Collaborative Subjects for Embedded Systems Learning in the EHEA Frame: A Practical Approach

Pablo García, Jose A. Cancelas, Victor M. González
University of Oviedo. Dept.of Elec., Computer & System Engineering
Gijón, 33204, Spain
Email: pgarcia@isa.uniovi.es, cancelas@isa.uniovi.es, victor@isa.uniovi.es

***Abstract*—This paper presents a teaching experience related to the learning methodology in embedded systems within the European Higher Education Area (EHEA) frame. A Problem Based Learning (PBL) methodology has been applied for combining two courses of a bachelor degree in computer systems. The objectives are the integration and application of previously acquired and new knowledge to create an application for the supervision and control of a mobile robot. Technologies used during the development of the project include real-time programming using Ada language, digital control theory and communications.**

***Index Terms*—PBL, real-time systems, digital control, supervision.**

I. Introduction

TRADITIONALLY, technical courses at engineering degrees are organized following a logical sequence, being the earlier the base for the later. However, almost no effort is put on presenting the students with the idea that the integration of the different knowledge and technologies acquired along the different courses, is a fundamental skill in order to be able to solve complex multidisciplinary problems, like the ones they will have to face at disciplines like systems engineering, computer systems engineering, or embedded systems engineering [1]. After completing an engineering degree, the students end up with a wide collection of weakly related concepts and skills organized as watertight compartments [2]. This paper describes a model to mitigate this main drawback by combining two courses in a collaborative way. These courses are Supervision and Control of Systems (SCS) and Real-time Systems (RTS) which are part of the Computer Systems degree offered at the Bachelor Technical School of Computer Systems Engineering at the University of Oviedo, Spain.

The PBL methodology has traditionally been used in higher education, mostly in medical training, and has proof to be a very powerful contextual, collaborative and constructivist learning model [3]. The PBL methodology [4] allows posing a complex problem, which requires integration of knowledge, before learning [5]. Following this methodology, the students are able to discover that in order to solve the problem they first need to acquire new knowledge. However, an eye must be kept on PBL detractor's arguments which suggest that students taking courses based on PBL model show potentially significant gaps in their cognitive knowledge base and do not demonstrate expert reasoning patterns [6].

A blended learning model was adopted by both courses in order to get the best from PBL and minimize its drawbacks. On one hand, PBL was selected as the learning methodology for the hands-on laboratory classes, so the students were able to develop their constructive, self-directed, and collaborative capabilities. On the other hand, a more classical learning methodology was maintained for the theoretical classes, thus assuring that the students developed their cognitive knowledge base and expert reasoning skills. Besides, an innovation was introduced to the PBL methodology so multidisciplinary problems solving objective was achieved: both courses shared a common complex assignment by yielding part of their class time for its commissioning, merging together students from both courses at the same laboratory and class time. The professors of both courses collaborated by merging partial objectives into a common global one, and working side by side as tutors or facilitators.

A common assignment was selected as the final evaluation method because implementation of real-time control systems requires, in general, the interaction between different process layers: real-time control of individual systems, real-time coordinated control and supervision. Often, these layers are programmed in different languages and require the use of different tools in order to build the overall system. The inner control loop is usually implemented in a specific hardware, such a microcontroller or a DSP. In these systems, the programming language is usually C or assembler. For this particular project a DSP microcontroller programmed in C was chosen. For the outer control loop, there are many possibilities depending on the particular application. The running platform can be a computer, an embedded system or a PLC. The platform requirements at this level are related to the need of communication with both the inner control loop and the supervision layer and to the tasks synchronization. A common used language for fulfill this requirements is Ada [7], [8]. Finally, the supervision layer is usually implemented in a language allowing designing the graphical user interface (GUI) as well as tools for data visualization and storage. Common choices are C++, C# or Java. For the development of this course we have selected C++.

