Project Based Learning of Robot Control and Programming

G. López-Nicolás[1], A. Romeo[2], J. J. Guerrero[3]

[1~3] Dept. Informática e Ingeniería de Sistemas, University of Zaragoza
C/ María de Luna 1, 50018, Zaragoza, Spain.

gonlope@unizar.es[1], romeo@unizar.es[2], jguerrer@unizar.es[3]

Abstract

: We describe an active learning experience in the field of Robotics in the context of a degree on Industrial Engineering. It is well known that the field of Robotics involves many disciplines taught in higher education, as mechanics, electronics or computer science. Two aspects are mainly treated in a course of Industrial Robotics for an engineering degree: The modeling of the robot and the practice to obtain programming skills for solving robotic tasks. The robot modeling has a high geometric and mathematical complexity, which is added to the difficulties (time and budget cost) of the students for working with real robots. These issues can be benefited by simulation tools. For several years we have conducted a learning project-based experience in which students must model and solve kinematics for a commercial robot, design some aspects of its control system and program the robot in an industrial application. Simulation tools are essential for student's motivation and understanding of the problem, and they give good learning results. We present the active learning experience and the simulation tools developed for this project-based course.

Introduction

In a broad perspective Industrial Robotics is an inherently cross-disciplinary subject whose in-depth knowledge involves a variety of tasks such as modeling, design, simulation, control, optimization, and performance evaluation. In addition, successful application of industrial robotics involves the integration of various tools from related disciplines. The multidisciplinary issues involved, like kinematics, dynamics, electronics or computer science, as well as the complexity of the theoretical concepts required make this discipline difficult for students to get deep understanding with classical teaching.

The Bologna process started with the signing in 1999 of the Bologna declaration by Ministers of Education from European countries. The purpose of this process is to create the European higher education area by making academic degree standards and quality assurance standards more comparable and compatible throughout Europe. The new model comes closer to the North American and Japanese systems. It gives greater weight to practical training and to intensive research projects [1], [2], [3]. Thus, the student has to play the leading role of his learning, basing the learning process in active methodologies and student autonomous work to the detriment of traditional lecture methods, in which professors talk and students listen. In essence, students should do more than just listen. They should read, write, discuss, or be engaged in solving problems. Specially, students must engage in such higher-order thinking tasks as analysis, synthesis, and evaluation to be actively involved. Strategies promoting active learning are defined as instructional activities involving students in doing things and thinking about what they are doing [4]. This is the framework of the project-based learning activity proposed.

In this paper we describe an active learning experience in the field of Robotics in the context of a degree on Industrial Engineering. Two aspects are mainly treated in a course of Industrial Robotics for an engineering degree: The modeling of the robot for the design of its control system, and the practice to obtain programming skills using specific robot languages. Both complementary learning aspects can be benefited by simulation tools. The robot modeling has a high geometric and mathematical complexity, which is added to the difficulties in terms of time and budget required for programming industrial tasks with real robots. The proposed activity has been conducted in two different courses in engineering degrees at the Computer science and system engineering Department at the University of Zaragoza: Control and Programming of Robots (four-month long course) and Industrial Robotics (full year course).

For several years we have proposed this learning project-based experience in which students must model and solve kinematics for a commercial model of robot manipulator, design some aspects of its control system and program the robot in an industrial application. Simulation tools, developed for this specific project, are essential for student's motivation and understanding of the problem.

