# ROBOTICS EDUCATION FOR COMPUTER SCIENCE STUDENTS

Vesna Kirandziska, Nevena Ackovska
Institute of Informatics, Faculty of Natural Sciences and Mathematics
Ss. Cyril and Methodius University
Skopje, Macedonia

ABSTRACT

Robotics is an emerging field in the area of intelligent systems. Design, construction and operation of robots are crucial parts of Robotics and this should be the domain of teaching Robotics for students that are introduced to this field for the first time. This paper proposes an approach of teaching Robotics to computer science students. Robotic systems should be taught in four different abstraction levels: a mathematical abstraction level, a model abstraction level and a hardware abstraction level and the hardware itself. Each level is based on the previous, but each shows a different perspective of robotic systems. Mathematical programs, modelling tools and software for controlling real robotic systems, as well as the robot hardware itself, are the tools students have to use while studying Robotics.

*Keywords* – computer science; education; robotics.

## I. INTRODUCTION

Robotics is a relatively new, ongoing and upgrading topic in computer science, but the idea of robots dates from many centuries BC. Mechanical man and artificial beings are mentioned in Greek myths (Myth of Hephaestus). Later, in Egypt, citizens turn for advice to oracles, statues with priests hidden inside. Back than dates the idea of thinking machines.

Today robots are manufactured intensively and are in use for entertainment, commerce, industry, military and advanced research.

Robotics is included in education, especially the engineering and computer science education. However, robotics is thought to high school pupils [5] and even to preschool children. There are many Robotics competitions: "Robofest" [2], "RobotCombat" [3] etc. Some of them, like "First Lego League" [4] are for children aged 9-14.

The future of Robotics is in intelligent robots that in some particular area help humans of just copy the "perfect" humanistic behaviour. Robots being our friends, robotic mechanisms healing our body, interacting with robots, are just part of the future human-robotic society we may witness.

We can conclude that Robotics, as a computer science field, is very interesting and still evolving in every since. There is much more to be done. That is one reason why students should learn Robotics.

The course Robotics at the Institute of Informatics, at the Faculty of Natural Sciences and Mathematics in Skopje, Macedonia is just a natural consequence of the trend in the world. In addition, three Lynxmotion robots located in the institute's laboratory (Fig. 1), made this course possible. This course was held the previous two years and this paper presents a slightly different approach on teaching Robotics for computer science students. The objective of the proposed approach is to give students introduction, solid ground and good feeling for Robotics itself.

Figure 1: Lynxmotion [6] robots in our laboratory (Robotic arm, Hexapod and Biped Scout).

## II. ABOUT THE COURSE

Robotics course at the Institute of Informatics has the following weekly distribution of the teaching material: 2 classes per week for the theoretical lectures, 1 for theoretical exercises, and 2 classes per week for practical laboratory exercises. It is recommended that students form third or fourth year take this course.

As a computer science topic, the task of Robotics is to design, construct and operate or manipulate robotic systems. Because this course is aimed for beginners - students with no previous knowledge in Robotics, the scope of the theoretical exercises of this course is narrowed to serial manipulators. Practically, students will have a chance to manipulate more complex robotic system.

Serial manipulators are very simple, but practical robotic systems. They are the most common industrial robots. A serial manipulator, or just manipulator, consists of a number of rigid links connected with joints. There are revolute joints that can revolve around their axis and prismatic joints that can be extended or shortened, accordingly. These manipulators usually end with an effector which could be a hand's grip, for example.

This part of the course syllabus contains definition and kinematics of serial manipulators as well as controlling robotic systems in general.

Kinematics is a brunch of classical mechanics that describes the motion of bodies and systems without consideration of forces that cause the motion. One basic task of serial manipulators is moving end effectors to some specific position and orientation. This task is in the scope of kinematics. Motion that depends on the environment and intelligent motion are not in the scope of this course, but the knowledge gained in this course is necessary for accomplishing more difficult robotic tasks.

Once students learn about manipulators, they could extend their knowledge to more complex robotic systems. Still, basis is always the most important.

## III. Background Knowledge

Students who are willing to take Robotics have to have background knowledge in some areas. Mathematics is the base for all natural sciences, so students have to know some mathematics in order to study Robotics. Indeed, algebra, including definition of matrices, matrix operations and inverse matrices are important in learning Robotics. We will see in the following sections why algebra is so important. Students should also have basic knowledge in trigonometry, as well as some basic geometry, especially analytical geometry.

Other than mathematics, Robotics as a multidisciplinary field includes physic too. Robots are bodies and as such they are subject to physical lows. This course will mainly be concerned with the kinematics of a robotic system.

Servo-motors and microprocessors are electronic components from which one robotic body consist of. Basic in electronics is quite important when dealing with real robots, especially when constructing one. Mechanics can be considered as crucial part of Robotics because robots are built of mechanical components. When constructing robots this knowledge is really important as a student will find out while working with real robots.