In order to carry out such assignment the students must not only apply the concepts acquired at these subjects, but also using some other knowledge from other courses of the degree. The students need to search for new sources of knowledge, to combine different sources of information, to make use of a wide variety of technologies, to improve their capability of self organizing and self learning and to develop their skill

978-1-4244-6571-2/10/$26.00 ©2010 IEEE

April 14-16, 2010, Madrid, SPAIN
IEEE EDUCON Education Engineering 2010 – The Future of Global Learning Engineering Education

of team working. Each student (or group of students) must reach a milestone before going ahead with the next. One of the evaluation criteria used was to measure how much help from the professor was needed by the student to fulfill each milestone. Every student has to register her/his work progress in a portfolio clipboard [9], which includes: managed sources of information, tasks, achieved goals, acquired knowledge and main difficulties. The new teaching methodology requires a more active role from the student in the learning process. A big effort was invested in carefully explaining the students the new rules: the milestones protocol and the evaluation criteria. By doing so, the students evolved from a tolerance attitude into a proactive participative one, making the experience theirs. At the end of the course, every student was faced a course evaluation form containing questions to measure out their view of the experience. Most of the students found that collaborative assignments help them to improve their skill of solving multidisciplinary problems.

This paper presents a collaborative experience between two courses related to the embedded systems engineering, which aim is to complement engineering student's knowledge acquired at the different degree courses, with the capability of facing complex multidisciplinary problems. PBL is used as the learning methodology for the hands-on classes. The way both courses collaborate and how PBL is extended is explained then. Finally, student's opinion about this experience is analyzed by means of one questionnaire.

II. Courses Description

A. Students

The students can take one or both courses. These two courses are in the third year of the Computer Systems degree. Before these topics, students are supposed to have passed topics in programming methodologies, electronic technology of computers, algorithms, data structures, industrial computing and networks.

One might think that the best learning outcomes will be obtained if all students take both courses, so all of them would have studied the theoretical concepts of both courses before facing the final assignment. But the experience has shown that because of this collaborative model, even in the worse scenario where no student has taken both courses, it produces great benefits, because it allows them to develop their team working skills. The key idea is to set up balanced groups made up of students from both courses. No group must be made up of students taking only one course. Those students that do not take one particular course get its basic concepts by interacting with those that do take it, because they need to understand that part in order to understand the whole picture. It is true that such students are not going to develop their expert reasoning at that particular topics of that course, but still they will get a flavor of them, that will allow them knowing its nature, its application, and its justification, and maybe raise their interest in that field, with little effort.

B. Courses organization

Both courses duration is 4.8 ECTS and are organized as follows. There are four hours every week separated into two lectures of two hours each. During the first bimester, first lecture is on theoretical content and second one a hands-on laboratory. It is worth noticing that even at the theoretical lectures students are in front of the computers. Every key aspect of the systems control or real-time theory is tested using computer simulation. Simulink was used for the systems control part and Ada over Linux for the real-time one. During the second bimester, all four hours are laboratory ones in order to provide the students with enough time to make the final assignment project.

Theoretical contents during the first bimester are headed to make the students learn the basics on real-time and embedded systems, following the timetable shown in Table I.

Course materials are stored in a mediaWiki [10]. The use of a wiki allows to easy share materials needed in both courses as well as materials from other courses taught within the Department.

C. Assessment method

Engineering educators recognize that student's exams, class exercises and homework can effectively measure mastery of facts and formulas. However, these sorts of assessments do not encourage students to develop their analytical capabilities as they do not measure the student's skills to understand and apply what they had learned. Nowadays, industry is demanding engineers with a set of skills that are beyond the knowledge that a student can achieve studying a book. In contrast, projects, allow educators to emphasize "the important role that experience plays in the learning process" [11]. We focus our subjects in problem based learning. Our teaching style combines the lectures and labs with the development of some projects. Usually, the students do not have all the pieces to develop their assignments, but need to search in the additional contents presented in the web page of the subjects, in Internet or other media to achieve their projects. Obviously, a traditional exam is not a valid media to measure the efforts and the learning of the students in the problem based learning. For this purpose, we use a portfolio. "A portfolio is a purposeful collection of student's works that demonstrate their efforts, progress, and achievements in selected areas of the curriculum" [12]. It can include the best works, but also preliminary designs or any artifact that could show the progress of the student. Usually, most of our students focus their efforts in programming instead of thinking or analyzing the best solution. Classical assessment based on exams does not promote the critical thinking. Many times when they do team work their tactics is dividing and joining. Each member of the group focuses in one aspect of the problem. This is not the best approach to team working. By using a portfolio, the students can incorporate the results of their meetings, the changes that they have agreed and how they are learning.