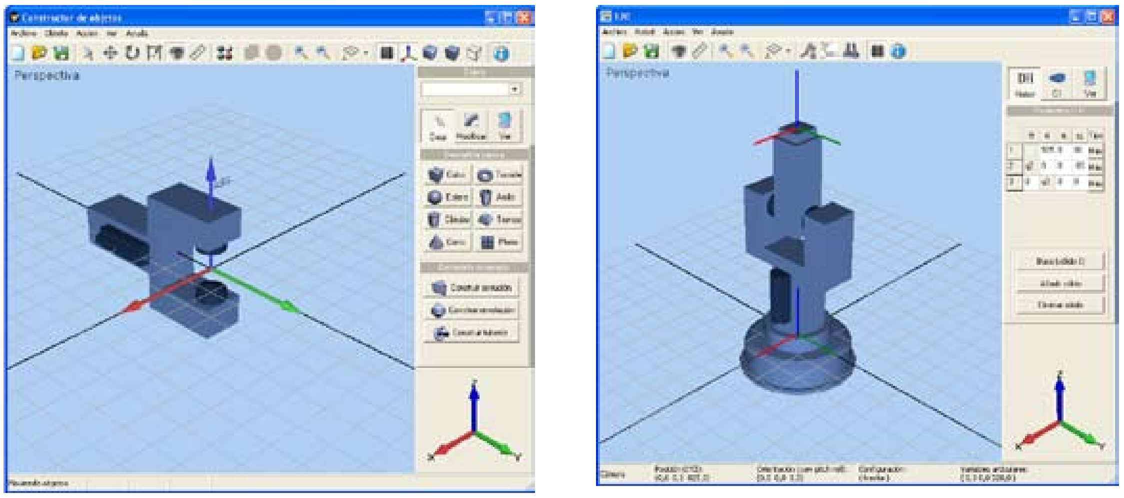
Simulation tools

In this section, we introduce the simulation tools developed for this project-based course: RobotScene and SGRobot. RobotScene [15] is a specific software tool that provides a graphical interface for solving the kinematics, following the Denavit-Hartemberg procedure [13], and dynamics of a robot. RobotScene also provides a framework for programming the previously created robots. It is composed by three modules specialized on solid, robot and scene creation respectively. SGRobot [16] is a graphical simulator for manipulators where robotic tasks can be programmed using a VAL II-based language. There are many robotic platforms that provide simulation frameworks in which users can develop robotic applications [5], [6], but they are not specifically oriented to robot manipulator design. Other platforms provide tools useful for robot kinematics and dynamics simulation, for example the Robotic Toolbox for Matlab [7] or Spacelib [8], but they have not elaborated graphical interfaces and do not provide robot programming tools. Some projects as OROCOS [9] and ROBOOP [10] allow covering this empty space, but they do not provide an easy graphical interface and so, their use requires important learning efforts. Furthermore, they need an external compiler in order to perform any simulation. In addition, there are several simulation platforms as ROBOGUIDE [11] and RobotStudio [12] developed by robot manufacturers in order to provide off-line programming tools specifically designed for their robots. RobotScene provides modules for creating solids, robots and robotic scenarios in an easy manner.

Robot modeling and kinematics

The first stage of the learning project is to model and solve the kinematics of a commercial robot. To achieve this, students follow the Denavit-Hartemberg (DH) convention [13], which allows obtaining forward kinematics in a systematic way. Once the students have analyzed the morphology of their assigned manipulator, identifying its joints and its solids, and assigning them all their reference frames according to the DH rules, they can begin to model their robot using RobotScene. Fig. 1(left) shows an example of a solid composed by several pieces.

Fig. 1. Example of the definition of a solid (left) and the robot obtained after assembling the robot solids (right).

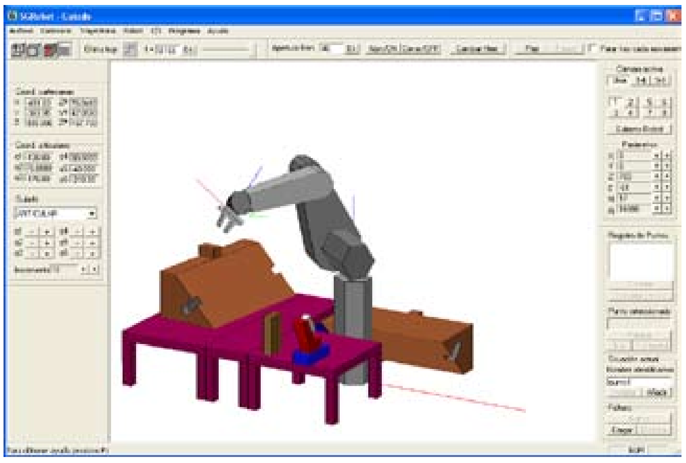


Once all the manipulator solids have been created, students can assemble their robots using the Robot Constructor Module. At this point, students need to have previously determined both the DH parameters as well as the equations that solve the inverse kinematics of their robot. It is important to note at this point the role of the Denavit-Hartenberg convention in the robot modeling process with RobotScene: it determines not only the main part of the joint modeling, but the form in which each solid has been created as well. Once students have finished the robot assembly process, Fig. 1 (right), they can use a robot guidance tool that allows them to move the robot by dragging the joint-associated cursors or by specifying robot destinations in joint coordinates. In order to complete the robot modeling stage, students must implement the equations that solve the inverse kinematics of their robot. These equations must have been previously derived by using either geometric or algebraic approaches. For making possible the mentioned implementation, the Robot Constructor Module provides a specific programming tool which allows editing, performing syntactical checking and compiling source code. During this programming phase, students must pay attention to several problems inherent to inverse kinematics, as ill-conditioned equations and singularities detection and their treatment. Once inverse kinematics is implemented, students can easily check its correctness by using the robot guidance tool.