Other brunches of computer science like Programming languages and Design are important for students who plan to take Robotics. Indeed, programming is needed as a mean for manipulating robots and design is crucial for modelling robotic systems, as we will see later on.

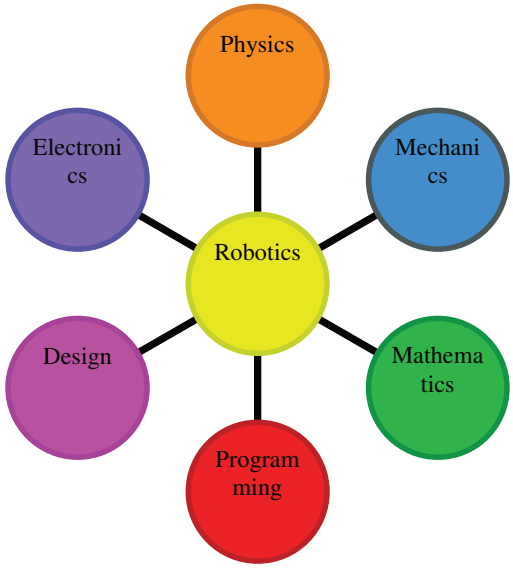


Figure 2: Schematic representation of areas that affect Robotics.

All things considered, as illustrated on Fig. 2, Robotics is affected by many different fields and this only shows how multidisciplinary Robotics is. The colourful schema describes the diversity of Robotics.

## IV. Robotics Education for Computer Science Students

Robotics is a field that has to be studied systematically, step by step. This paper proposes having four levels of abstraction for learning robotic systems in general. These abstraction levels will take the computer science students from theory to practice. They are illustrated on Fig. 3.

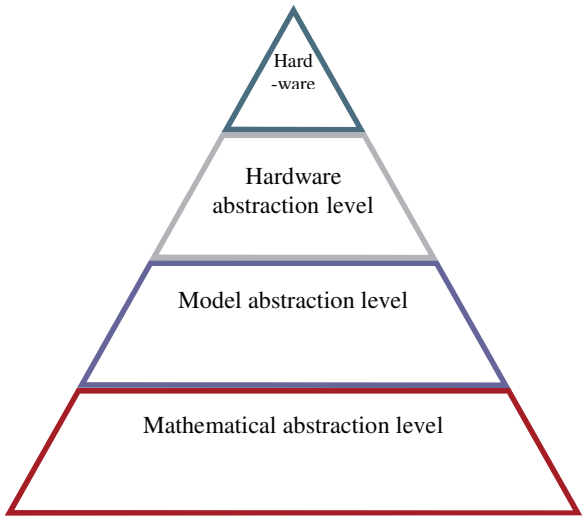


Figure 3: Abstraction levels.

Upon reaching the third year of studies, the Computer science studens at the Institute of Informatocs have alredy gained solid mathematical knowledge, as well as good programming skills. This is used in the way we teach Robotics. We start with the most abstract level, end follow to the most practical and most interesting level. As shown on Fig. 3 the mathematical abstraction level is the base level which enables the other levels. Following the mathematical abstraction level, the more practical levels of abstraction follow: model abstraction level and hardware abstraction level. In the highest level of abstraction – hardware itself - students will have to learn how to design, construct and operate with real robotic systems. Examples of real robotic systems we use in this course were shown on Fig. 1.

In the following sections these four levels of abstraction will be described in detail. More specifically we will describe the domain that will be taught in each abstraction, as well as the methodology and tools needed.

### A. *Mathematical abstraction level*

In this abstraction level a robot is described using mathematical definition. Kinematics problems are described as mathematical problems. We will describe shortly what students should learn from this abstraction and how it should be presented to them.

*1) Domain*

Unambiguously, a robotic serial manipulator can be described with a DH-notation (Denavit-Hartenberg notation). By this notation, each robotic joint as well as the end effector, is described with four parameters so the manipulator itself is described by list of four parameters: $DH=\{(a_i,\alpha_i,d_i,\theta_i),i\in 1..m\}$. Some of these parameters are variables. These variables make the input coordinates which form a so called vector of input coordinates: $\rho=(\rho_1 \dots \rho_k)$. The vector of output coordinates r=(x y z) contains the coordinates of the end effector in a referent coordinate system in the three-dimensional space. The velocity and acceleration of the coordinates in space are given by $r'$ ( $r''$) and $\rho'$ ($\rho''$) accordingly for output and input coordinates. These notations are part of the mathematical definition of serial manipulators.

Kinematics problems considered in this course are: 1) direct and inverse position kinemics 2) direct and inverse velocity kinematics 3) direct and inverse acceleration kinematics. Direct kinematics considers the problems of calculating output parameters given the input ones. Inverse kinematics is concerned with the reverse problem i.e. calculating input parameters by taking output parameters as input.