Besides, with the portfolio development, the students can improve their ability on writing technical documents. This is recognized as one of the key aspects needing improvement on the new engineers [13]. Portfolios have also been suggested as a direct source for research into student knowledge [14].

The most difficult aspect of a problem based learning course is determining if the desired goals and objectives have been

TABLE I
COURSES TIMETABLE.

Week	Lectures		Labs	
	SCS	RTS	SCS	RTS
1	Basics on control systems	Introduction to RTS		
2	Introduction to dynamic models and feedback	Languajes for RTS applications: Ada	Matlab/Simulink (I)	Initial programming in Ada
3	Time response	Reliability and error processing	Matlab/Simulink (II)	Fault tolerance programming applications
4	PID control	Concurrency	Analog units	Modular programming
5	Introduction to digital control	Real time programming	DC motor control	Concurrent programming
6	Discrete systems	Low level software programming	Microcontrollers (I)	Lego sumo robot
7	Digital control implementation	Real time sofware design	Microcontrollers (II)	RTS design methodology
8	SCADA basis theory	Real time operating systems	DC motor digital control	Programming a railroad model
9 - 15	Final Assigment		SCADA (I) SCADA (II)	

TABLE II
PORTFOLIO EVALUATION.

Portfolio content	Evaluation item			
	writing communication	planning	self-analysis	team work
auto evaluation			X	
laboratory reports	X			
design documents	X		X	X
meetings schedule		X		X

achieved. Student's feedback is used extensively to evaluate the performance of both the teaching staff and the subject. Again, traditional assessment methods reveal as a poor tool to determine the achievement of the objectives. Portfolio is an efficient way to demonstrate the achievement of the skills that are signaled as objectives to the course. Student portfolios are listed as a possible means of assessment under the basic level accreditation criteria according to the Accreditation Board for Engineering and Technology (ABET) "Engineering Criteria 2000."

We are using electronic portfolios in a semi-structured way. Because that is our student's first attempt, we suggested them some minimum elements, thus giving them a template than they can freely improve. The student portfolios, include student goals for learning, works in progress, and reflection on the work and processes. Table II shows how any aspect of the evaluation is reflected in the portfolio information.

The use of the portfolio as the evaluation method has proved to be quite satisfactory. The tasks the students needed to complete has been achieved and the implication of students was high, according to the students response shown later in section V.

III. LECTURES

A. Lectures description

1) Supervision and Control of Systems:

- Basics on Control Systems: Introduction to the problem of control. Through the use of a model that does not incorporate dynamics, the effects of disturbances and open loop/close loop behaviors are explained.
- Introduction to Dynamic Models and Feedback: The model presented in the previous lecture is extended in order to incorporate dynamics. Linear and invariant systems. Lapace transform and impulse response. Effects of dynamics in the velocity control of a vehicle are shown.
- Time response: Analysis of time response when arbitrary input is applied to a system. Convolution integral. Step and ramp response. Systems order. Position control of a vehicle.
- PID control: Impact of controller actions (P,I,D) in the close loop response of a controlled system. Reference tracking and disturbance rejection. Control design for electromechanical systems: Truxal pole cancellation. Formulation and implementation of a speed control for a DC motor.
- Introduction to digital control: Analog control vs Digital control. Additional elements in the control loop for digital control implementation. Signal Sampling: sampling theorem and aliasing effects. A/D conversion. Quantization effects. D/A conversion and PWM.
- Discrete systems: Sequences and discrete systems. Difference equations. Discrete time implementation of integral and differential operations. The discretization problem. Linear and Tustin approximations.
- Digital control implementation: Foundations for using the theoretical concepts in a microcontroller based system. Events driven programming. Interrupts. Program structure for digital control realization.
- Basics on supervisory control and data acquisition (SCADA) applications: data acquisition module, graphical representation of the process, alarm handling, data base module, graphical representation of signals, etc.