Robot task programming

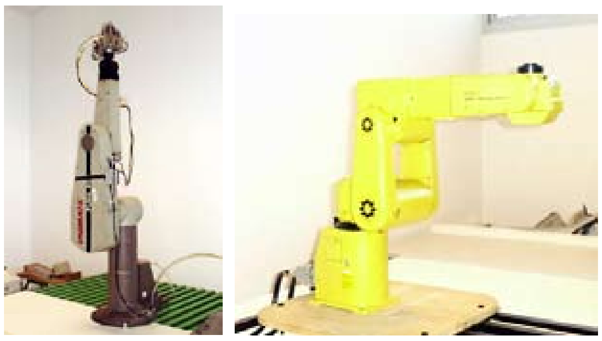
The second stage of the learning project-based experience consists in designing an industrial robotics task. For this part of the activity, a scene containing the robot arm and the objects involved in the task is given (Fig. 2).

Fig. 2. Example of robot arm in a scenario performing an assembly task.



The task the students have to design consists of several steps: First, the robot has to take the tool, which simulates a glue gun. Then, the tool needs to be calibrated and the objects of the scene located. The glue is spread around the surface of the base object following particular specifications and finally the set of the four pieces are mounted. Students have to program the robot by using the provided language. For this purpose the SGRobot tool is used. SGRobot is a graphical simulation tool that allows the programming of robot arms in a graphical way (guided) and textual way (using a defined programming language similar to the commercial language VAL II). The SGRobot consists of three programs: The object building, for building the objects to be manipulated by the implemented robot; the editor-compiler of the robotic programming language CVAL2 developed for SGRobot, and the guidance tool, which is for programming the robot by guidance, graphical representation of trajectories, object manipulation, etc. The learning objectives of the robot task programming are that students achieve the competences of planning a robotic task in a simulation environment, implementation of the task by means of guidance and explicit programming language, and solve successfully the usual problems of obstacle avoidance, object localization, detection and avoidance of robot singularities and robot configuration selection. The concept of the singularity is complex to understand but, thanks to the help of the simulation tool, the student can learn this concept intuitively. The simulation tool gives the option to compile and execute a program in CVAL2 and additionally, it allows traducing automatically the code to VAL II, the standard language that can be directly executed in the real robot. The code that can be generated from the simulation tool is used in a practicing activity with the real robots (Fig. 3). The goal of this part is that students work with real robots learning the basic elements of an industrial robot and its operation. They also learn and practice the different ways of robot programming while designing the robotic task by guidance and textual programming.

Fig. 3. PUMA and FANUC robots used in the practicing activity.



Learning improvements

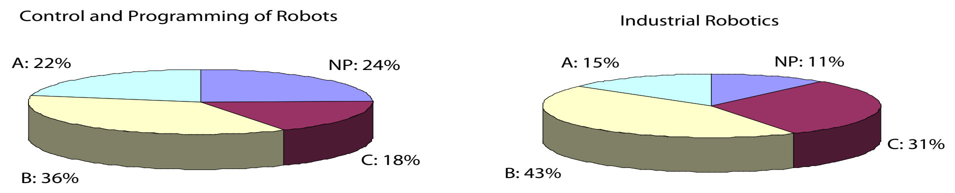
In the framework of new methodologies, laboratory and project-oriented courses are encouraged. In our case, two main learning aspects can be benefited from the use of simulation tools in the context of an industrial robotics related subject: the robot control and modeling in-depth comprehension, and the acquisition of robot programming skills.

The main learning improvements provided by the use of RobotScene during the robot modeling phase, are: the better comprehension of DH convention, and the in-depth understanding of the inverse kinematics problem. RobotScene provides a framework in which students can check in a visual way their DH parameters, because any error will be reflected as a wrong solid assembly. On the other hand, the inverse kinematics programming allows understanding its authentic complexity, because students must take into account some different problems related to its nature and implementation. Finally, SGRobot is useful to know and appreciate the utility of a graphical robotic simulator. Students can learn and practice with the different ways of programming robots. The activity requires that students implement under the simulator an assembling task using guidance and explicit textual language. They also need to document their final application including explanations about the reference assignment and the motion planning design. In order to perform the robotic task successfully the students need to solve usual problems that appears in programming robotic tasks, like obstacle avoidance, object localization, singularities avoidance, robot configuration selection, etc.

In general, it should be noted that instruction is less effective and less efficient than instructional approaches that place a strong emphasis on guidance of the student learning process [14]. The advantage of guidance in favor of active learning begins to recede only when learners have sufficiently high prior knowledge for autonomous learning. Therefore, in order to perform these activities with guarantees of success a prior knowledge is required from the students.

