other robot, but it was not very reliable as sometimes the claw missed the target and the robot had no way of recovering from this, not even detecting the situation. A fourth design was named aggressive design where robots designed this way often moved very slowly but had a great deal of power, which they used against their opponents to make them drop the puck. This design often failed because it did not manage to make the other robot drop the puck or was unable to locate the opponent.

2.1.3 The Robo-Pong Contest

The Robo-Pong contest was designed to support the possibility for students to develop robots in many different ways. The Robo-Puck was based on a single game object, the puck, while the Robo-Pong included multiple game objects with a total of fifteen ping-pong balls. The game was played on a double inclined surface with a plateau in the centre. The goal for the robots was to move the ping-pong balls onto the opponent's side of the table. At the start of a round each robot had six balls placed in the base side of the slope while three balls were located on the centre plateau. The balls placed on the centre plateau were there to encourage a greater range of strategies. The Robo-Pong contest lead to a wider range of robots than had been seen before. The robots could be very simple, from just climbing the hill to hopping and knocking down a ball, and by that hopefully winning, to be able to climb up and down the hill and manoeuvre on both side of the hill. Three basic robots designs were used, the harvester, the eater and the shooter. The harvester's approach was to scoop the balls into some sort of open arms and then push them towards the opponent's side of the playing field. The eater approach was similar to the harvester but it distinguished itself by collecting the balls it found inside its own body. The shooter design fired away every ball it found to the opponent's side; this was without question the most difficult robot to design and build.

2.1.4 The Robo-Cup Contest

When the course leaders designed the Robo-Cup contest they had in mind the flaws they had found in the Robo-Pong. The Robo-Pong contest was not structured enough to help the students do their best and the other major flaw was that the Robo-Pong was vulnerable to a simple aggressive strategy, which could overrun more sophisticated robots. The Robo-Cup tried to make a better balance between different strategies and by so doing made it more difficult for a robot to score a point and hence be able to win. The task was to collect ping-pong balls from ball dispensers and transport them to the appropriate goal. The play area was painted mainly in white with the two opposite corners painted in dark paint, so the robots easily could determine the painted areas. The white area formed a wide path from one goal to the other, and so made it possible to move to their opponent's goal and to try to get a ball into it. On the top of the goals were high-polarised light lamps to guide the robots. The goals themselves were divided into two areas by a protection bar that both stopped robots from going into the goal and encouraged them to shoot their balls. Giving the balls that had been shot one point more than the ones that were rolled encouraged the students to try this possibility.

In order to score a point in the Robo-Cup contest, a robot had to perform a number of specific tasks, moving from the start position to the location of the ball dispenser, depressing the ball request button and transporting or shooting the ball into the goal.
This contest did not make the students develop the diversity of strategies as in the past two years of contests. This was not surprising as Robo-Cup had a relatively high degree of structure when compared to the prior contests.

The Robo-Cup robots could be categorised into two types, the Ball Carriers and the Ball Shooters, where the first one was by far the most dominant. The Ball Carrier caught the ball in a holding chamber and then walked over to the opponent's side where it released the ball into the opponent's goal. The major difference between the robots that applied this design was how many balls they could hold in their chambers. The Ball Shooter was a design where the robots tried to shoot the ball into the opponent's goal instead of moving towards it and by doing so loosing time.

2.2 LEGO course at the Case Western Reserve University

In the Case Western Reserve University (CWRU), in the USA, there is a course in autonomous robotics that has been held every year since 1995. The course is called Autonomous Robotics (AR) and is given in a class of 30 students, divided into groups of three. The aim is that each group has at least one skilled programmer and one skilled

mechanic. ”This course uses robotics to learn out interdisciplinary teamwork-oriented approach to the synthesis and analysis of integrated real-world systems” (Beer [2]).
The course lets students design, build and analyse their own creations, using LEGO bricks and Motorola 68HCll microprocessor as the main building parts. AR, as mentioned, was created in 1995 and grew out of ongoing research on biologically inspired robotics at CWRU and educational technology already developed at MIT.

AR has as a goal to make sure that the students get outside their respective discipline and realise that their discipline is just one of many needed to build a complex system. To succeed they need to work together with people responsible for other parts of the system that they are not experts in. Each group in the AR class gets a table and chairs, a computer and a robot kit that consists of LEGO bricks, sensors, a microprocessor and some other equipment needed to build and program their robotic creation. The different parts of the robot kit are modular, so that students can combine them easily in any way they think is useful and so create things that were not thought about before. The microprocessor is programmed by using a subset of C known as Interactive C (IC) that contains a variety of libraries and support for multitasking. The AR course starts with the students performing a series of structured exercises designed to familiarise them with the different parts of the robot kit. As the microprocessor is the brain of any creation the students wants to build, the course starts by learning how to program the microprocessor using IC. Then they learn what kind of external interface the microprocessor has that they can use to connect motors, LED and a variety of light sensors, touch sensors, bend sensors and break-beam sensors.