2) Real Time Systems:

- Introduction to Real-Time Systems: Description of the characteristics and main properties of RTS.
- Languages for RTS applications; Ada: Description of the desirable characteristics that a RTS programming language must cover.
- Reliability and error processing: Study of the aspects

related to the construction of free error programs and what can be done when errors appear during the program execution. Ada mechanisms for error management are shown. Fault tolerance and exceptions handling.
- Concurrency: Concurrent aspects of the RTS. Tasking. Interprocess communication. Shared memory and programming with Protected Objects in Ada. Interprocess communication and message passing mechanism, rendezvous in Ada. Practical cases solving.
- Real-Time Programming: Time management, programming applications using time functions. Practical cases solving, time-outs, asynchronous transfer control, periodical tasks programming, sporadic tasks programming. Scheduling and time requirements. Solving priority inversion problem. Ada programming facilities to solve all these aspects.
- Low level software programming: Requirements to constructs programs adapted to the characteristics of hardware devices. Interfaces between Ada and C.
- Real time software design: Hrt Hood Methodology. Study of a practical Case.
- Real time operating Systems: Description of the of the desirable characteristics that a RTS Operating System must has. Examples.

B. Labs description

1) Supervision and Control of Systems:

- Matlab/Simulink (I): Introduction to Matlab/Simulink. Implementation of a velocity control for a vehicle without dynamics. Comparison between open loop and close loop control strategies.
- Matlab/Simulink (II): Transfer functions in simulink. Import of model parameters and export results from/to Matlab. Implementation of a velocity control for a vehicle incorporating dynamics.
- Analog units: (Fig. 1). Time response of first and second order dynamic systems by using analog electronic circuits. Measurement of electrical variables using a digital scope. Measurement of system properties such a settle time, peak value, overshoot and final value.
- DC Motor Control: (Fig. 2). Control of a DC machine using analog units and *Feedback*® mechanical unit. Current control, velocity control and position control. Relation between the electrical variables (armature current) and mechanical variables (speed/position). Selection of controller's gains an impact on the control dynamics.
- Microcontrollers (I): DSPic 30F6010 features. Introduction to MPLAB. Configuration of I/O ports, Timers and AD converter. Using the simulator for testing purposes.
- Microcontrollers (II): Configuration of PWM and motor PWM modules. Serial communications. Interface between Matlab and the DSPic.
- DC motor digital control: (Fig. 3). Implementation of a speed control for a DC motor using the DSPic 30F6010. A low level library for programming the dspPIC has been developed by our own, so making possible for the students to focus on the control problem instead on the digital system hardware configuration.

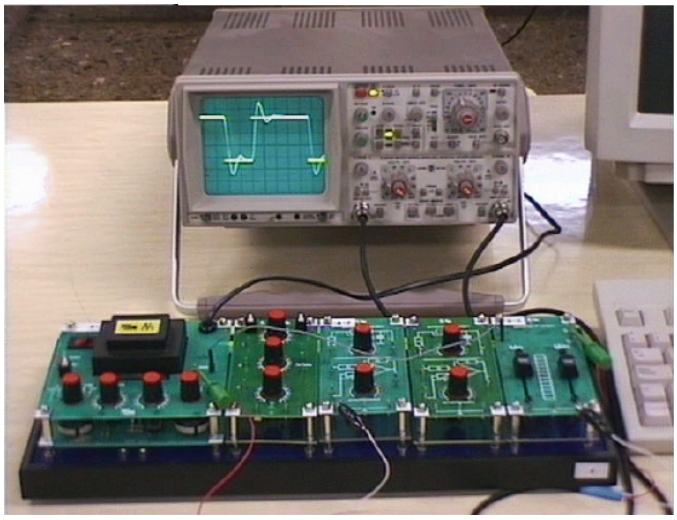


Fig. 1. Analog units for system response measurement.