These problems, mathematically, are given by matrix equations. For example, if we consider direct position kinematics, matrix equations can be extended to a system of 3 equations with k parameters and 3 variables. These equations are trigonometric equations in case of position kinematics. In case of velocity and acceleration these equations are differential equations of first and second order, accordingly.

Given the mathematical definition and formulation of one kinematics problem, a computer science student, which has great background mathematical knowledge, will be able to solve this problem. Others who have less mathematical knowledge could use some mathematical tools. For mathematical calculations, mathematical software like Mathematica 5 can be used. Using this software matrix operations, trigonometry operations, differential calculus and solving equations can be performed.

*2) Methodology*

Definition of all terms must be taught in the beginning. These definitions should be precise and concise. In order students to learn how to solve kinematics problems by themselves, a lot of practical examples in Mathematica 5 should be given to them.

The laboratory exercises consist of mathematical problems as described in the lectures and finding a solution for these problems using Mathematica.

*B. Model abstraction level*

Taking on a higher level of abstraction we can view robots i.e. robot manipulators as models. Models of serial manipulators can help students visualize manipulators or robots in general. Real robots are quite expensive and as we showed on Fig. 1 we have just few of them on our labs. That is why models are so important. Models can be used as a starting point for building new robots or examining some properties of real robots whose real-life operation is expensive and hard to explore.

Because models are so important, students should learn how to design a robot and also how to simulate some specific tasks. From a simulation, a conclusions related to real robotic systems can be made.. Building models and making simulations of other robotic systems is much more complex to do and should be omitted in a Robotics course for beginners.

*1) Software*

In Robotics course we will use modelling software called Rhinoceros [1]. Rhinoceros can be used for modelling all kind of different objects. As an extension to Rhinoceros there are available plug-ins for rendering and animation. Another great feature about this product is that users – programmers can make their own plug-in using Microsoft Visual C++ 2005. Installation of Rhino SDK (Software Development Kit) is needed for that feature. We will use Rhino SDK 4.0. When installation finishes users can create projects of type: Rhino Plug-in or Rhino-DLL.

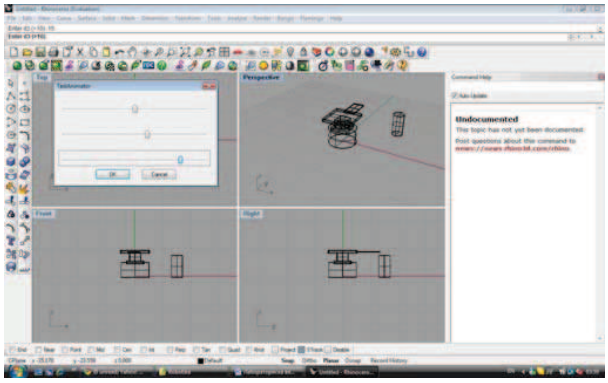

Figure 4: Rhinoceros – a modelling tool.

*2) Domain*

First, student should learn the offered software. A window of this software is shown on Fig. 4. Rhinoceros as a modelling software has huge functionality. The sense and knowledge of design is really helpful in learning Rhinoceros. Computer science student are more familiar with programming. That is why making plug-ins for Rhinoceros in Microsoft Visual C++ is presented to students. All of the options and operations in Rhinoceros can be programmed in C++. Students should learn how to build a plug-in for constructing a serial manipulator. In order to program a serial manipulator or simulate some manipulator motion the mathematical abstraction is used.

More important is students to understand a robotic manipulator model. For a given model, students should examine its properties. In order to do that, students have to learn how to represent manipulator models in the DH-notation. Given that notation we saw that we can solve different kinematics problems. First, we could make a robot by having the input coordinates only. Second, using the

mathematical solution of the inverse kinematics problem we can model a robotic system that is positioned as given by the output coordinates.

*3) Methodology*

A lot of example models made in Rhinoceros are presented, so as to give students insight of what a model is and what it can do. Homework about mathematically representing a model should be given to students. Projects for extra points can consist of modelling and simulating a robotic manipulator given some input data. These projects can be done as plug-ins for Rhino in Microsoft Visual C++.

### C. Hardware abstraction level

In this abstraction level a robot is viewed as a microcontroller that takes input messages from a computer. A kinematics problem can be viewed as a problem of finding a sequence of messages that need to be sent to a microcontroller, so the problem will be solved.

*1) Software and hardware*

In the Robotics course we have two types of microcontrollers. The first one is Mimi SSC II and it is shown on Fig. 5. The second is SSC-32. The first microcontroller can control 8 servo-motors and is used by the robotic arm. SSC-32 can have maximum 32 servo-motors and is used with the Biped Scout and Hexapod.