The first design project is to build a creation that can move without using wheels or treads. To inspire students into building mobile devices of types they might not think of themselves, a series of mini-lectures is given that provides the necessary technical information about robot control, biological background, on how different kinds of animals and insects move around and other relevant biological information. Students then start with a series of exercises that expose them to most important aspects that have to be dealt with, to be able to manage the final egg hunt competition. The first exercise is to build and program a servo-mounted sensory platform that can orient itself to a polarised light source. The second exercise is to build and program a device that has to be able to see the difference between pastel and black plastic eggs. The purpose of these exercises, that are expected to be building blocks for the final creation to be used in the competition, is to raise important trade-offs between mechanics, electronics and software control.

Each group starts then to build a motorised platform that can continue moving while avoiding obstacles. This task gives the students more knowledge in LEGO geartrain design and even more important, it raises design issues, for example there are many ways to discover obstacles using different kinds of sensors or a combination of them. The students have to find out which is the best and what kind of software complexity it creates. Every group has now the necessary building blocks to create a robot that can orientates it self by light and avoids obstacles. The last thing to do is to incorporate the egg discriminator to produce a robot that can retrieve one pastel egg and deliver it to a nest. The students have now tackled all of the main problems raised by the final competition. Unfortunately their robots often do not perform so well, because the mechanical and software design of each part occurred mainly in isolation from the rest. Some students are unable to finish this last assignment because of a bad construction. The teachers have found that only by letting the students’ experience these difficulties, they can fully appreciate the importance of integrating issue when designing a complete device. When half of the course time has passed the structured exercises end and the teacher hands out the complete rules for the egg hunt competition. The three-student groups are now paired into teams that will create two robots for the final competition. Based on how well the different student groups were in the earlier exercises the teacher tries to balance the teams as well as possible.

The first thing they have to do as a new team is to design a joint strategy for the final competition. The point for this task is to train the three-student groups in developing new group dynamics within the larger team. When each team has completed their design they have to present it to the teacher and be prepared to defend it. After the teacher has approved the team's robot design, the students start to build it for the contest, which will take place four weeks from then.

Four weeks make it possible for the students to totally rebuild parts of the whole robot if faults have been discovered in their construction. The students are able to apply the knowledge they have learned, and by doing so make a better construction that fits their team's overall strategy. During the first half of the course students discover that poorly structured software is a problem for managing the growing complexity of the robot behaviour. Therefore students now start to create more advanced software that often consist of finite state machine architecture, a multiprocessing architecture and/or a subsumption architecture in which higher-level behaviour can override lower-level behaviour.

At the end of the semester a series of test contests that closely simulate the condition of the final competition is held. Participation in these test contests is required because even if students often test their robot by themselves these tests are often too weak because students have a tendency to test a behaviour isolated from the rest, and the robot is often also alone in the test arena. This is in contrast to the final competition when there are four robots in the arena at the same time, the distribution of the eggs is constantly changing and the students are not allowed to enter the arena during a round. These test often lead to disappointing results, which in turn lead to debugging and redesign in the days before the final competition.

The egg hunt competition is the final task for the students where the result of many hours of work will be tested. The competition was designed with the purpose to connect all that has been learned through the semester and to reflect the course educational aims. The educational aims are to encourage as many different strategies and designs as possible. This is done by letting the task contain a mixture of co-operation and competition and be technically challenging so that students would be forced to work together to the best of their abilities and with the materials that were given to them.

The egg hunt competition is a competition, where two teams' robots compete to collect as many pastel plastic eggs as possible, while avoiding plastic eggs painted completely black. The competition takes place in a closed arena with a nest at each end where a grey carpet covers the floor and the walls of the arena are painted completely black. The walls and floor of the nests are painted white. At the back of each nest there is a polarised light beacon, one nest is vertically polarised and the other one horizontally. There is a border around the nest that prevents the eggs from falling in or out of the nest. At the start each team is randomly assigned a nest and the robots are placed in front of their nest. Forty pastel- and ten black plastic eggs are uniformly distributed on the arena. One round lasts for ten minutes and students are not allowed to interfere during this time. At the end of a round each nest is checked and the score is counted by giving one point for each pastel egg and minus four points for each black egg. By this scoring method a robot that cannot see the difference between pastel and black eggs would come up with an average of zero points. If two nests have equal scores they play again and the first to get a positive score wins.