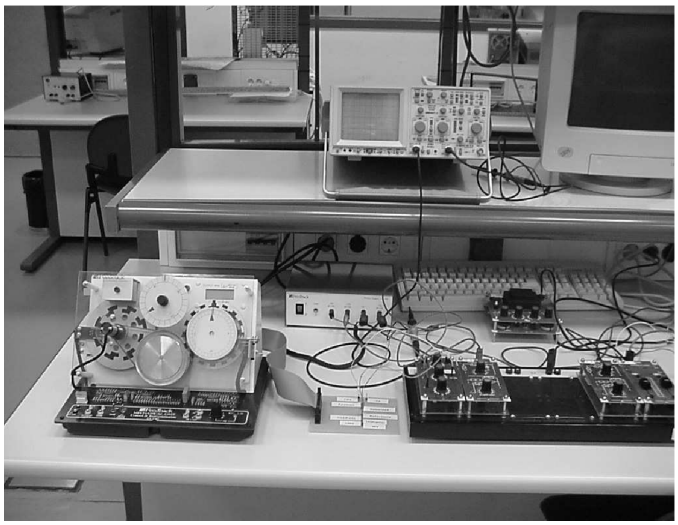


Fig. 2. *Feedback*® mechanical unit educational system for the control of a DC motor.

- SCADA (I): Design and implementation of the skeleton of the data acquisition module using the publication/subscription paradigm. TCP/IP technology is used to implement the communications to the Ada control layer.
- SCADA (II): Design and implementation of the skeleton of the graphical interface of the system that allows following the evolution of the robot.

2) Real Time Systems:

- Initial programming in Ada: Basic structures and data types to implement and Ada program. Modular decomposition.
- Fault tolerance programming applications.
- Modular programming. Generic Units and package.
- Concurrent Programming. Tasks creation. Solving synchronization problems.
- RTS Design Methodology. Using Stood to create a model to solve a RTS problem .
- Lego Sumo robot (Fig. 4(a)): Construction and programming of a Lego boot using Ada. This is a team

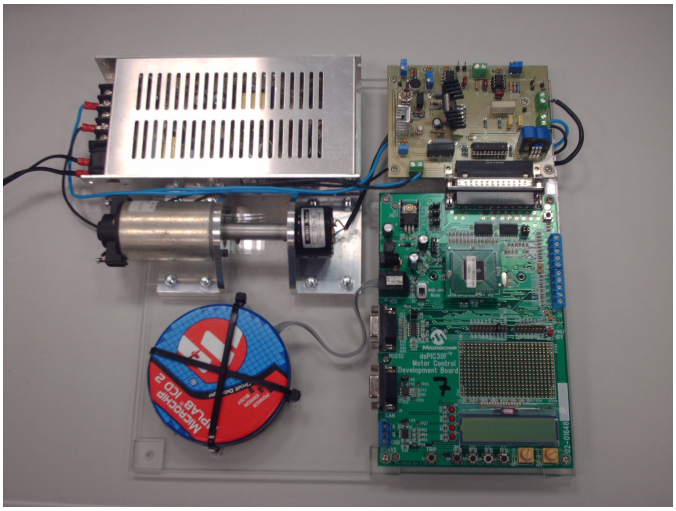


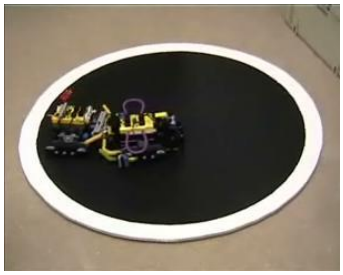
Fig. 3. dspPIC an custom board for the control of a DC motor.

work where the students have to build their own sumo bot, searching information on the net and books, and programming them using an Ada Compiler developed by [7]. With Lego RCX is easy to construct a robot, including different sensors and motors without electrical or mechanical knowledge. This set is frequently chosen for its low cost and ease of use. However, most of the work being done with it is in teaching reactive robotic architectures, [15], [16]. As the robot is a sumo bot, its behavior depends of its rival and also on the limits of the fight pitch. In this team work students must write their portfolio to collect their advances and decisions in order to improve the original design. After a first combat they have an opportunity to re-design their bot for a new combat.