The improvements observed because of the activity proposed are supported by the rate of success and the good reception of the courses involved. The mean values of the grades obtained by students from the last 5 years (from 2003 to 2008) are shown in Fig. 4. The grades are defined in a scale from 0 to 10 with A between 9 and 10, B between 7 and 9, C between 5 and 7, D failure to pass, and NP if the student does not apply to the exam (the presentation of the project). Note that there are no D grades in the results. This is because, given the project-based focus of the course, the students work is all up to date. Moreover, each student knows before the end of the course if he has acquired the competences necessary to pass. Otherwise, they are advised during the course how to improve they performance in order to complete the course successfully. If not, the student can choose not to apply to the evaluation.

Fig. 4. Mean grades from years 2003 to 2008. The grades are (NP, D, C, B, A)



We have polled 18 students during this academic year 08-09 about their dedication to the course of Control and Programming of robots. The mean result is that they spent 50 hours (35%) in studying the theory and assistance to class hours. They spent 43 hours (30%) in practice sessions, including previous preparation, assistance, and elaboration of the corresponding reports. And they spent 51 hours (35%) in their work projects. Note that students estimate that 65% of the time dedicated to the course corresponds to practice, and relevance of practice is a characteristic of the competences-based learning. The course is designed with 4.8 ECTS (European Credit Transfer and accumulation System), approximately 116 hours of student work, including 25 hours of teaching classes, 21 laboratory hours, and 70 hours of personal work and other activities. Comparing with the poll results, students estimate that they spent in mean 28 hours of personal work over the corresponding designed time of the course. In order to reduce this difference we are going to introduce more active learning methodologies in teaching classes to improve acquisition of theoretical concepts required to fulfill the project-based activities.

References

01. De Wit, K. (2003), "The Consequences of European integration for higher education," International Association of Universities (IAU), in Higher Education Policy, vol. 16, no. 2, pp. 161-178.
02. The official Bologna Process website (2009). http://www.bologna2009benelux.org/
03. Halvorsen, T., Mathisen, G., and Skauge, T. (2005), "Identity Formation or Knowledge Shopping: Education and research in the new globality," Norwegian Centre for International Cooperation in Higher Education (SIU), Bergen.
04. Chickering, A. W., and Gamson, Z. F (1987). "Seven Principles for Good Practice." The American Association for Higher Education Bulletin, 39: 3-7. ED 282 491.
05. Microsoft (2008). Microsoft Robotics Developer Studio, msdn.microsoft.com/robotics/.
06. Mellado, M., Correcher, C., Catret, J. V., and Puig, D. (2003). "VirtualRobot: an open general-purpose simulation tool for robotics." The European Simulation and Modelling Conference (ESM2003), EUROSIS, Naples, Italy, pp. 271–350.
07. Corke, P. I. (1996), "A robotics toolbox for MATLAB", IEEE Robotics and Automation Magazine, Vol. 3, No. 1, pp. 24-32.
08. Legnani, G. (2006). "SPACELIB: a software library for the Kinematic and dynamic analysis of systems of rigid bodies". U. di Brescia. http://bsing.ing.unibs.it/~glegnani/.
09. Bruyninckx, H. (2001). "Open robot control software: the OROCOS project." IEEE International Conference on Robotics and Automation (ICRA), pp. 2523–2528.
10. Gourdeau, R. (1997). "Object oriented programming for robotic manipulators simulation." IEEE Robotics and Automation Magazine, Vol.4, No.3.
11. Fanuc (2008). ROBOGUIDE: a family of offline robot simulation software, http://www.fanucrobotics.com/.
12. ABB (2008). "RobotStudio: of offline robot programming for ABB robots," http://www.abb.com/.
13. Denavit, J., and Hartenberg, R. S. (1995). "Kinematic notation for lower-pair mechanisms based on matrices." Transactions of the ASME, Journal of Applied Mechanics, Vol. 23, pp. 215-221.
14. Kirschner, P. A., Sweller, J., and Clark, R. E. (2006). "Why minimal guidance during instruction does not work: an analysis of the failure of constructivist, discovery, problem-based, experiential, and inquiry-based teaching", Educational Psychologist 41 (2) 75-86.
15. Romeo, A. (2009) "The Role of Simulation Tools in the Teaching of Robot Control and Programming," 40th International Symposium on Robotics. To appear.
16. Guerrero, J.J. "Practicas de Control y Programación de Robots. Incluye Simulador Grafico SGRobot 2.0", Dep. legal Z-178-2003, Universidad de Zaragoza. Email to jguerrer@unizar.es to obtain a free copy for teaching.