The teams can choose to use one of many strategies in the competition. The simplest strategy is to try to collect as many pastel eggs as possible, another more complicated strategy is to see if the egg is black then move them to the opponents nest and by so doing make them loose four points. Then one can try to block the opponent to put eggs in the own nest, and so on.

3 Educational Issues

When one considers the material for a course in robotics one must make a trade-off between low-level and high-level hardware and software. At a lower level the students learn to understand the basic building blocks which are needed to create robots. At a higher level the teacher takes the overall approach, i.e., he shields the student from complexities and uncertainties often concerning hardware. Instead the teacher encourages a deeper exploration of what a robot is able to do within the limits defined by the given building blocks.

3.1 Hardware

Hardware is always something that is of great concern when one wants to build a robot. Students at robotics courses like the ones at CWRU and MIT face this when creating their robots. "An electrical problem with the hardware of the computer could cause a crash" (Martin [1]). In this case the knowledge about hardware behaviour can be critical for any built robot to have a chance to execute the ideas of its creators. Hardware problems can be caused by faults in the hardware itself, but can also be caused by the software executing on the hardware. One must understand that this kind of knowledge about hardware failures is typically not a part of computer science curricula. In most academic courses students do not have to worry about the reliability of the computer hardware itself. In robotic projects however, the computer boards as well as sensors, motors, etc., have to be taken care of.

3.1.1 Sensors

Sensors are a major source for complex errors of the software. "The student might create an algorithm that would perform properly if given the sensory input that the designer anticipated" (Martin [1]). The software does not create the problem here; instead, this is caused by the fact that sensors do not behave the way the software creators expect. The sensors can account for all, from small to large differences of the expected behaviour. For the students a major sensor error is a lot easier to detect. A sensor fault on a smaller scale or with randomised behaviour is very hard to detect. This in turn leads to the strange behaviour of the robot and students might in some cases have to accept the behaviour and try, if possible, to make changes in the software and remove the influence that the sensor has on the overall behaviour of the robot. According to Martin [1], sensor failures are tricky because of four reasons:

1. Students do not like to think that their algorithm is inadequate because the sensor produces anomalous values;

2. Sensor data tend to be noisy in a fashion that is difficult to model and to compensate;

3. Students do not realise that they do not understand the sensor;

4. It is often the case that a given sensor algorithm mostly works, i.e., most of the time it gives an acceptable performance.

To remove some of the named fault sources a teacher could provide a system that supports debugging facilities, for instance some sort of visual display. The students must be encouraged to study the sensor they are using and be able to track the error of a certain sensor. With a test board, for instance, students can test isolated pieces of their code for the robot that handles a special sensor or, if possible, more than one sensor at the same time.

3.1.2 Communication

A radio communication interface is a good thing to supply the students with, so that it will be possible for them to build a robot system that can communicate using simple messages. This makes it possible to build robots for more complex tasks, where co-operation among two or more robots is needed to fulfil a task, e.g., a football game.

3.1.3 PC versus Robotic System

A robot system often contains a cheap and bare-boned microprocessor that, compared to normal PC microprocessor, has smaller resources. The student has to consider that there is not a huge memory, no mass storage and that the processor runs at a slower cycle time. In a PC system the operating system and the high-level programming system that lay between the hardware and the programmer, uses a big fraction of the given resources. These layers are useful because they make it unnecessary for the programmer to understand details of a special processor implementation. Unfortunately the computational overhead required to maintain such abstraction barriers is usually unacceptable for the simplest microprocessors.

3.1.4 Microcontrollers

An important subclass of microprocessors called microcontroller has become available. “A microcontroller combines the small size, low power consumption, and computational abilities of an inexpensive microprocessor with the signal-processing proficiency of discrete circuits” (Jones [3]). This means that students can get a lot of functions for free, such as serial communication, several timers, analog-to-digital converters, pulse counters, etc. With these functions less work has to be done on making a processor able to talk with sensors, motors, speakers and it makes it simpler to connect the robot device to a PC.

The hardware problems students have when creating robots in courses such as AR are removed in simulator-environment-based course. The problem is that when one removes a part, in this case the robot hardware, it often needs to be replaced by something else, and this often causes other kinds of problems. For software based simulators for example, the learning curve to deal with communication with the simulator and all other kind of problems concerning the simulator interface and design (Heintz [5]) take a lot of the student’s time.