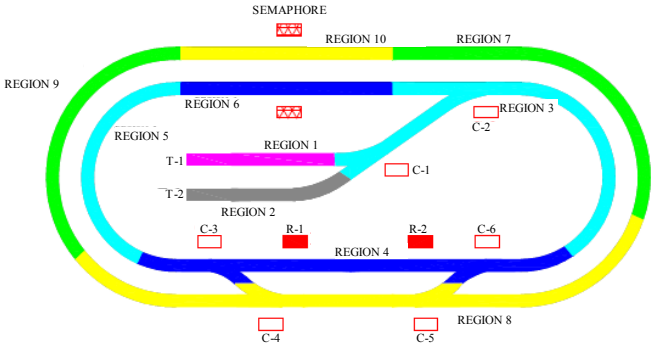
- Programming a railroad model (Fig. 4(b)): Using simulators is many times perceived as a poor experience by students due to the fact that they are only using software instead of a real plant. A richer experience is achieved when is possible to use a physical model. Model railroads provide a wealth of problems from both the discrete and continuous real-time domains. The electronics are easily understood by most undergraduate computer science students. Students are highly enthusiastic about writing software to control a model train layout [8]. We have a simple railroad with two concentric ovals and the tracks to change from one to another. Train's reference position is obtained through two cameras and a computer that process images. The team must control the movement of two trains so they follow a pre-defined trajectory with no collisions and as quick as possible. The team work must collect results at their portfolio.

IV. Final Assignment

In order to practice the learned knowledge, students are faced to a final laboratory project at the end of the semester. In this project, students work together in groups of two or three people. The main idea of the project is to force



(a)



(b)

Fig. 4. RTS labs. a) Lego Sumo robot, b) Railroad model.

the students to discover that using the previous acquired knowledge is possible to solve more complicated problems. Besides, the students need to make use of formerly acquired knowledge and technologies such as: TCP/IP communications, C++ programming, data bases, RS-232 serial communications, graphics plotting, and using a professional development suite, in this case Microsoft®Visual Studio .Net or CodeGear® Builder. Each group of students programmed the whole system and made a presentation of the work during the last lesson.

Robotics, and in particular mobile robots, is an attractive platform where the students can put in practice the topics of the course [17]. A two steering wheel mobile robot was selected as the target platform for the collaborative assignment [18]. We used several platforms and tools to integrate the different process layers described before: real-time control of DC motors (implemented in a DSPic), real-time coordinated control (last course implemented in a PC but moving to an embedded system –TS-7250 from Technologic systems®– platform running an ARM processor in the next course) and supervision (remote PC).

For the proposed problem, a hardware platform has been designed by our own. Schematic representation of the designed platform is shown in Fig. 5. As shown, two of the wheels are manned by a DC motor, each of one driven by a H-bridge module connected to the PWM outputs of a DSP microcontroller (*Microchip® dsPIC30F610*). The angular velocity of the machine is measured using a speed sensor connected to the axis of the machine. The setup for the simulation of each wheel is shown in Fig. 3.

The inner control loops are responsible of controlling the angular velocity of each machine by means of a PI controller. Both DSP's are connected through a RS-232 interface to a

TABLE III
ASSESTMENT SKILLS.