In order the communication computer-microcontroller to work, charging batteries, power simply for the servo-motors, and serial cable for connecting a computer to the microcontroller are needed.

Messages can be written and sent in C or in Java, or any other programming language, but there must be some libraries or packages for communicating through a serial port. Besides programming languages there are specialized applications like Visual Sequencer or LynxTerm for sending messages to a microcontroller.

*2) Domain*

What students must learn is about the computer-microcontroller communication. For each microcontroller students have to learn about the messages that have to be received. Mini SSC II takes 3 bytes as one message. The first byte is a signalization byte (255) and the other two describe the servo-motor and his new position.

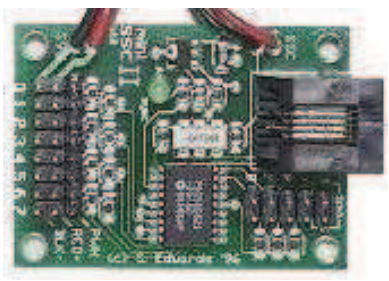

Figure 5: Microcontroller - Mini SSC II.

SSC-32 takes a more complex message as input. In that message we can tell the new position of each servo-motor and we can tell the velocity or time needed so that that operation is complete. The difference between these microcontrollers is that the first one can remember more messages. Sending message when another is being processed in case of the second microcontroller will cause the second one not to be executed.

When manipulating with the robotic arm i.e. with the Mini SSC II microcontroller, mathematical solution of kinematics problems or a simulated model can be used for constructing the right messages. The manipulation of the other two robots - the Biped Scout and the Hexapod, primarily consists of trial and error.

Students will know how to send messages through a serial port to a microcontroller controlling a robot. They will gain programming skills for computer-microcontroller communication.

### D. Hardware

As we said earlier we have three robots from Lynxmotion in our laboratory. They are robotic hand, walking biped scout and hexapod. In this abstraction level robots are seen as physical and mechanical objects. Actually, this abstraction level is the level where students work with the real robots in real environments.

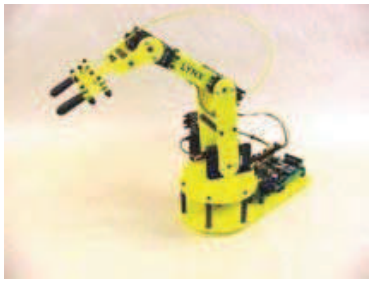

Figure 6: Robotic Arm manipulator: Lynx 5.

*1) Domain*

The top layer on Fig. 3 is the hardware. Students have to incorporate the knowledge gained from previous abstraction levels and use it to get a final product – robot movement. Students will see that mathematics works perfectly fine with models; but this is not true in real systems. A lot of errors arise from the ideal modelling case. Battery dead, microcontroller down, wrong wire connection are part of the problems that students will come across.

*2) Methodology*

Many hours working with robots are needed for the students to learn how to manipulate them. A fully equipped laboratory should be open for students how want to experiment with the robots. Students should be given a specific problem as: 1) operate the robotic hand so it moves from point A to point B with the end effector being horizontal all the time, or 2) do some stable motion on the walking

robot, like turning left. Students should be given 3-4 weeks time for solving the given problems. The maximum points won on this project should be 30 percentages of the total course points.

## V. FUTURE WORK

Robotics is a field that can be more explored in the future. Improvements are needed not just in the mechanical aspect of the robotic system, but in the intelligent aspect of the system which should incorporate the robotic system together with the environment. The interaction and the intelligent behaviour of a robotic system based on the input from the environment represent a higher level of abstraction. So, the proposed levels of abstraction could be supplemented in the future for teaching an upgraded Robotics course.

## VI. CONCLUSION

The objective of teaching Robotics is for students to understand the operation of robotic systems through the mathematical abstraction, to design robotic systems that can operate in a desired way by making models and simulating their operation and to construct robotic systems based on the best model and to deal with problems that arise from working with real hardware. The approach described earlier follows these guidelines.

Computer science (CS) students are much more comfortable with the software than the hardware part of the course. Using different computational tools and computer visualization of the basic problems of Robotics the Robotics course becomes more approachable for the CS students. This allows learning hardware, to become significantly more understandable for the CS students. Students that will gain great knowledge in this area could continue their research in this area and contribute to it later.

## VII. REFERENCES

[1] McNeel North America. Modeling tools for designers-Rhinoceros. Available from: http://www.rhino3d.com (2011)

[2] Lawrence Technological University Robofest. Available from: http://www.robofest.net (2011)

[3] Robot Combat. Available from: http://www.robotcombat.com (2011)

[4] FIRST. Available from: http://www.usfirst.org/roboticsprograms/ (2011)

[5] The Tech Museum of Innovation. Available from: http://www.thetech.org/robotics/activities/ (2011)

[6] Lynxmotion, Inc. Available from: http://www.lynxmotion.com (2011)