3.2 LEGO

LEGO is a building material which fundamentals are very easy to understand and still has the potential to be put together into a complex system. A teacher cannot expect that all students are familiar with the LEGO concept, so it is a good point to have a first task where students only concentrate on what LEGO can give them. “Their fist design project is to build a motorised LEGO device that can locomote in some way other than using wheels or treads. In addition to gaining experience with building robust, fast, and powerful gear trains using LEGO, this assignment gives student an opportunity to creatively explore the capabilities of the materials in the kit” (Beer [2]). Here the kit refers to be the robot kit that I have described in chapter 2. The idea at CWRU is to make the students explore other ways than wheels as the system that enables the robot to move. “It has led to an amazing variety of LEGO devices that walk, hop, swim, flop, tumble, slither, or inch their way across the floor” (Beer [2]). It seems that these constraints on the building fulfil their purpose.
LEGO as the building material makes it easy to find what is good and bad with certain constructions and reuse this knowledge in future models. “If a LEGO structure often breaks apart at a particular joint, is it more or less apparent which joint is faulty" (Martin [1]). For anyone who wants to know the secrets and the flexibility of the LEGO bricks I recommend “The Art of LEGO Design” written by Fred Martin (Martin [18]).

3.3 Software

In the same way as hardware is a major error source, so is the software running in any robotic creation. When coming to software that runs in a robot, two things are always of major concern, the software language and the load time. Load time means the time it takes to get compiled software files downloaded to the program memory in a robot plus, e.g., erasing the flash memory on a typical microcontroller. Some years ago schools could often just afford to buy microcontrollers where the program memory had to be erased by using ultraviolet light. The erase time was around 15-20 minutes plus the time to move the chip from its holder on the robot to the eraser equipment and back to the robot, or first to the software loader. One soon realises that the load time here has a major impact on the time spent programming. Today electrical erasable program memory has begun to be the one most used. This saves huge amounts of time in the development of the software for the robot.

The robotic software can be divided into two parts, software drivers (SD) and software controllers (SC). A SD is a piece of code that provides a well-defined interface between a hardware device (e.g., sensors) and a program that needs to use the device (Jones [3]). The focus of any robotic course decides the importance of SD knowledge. If the purpose of the course is to give, as one of its goals, a good view of how hardware and software interact, then SD knowledge is one piece of knowledge that has to be taught to the students. If this is not the case then detailed SD knowledge is not important for the course and the teacher should provide the SD necessary to the course. By doing so the teacher avoids students spending too much time on SD that could be used for SC development instead (Martin [1]).

Most of the students initially come up with a lot of complex ideas. They nearly always discard the idea if they find the mechanics too complicated for them to build, but complexity that resides in the software is often overlooked. Many students continue to believe that their robot is capable of very sophisticated behaviours until they try to implement those behaviours in actual code (Martin [4]).

3.4 Teamwork

Teamwork is one very important aspect for the students in a robotic project to accomplish in order to be able to solve the problems presented to them.
The AR course at CWRU had as a goal to teach teamwork between different disciplines. The teams had to be able not only to divide the workload but also manage analysis and constructions of every part of the robot. The students were divided into small teams at the start. Initially the teams worked swiftly and well. LEGO was used as mechanic glue that made everybody (or nearly everybody) in the teams able to work together on a robot creation, instead of one specialised mechanic doing everything. To succeed in the AR course, the students had to develop the skill of working together with people that were working on and were experts in other areas than themselves. This means that they had to go outside their respective discipline, and try to understand how their expertise could work with the expertise of others in order to reach the goal of the project.

3.5 Design

Students often do not see the problems with a design but only with what kind of features a certain design provides. Usually the students approach the design of the robot with a strategy, where they place themselves inside the robot and ask questions such as, what would I do if I were the robot? An example of this design strategy is in the following narrative written by one group of students. “Our strategy at present seems fairly well developed. First, our robot will immediately locate the other robot, roll over to it while lowering the forklift, and try to flip the other robot over. If it can knock it over, or even over the playing field wall, the robot has essentially won, as it can now freely place balls in the goal without interference. If this attempt fails after ten seconds or so, our robot will disengage, roll over to the ball dispenser, park and get balls and shoot them at our goal with a cannon mechanism. Should the cannon prove unreliable, we may have to collect balls and deliver them manually, but having a cannon would be much more nifty, and would avoid a lot of line- or wall-following difficulty” (Martin [4]). Here one can see that the students have not looked at all the aspects of a robot creation. The students have overlooked whether the mechanism aspect is possible and if so, how should the robot perform this task. The problem is that they use their human senses and mental faculties for thinking about the robot control mechanism. Here the problems start popping up for the students when they realise how primitive sensors are and that the robot does not have a bird’s-eye view of the situation as the students themselves have.

Many students underestimate the programming part of the work. It is not until they start designing and coding parts of the control code that they begin to understand what kind of robot behaviours are possible to implement, in the limits of time they have at their disposal. The preferable way is first to get know what kinds of possibilities exist for their robot. By testing how sensors, engines and mechanics work and what kind of expectation one can have, e.g., how reliable sensor data is, or the proportion between the power given to an engine and the final result of a wheel(s) speed.