Mobile robot tasks	Skills				
	control design	communications	supervision	Ada programming	microcontrollers
Control of DC Motor	X				X
Trajectory Tracking	X			X	
User input and feedback			X		
Subsystems interface		X	X	X	X

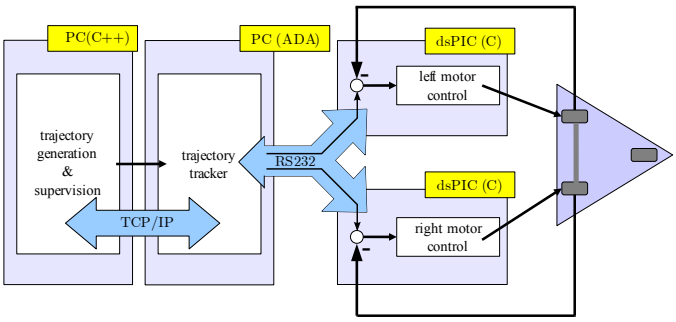


Fig. 5. Schematic for the designed platform.

TABLE IV
SKILLS LEARNING.

	SCS	RTS	PBL
control design	X		
communications		X	X
supervision	X		
Ada programming		X	
microcontrollers	X		

PC running the higher level real-time control algorithm. This control layer is implemented in Ada and its objective is to generate the velocity commands so the robot is able to track the desired trajectory. In addition, the Ada program must communicate with the supervision layer though a TCP socket interface. The supervision layer implements the interface to the process. It includes the GUI that allows the user to select the desired trajectory to follow from a base of trajectories. It also represents the reference trajectory and the actual one. It provides storage capabilities for generated data and for the representative values of the state of the process. Relation between needed skills and assessment main tasks as well as the course where the needed skills are learnt are shown in Tables III and IV, respectively.

A. Assignment development

The first PC (PC/HMI) is running the supervision application. The application runs two different tasks implemented using threads. The main thread is the responsible of attending the GUI events, painting in the screen the output of the process and storing the results in a database. The second thread, is executing the communication task. Communication is implemented using a TCP/IP socket, being the supervision PC the client. Although the students decided what was the information needed in order to track the process, a recommendation was made to receive at least the variables containing the robot position, (x, y, θ).

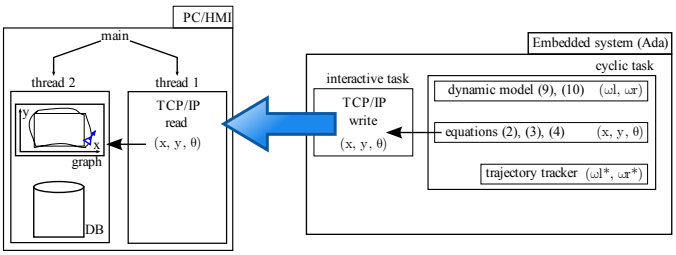


Fig. 6. Simplified logic diagram for the first stage of the final project.

The second PC was running the outer control loop, programmed in Ada. The PC was equipped and configured with the needed Ada tools. The application was executing also two different tasks. The main task, responsible of the generation of the velocity commands to the two wheels, was implemented as a cyclic task. The sample time was established to $10ms$. The actions programmed in this task were as follows: 1) read from the two RS-232 interfaces –one for each microcontroller– the actual wheels velocities, 2) using the equations described in section IV-B, calculate the actual velocity and position of the vehicle, 3) compute the new velocity commands for the wheels and, 4) write the commands to each RS-232 interface. The second task, running in the idle time, was attending the communication with the supervision PC. Communication was implemented using TCP/IP sockets and the Ada PC was configured as the server. The data sent to the client was the estimated position of the robot.

Finally, each microcontroller was running the inner control loop. This control loop was asserting that the motor velocity was following the command. Sample time for the control was set to $1ms$. Control loop was implemented in the interrupt service routine of the A/D converter. Actions taken in the routine were as follows: 1) read velocity from the velocity sensor, 2) take the reference received from the Ada through the RS-232, 3) calculate the new control action –voltage– as the output of the PI regulator, 4) update the PWM and, 4) write the measured velocity to the RS-232 interface.