Students often deceive themselves when it comes to stability of their control program (Martin [4]). For example, if a certain path of the control program has three different parts that have to function together to make it possible for the robot to succeed in doing a complex series of moves, then we can then estimate the possibility for this to succeed if we can estimate the probabilities for each part independently. If the three parts have following probabilities 0.9, 0.8 and 0.9 then the possibility for the overall movement to succeed will be 0.648 ≈ 65%. This will mean that the robot is going to fail one third of the times. Anyone can easily determine the possibility for any path depth by the formula:

$$(X_0 \bullet X_1 \bullet .. \bullet X_{n-2} \bullet X_{n-1}) = \prod_{0}^{n-1} (X_i)$$

3.6 Documentation

Depending on what the main goal for the course documentation is, it can be of more or less value. If the course has the design process in focus then a design notebook would be good (Beer [2]). This design notebook should include not only mechanical and software sketches of the final robot but also all that has been done to reach the course goal. It should be filled in on a daily basis and include initial ideas, early designs on mechanics and software, how testing has been done and notes about problems that have occurred. The notebook could be supported with photographic evidence of the ongoing work (Martin [6]) and interviews with each student, one at the start of the course and one after the project is completed. These interviews can include questions about the students expectations of the course and what it gave them, it should include discussions about the designs used. The teacher can use the data to make improvements in the structure and content of the course.

4 Educational choice

When teaching robotics, is anything gained in conducting the teaching in a more theoretical environment, i.e., combining theory with exercises in simulation environment or is theory better taught in a combination with real-world-based exercises? The question is if one of these is superior or not. I will try to find an answer by putting the two extremes against each other.

4.1 Robo-Cup-based versus LEGO-based course

After DeepBlue won against Garri Kasparov, the world champion in chess in May 1997 (IBM [9]), the AI researchers have been looking for something to replace the chess problem with. It seems today that Robo-Cup is going to meet the needs of AI researches as a new problem to aim for.

Robo-Cup is the name of a problem where two teams of autonomous robots or software agents play a game of football against each other. The aim of Robo-Cup is to have a team of robots play against the human world champions in 2050 and win (Robo-Cup [10]).

In University courses based on Robo-Cup there is often a temptation to use the Robo-Cup simulator as the only tool for the exercises given in the courses. It is not yet common that the students ever leave the Robo-Cup simulator and enter the real world. There exist a few Robo-Cup-LEGO-based courses as for example the “Wall-Ball” (Birk [11]). It combines the Robo-Cup game with the power of LEGO to create a Robo-Cup-based course that leaves the artificial environment of a simulator behind and enters the real world with help of the flexibility and power of LEGO.

Robo-Cup has been used as a teaching tool for undergraduate students in courses in AI at Linköping and Stockholm Universities. The aim for using Robo-Cup simulator instead of real robots were here to let the students to focus more of their time on software agents, Heintz [13]. They found out that the poorly documented code base of Robo-Cup together with the students problem of understand the AI domain created a huge problem. To solve the Robo-Cup issue Heintz developed a software library called RoboSoc. RoboSoc is meant to provide the functionality needed for the courses and be well documented to make it easier for the students to do their assignments. The courses have problem-based-learning (PBL) as a centre stone and by so forcing the students to be more active in their own learning process. They had found by experience out that Robo-Cup is well suited as a tool in learning AI in form of PBL. The students formed teams of two or three people. For the students the challenge has been in trying to solve the problem in any way that suited them and not be lead into a one correct path by their teacher.

The knowledge that there will be a competition between the student teams creates in itself an excitement among the students and makes them put in more effort into their work. This has lead the students to get faster a deep understanding of the complexity of their solution in the assignments. Another major problem is that the students often lack good knowledge in non-AI areas such as multi-threading and network programming needed to solve the AI assignments.

RoboSoc solved a lot of these problems and the teams developed with help of RoboSoc were able to compete with the midfield teams from the 1999 World Cup in Robo-Cup.

The RoboSoc together with the changes declared by Noda (Noda [14]) might just move the Robo-Cup simulator from a wild-grown piece of software towards a slim and easy understood and maintained simulator that would speed up the research and teaching processes based on Robo-Cup.

In 1999 Robo-Cup Junior was first used in Stockholm to teach robotics to 9-14 years old children (Lund [15]). This was a good opportunity for teachers to see how to use Robo-Cup below high-schools level. It gave also something for the developers that maintained the Robo-Cup Junior concept on how to improve it so that it can became easier for a teacher in the future to use Robo-Cup Junior in the school. Here we once more see the combination of LEGO and Robo-Cup concept and that they are not two things that exclude each other, instead they together strengthen each other's usefulness in education. A problem with using Robo-Cup Junior in below or at high-school level is that it is not easily integrated into any existing subject (Kummeneje [12]) and that it puts tremendous demand on the teacher.

4.2 Khepera versus LEGO-based course

To develop robots with Mindstorms™ is very easy and cheap. Using Mindstorms™ have made it possible for non-physicist and non-computer scientists to enter the robotic area (Baum [17]). Mindstorms™ has some drawbacks when it comes to the power of the Robotic Command Explorer (RCX) bricks memory capacity and flexibility of user program (Kumar [U1]) which can be a problem for very advanced robotic courses. Regarding the programming and downloading of the final firmware to a robot there is no major difference between a Khepera- or Mindstorms™-based robot. The major difference lays in the design flexibility that the LEGO bricks give.

Khepera is a module-based robot, i.e., it is build up in layers that can do different tasks (LAMI [19]). The first layer is always the wheel and engine module that enables Khepera to move around in the physical world. After that the following layers are up to the builder to choose, there are already some modules to choose from such as for example a radio module. Khepera is a lot more expensive to buy than Mindstorms™, around 18 000 Swedish Kronor (K-TEAM [21]) per unit. The software interface, which comes with Khepera, makes the robot difficult to use (Harlan [20]). A very nice quality that Khepera has is the possibility to run the robot control program in the host computer while able to query the sensors of the robot and controlling everything that would be possible if the program was in fact running in the robot.

5 Conclusion

In this chapter I summarise the knowledge from all material that I have used and extract that part which answers my questions, which I stated in the first chapter, about LEGO-based courses.

I cannot find any reason why LEGO-based courses would be supreme compared to other types of courses. But what I found was that LEGO gives the students a rich and flexible material, which they can use in their design of their robotic creation. The main reason why one should be using LEGO is:

- The incredible flexibility that LEGO bricks gives the students in the design and build phase of their robot.
- The price for LEGO kits such as Mindstorms^{TM} are very inexpensive compared to other kits. This even gives the possibility to let a team of students buy their own kit for the course if necessary.
- The fact that a big toy company, as LEGO, together with MIT are producing and further developing these kits, gives stability to using them in courses knowing that there will exist kits to buy also in the future with even more processing power and features, and that the new kits will be backward compatible with the old ones.

LEGO-based courses should be held in a way that permits the students to both grasp and to use the power of LEGO. To allow this to occur a teacher should think of following things when thinking of creating a LEGO-based course:

- Force the students to go beyond wheels and caterpillar treads when they design the mobility part of their robot.
- LEGO seems on the surface very simple but it hides a lot of power that is not common knowledge. Because of this a good point is to let the students read a paper such as “The Art of LEGO Design” to really understand the fundamentals of LEGO.
- Robotic courses are a perfect opportunity to teach about teamwork and the way things are done in the real life.
- Design documentation gives the students the possibility to see why they did something in a certain way and to be able to go back a number of steps in their design. This also gives the possibility to study each robotic design together with the whole class in the end of course.

The lack of interconnecting courses, i.e., courses that have used both LEGO and some other building block for their courses make it very hard to tell if LEGO is supreme. Those who have used something else often have migrated from simulator-based courses towards hardware-based courses in several steps and the courses have changed a lot under the way.

Bibliography

Reference

[1] Fred Garth Martin, "Circuits to Control: Learning Engineering by Designing LEGO Robots", 1994, Ph.D. Thesis, Massachusetts Institute of Technology

[2] Randall D. Beer, Hillel J. Chiel and Richard F. Drushel, "Using Autonomous Robotics to Teach Science and Engineering", 1999, Communications of the ACM, Vol. 42, No. 6

[3] Joseph L. Jones, Bruce A. Seiger and Anita M. Flynn, "Mobile Robots Inspiration to Implementation", A K Peters, Second Edition, 1999

[4] Fred G. Martin, "Ideal and Real Systems: Study of Notions of Control in Undergraduates Who Design Robots", MIT, Technical Report, 1994

[5] Fredrik Heintz, Johan Kummeneje, "Simulated RoboCup in University Undergraduate Education", Department of Computer and Information Science at Linköping University and Department of Computer and System Sciences at Stockholm University and the Royal Institute of Technology, Technical Report, 2000.

[6] Fred G. Martin, "Kids Learning Engineering Science Using LEGO and the Programmable Brick", MIT, Technical Report, 1996

[7] LEGO, http://www.lego.com/info/businessarea.asp, 2001-06-24

[8] Fred Martin, "The 6.270 Robot Builder's Guide", MIT, 1992, http://fredm.www.media.mit.edu/people/fredm/projects/6270/, 2001-06-24

[9] IBM, http://www.research.ibm.com/deepblue/home/may11/story_1.html, 2001-06-24

[10] Robo-Cup, http://www.robocup.org/, 2001-06-24

[11] Andreas Birk, http://arti.vub.ac.be/~cyrano/COURSES/wall_ball.html, 2001-06-24

[12] Johan Kummeneje, "RoboCup as a Means to Research, Education and Dissemination" Department of Computer and Systems Sciences Stockholm University and the Royal Institute of Technology, Licentiate Thesis, February 12 2001

[13] Fredrik Heintz, Johan Kummeneje and Paul Scerri, "Using Simulated RoboCup to Teach AI in Undergraduate Education", Department of Computer and Information Science at Linköping University and Department of Computer and System Sciences at Stockholm University and the Royal Institute of Technology.

[14] Itsuki Noda, “Framework of Distributed Simulation Systems for Multi-agent Environment”, Electrotechnical Laboratory, Tsukuba 305 Japan

[15] Henrik Hautop Lund, http://www.daimi.au.dk/~hhl/RoboCupJr.html, 2001-06-24

[16] Adam Currie, “THE HISTORY OF ROBOTICS”, http://cache.ucr.edu/~currie/roboadam.htm, 2001-06-24

[17] Dave Baum’s, Definitive Guide to LEGO Mindstorms, APress, 2000

[18] Fred G. Martin, “The Art of LEGO Design”, MIT, Technical Report, 1995

[19] LAMI (Laboratorie de Microinformatique), http://lamiwww.epfl.ch/Khepera/index.html, 2001-06-25

[20] Robert M. Harlan, David B. Levine, Shelly McClarigan, “The Khepera Robot and the k-Robot Class: A Platform for Introducing Robotics in the Undergraduate Curriculum”, Computer Science Department, St. Bonaventure University

[21] K-TEAM, http://www.k-team.com/, 2001-06-25

[22] Robin R. Murphy, "Using Robot Competitions to Promote Intellectual Development", AI Magazine, Vol 21, no 1, Spring, 2000, pp. 77-90

[23] Robin R. Murphy, "Robots and Education", IEEE Intelligent Systems Magazine, November/December 2000, pp.14-15

[24] Ian Horswill, "A Laboratory Course in Behavior-Based Robotics", IEEE Intelligent Systems Magazine, November/December 2000, pp. 16-21

[25] Bruce A. Maxwell and Lisa A. Meeden, "Integrating Robotics Research with Undergraduate Education", IEEE Intelligent Systems Magazine, November/December 2000, pp. 22-27

[26] Karen T. Sutherland, "Undergraduate Robotics on a Shoestring", IEEE Intelligent Systems Magazine, November/December 2000, pp. 28-31

[27] Michael Rosenblatt and Howie Choset, "Designing and Implementing Hands-On Robotics Labs", IEEE Intelligent Systems Magazine, November/December 2000, pp. 32-39

Readings

- Tim Portnoff, “Objektorienterad robotstyrning och simulering i realtid”, 2000, Department of Computer Science, Lund University

- Henrik Lernmark, “Objektorienterad Robotsimulator”, 2000, Department of Computer Science, Lund University

- Charlotte Magnusson, “Artificiell Intelligens – om mötet mellan mänskliga behov och tekniska möjligheter”, 1994, SIH , ISBN-91-88452-166-6

- Fred G. Martin, Deirdre Butler and Wanda M. Gleason, “Design, Story-Telling, and Robots in Irish Primary Education”, IEEE Systems, Man and Cybernetics conference, October 8-11, 2000, Nashville, Tennessee

- Deirdre Butler, Fred Martin, Wanda Gleason, “Empowering Minds by Taking Control: Developing Teachers’ Technology Fluency with LEGO Mindstorms”, Presented at the Society for Information Technology and Teacher Education (SITE) conference, January 2000, San Diego, California USA.

Unpublished papers

[U1] Amruth N. Kumar, “Using LEGO Mindstorms Robots in a Traditional AI Course”, 2001, Presented at AAAI Spring Symposium on Robots and Education, Stanford, 2001

Robot Links

- RHEX, the compliant hexapod robot
  A six legged robot for tough terrain
  http://ai.eecs.umich.edu/RHex/
  2001-06-24

- BARt-UH, Bipedal Autonomous Robot - Universität Hannover
  Two legged robot for service purpose
  http://www.irt.uni-hannover.de/~biped/
  2001-06-24

- Aibo
  A robot dog from Sony
  http://www.aibo.com/
  2001-06-24

- Khepera
  A wheel based robot that has a modular construction
  http://lamiwww.epfl.ch/Khepera/index.html
  2001-06-24

Miscellaneous Robot Courses

- MIT
  Computational Laboratory in Cognitive Science
  http://www.ai.mit.edu/projects/cbcl/courses/course9.39/
  2001-06-24

- Det Naturvidenskabelige Fakultet Aarhus Universitet
  Adaptive Robots
  http://www.nat.au.dk/nf/studier/studordn/uk/aktuel/datalogi/c_mod/ada_robots.htm
  2001-06-24

- Khepera controlled through Internet
  http://www.otc.kyit.edu.tw/ICEE1999/papers/427/427.htm
  2001-06-24

- Umeå Universitet
  Artificiell Intelligens II
  http://www.cs.umu.se/kurser/TDBD93/VT01/
  2001-06-24

- Technische Universität München
  Praktikum Echtzeitsysteme
  http://www6.informatik.tu-muenchen.de/lehre/prakt/khepera/khepera.html
  2001-06-24

- University of Florida/Machine Intelligence Laboratory (MIL)
  http://mil.ufl.edu/imdl/
  2001-06-24
  EEL5666 - Intelligent Machines Design Lab (Design, simulation, fabrication, assembly, and testing of intelligent robotic machines)
  EEL6668 - Intelligent Robot Systems (Overview of techniques available for vision in robotics)
  EEL6935 - Machine Learning in Robotics (I and II) (Neural networks, locally weighted learning, stochastic optimization, Bayesian learning, Expectation-Maximization algorithm, Markov chains and hidden Markov models, Markov Decision Processes (reinforcement learning).)

- Stanford University
  http://robotics.stanford.edu/groups/manips/courses/index.html
  2001-06-24
  CS223A: Introduction to Robotics (winter)
  CS225: Experimental Robotics (spring)
  CS327A: Advanced Robotic Manipulation (spring)
  CS528: Broad Area Colloquium for AI, Geometry, Graphics, Robotics and Vision (autumn)

- Institutionen för Data och Systemvetenskap, SU/KTH
  http://www.dsv.su.se/~mab/7v99/Index.html
  2001-06-24

- University of South Carolina
  http://jmvidal.ece.sc.edu/822/
  2001-06-24

- Högskolan i Halmstad
  http://www.hh.se/staff/ulho/mekatronik/mekhem.html
  2001-07-10

URLs to some LEGO-based courses

- University of Southern California
  http://www-scf.usc.edu/~csci445/
  2001-06-24

- Villanova University Computing Sciences
  http://www.csc.villanova.edu/~klassner/csc4500/
  2001-06-24

- Case Western Reserve University
  http://www.eecs.cwru.edu/courses/lego375/
  2001-06-24

- MIT
  http://web.mit.edu/afs/athena.mit.edu/course/6/6.270/www/home.html
  2001-06-24

- Indiana University
  http://www.indiana.edu/~legobots/tectrac/index.html
  2001-06-24

- The University of New Mexico
  http://wotan.me.unm.edu/~starr/lego/lego.html
  2001-06-24

Appendices

A.1. Abbreviations

AR	Autonomous Robotics
CWRU	Case Western Reserve University
LEGO	LEg GOdt (meaning “play well”) [7]
MIT	Massachusetts Institute of Technology
PBL	Problem Based Learning
SC	Software Controller
SD	Software Driver
RCX	Robotic Command Explorer

A.2. Robot History

In the first half of the 20:th century the Czech playwrighter Karel Capek invented the word “robot” (Currie [16]). The word robot was created from the Czech word for forced labour or serf. The use of the word robot was first introduced in his play R.U.R. (Rossum’s Universal Robots) which opened in Prague in January 1921. It was as great success and copies of the play soon opened all around Europe and USA.
The Russian-born American scientist and writer Isaac Asimov created the term “robotics” to refer to the study and use of robots. Isaac Asimov first used the word robotics in a short story called “Run-around” in 1942. Together with several other short stories it became the book “I, Robot” that was published 1950. Here Asimov also proposed three robot laws and later he added a zeroth law.

- Law Zero:
- A robot may not injure humanity, or, through inaction, allow humanity to come to harm.
- Law One:
- A robot may not injure a human being, or, through inaction, allow a human being to come to harm, unless this would violate a higher order law.
- Law Two:
- A robot must obey orders given it by human beings, except where such orders would conflict with a higher order law.

- Law Three:

  A robot must protect its own existence as long as such protection does not conflict with a higher order law.

George Deval and Joe Engelberger developed in the late 50’s and early 60’s the first industrial modern robot called “Unimate”. Engelberger formed a company called “Unimation” and was the first to sell robots on the market. Due to that Engelberger has been called the “father of robotics”.