B. System modeling and algorithm description

During the theoretical lessons on control theory within the SCS topic, the students are headed to the objective of designing a close-loop control for the mobile robot. Using *Simulink*®, the dynamic model of the system is explained, built and simulated. The proposed dynamic model has been taken from [18] as shown in Fig. 8. For the implementation of the control, a simplified version neglecting the feedforward decoupling mechanism shown in [18] has been implemented.

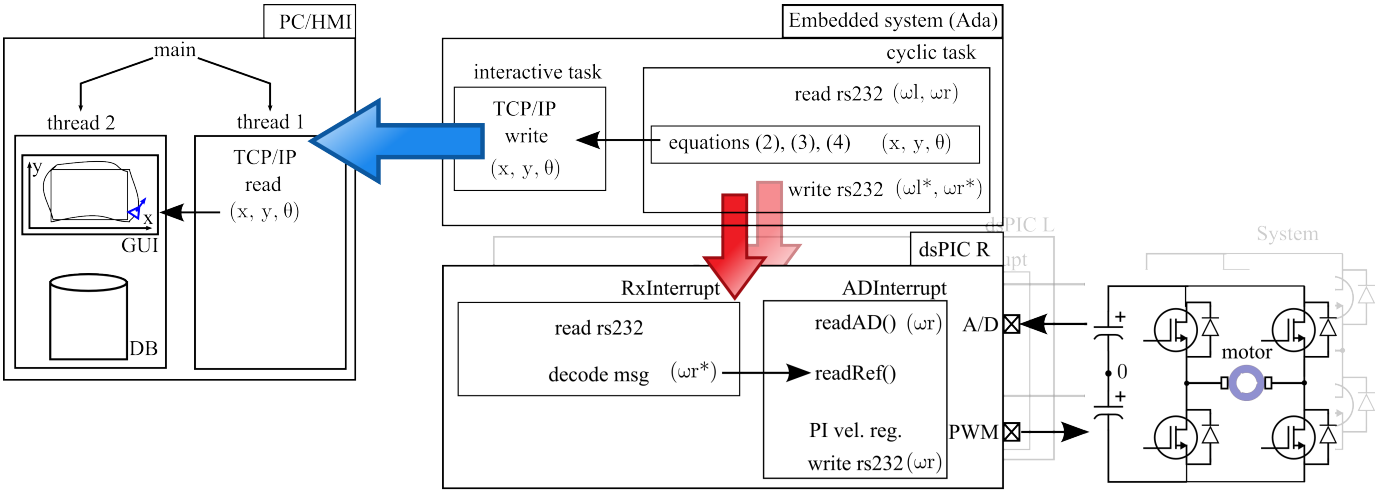


Fig. 7. Logical diagram of the final assignment. It shows the three systems interaction as well as key processes running on each system. For the sake of simplicity, only one microcontroller is shown in the top layer of the diagram.

From here, the complete model equations are described. The derived equations are provided to the students along with the aforementioned paper [18], so they can focus on the implementation details. In the subsequent subsections, first the cinematic and dynamic model of the vehicle are explained and second the implementation of a trajectory tracker is explained.

1) Dynamic model: Dynamic model is explained though the decomposition into three different models: steering cinematic model, vehicle cinematic model and vehicle dynamic model.

a) Steering system cinematic model: Cinematic model of the steering system allows obtaining linear and angular velocity of the vehicle from angular velocities of the wheels, as shown in (1).

$$\begin{bmatrix} v \\ w \end{bmatrix} = \begin{bmatrix} \frac{R_R}{2} & \frac{R_L}{2} \\ \frac{R_R}{T} & \frac{-R_L}{T} \end{bmatrix} \begin{bmatrix} \omega_R \\ \omega_L \end{bmatrix} \tag{1}$$

where, R_x and ω_x are the wheel radius and the angular velocity of wheel x respectively and T the vehicle wheelbase.

b) Vehicle cinematic model: Vehicle's cinematic model relates linear and angular velocity with vehicle position in a 2D space. It is worth noticing that vehicle position is given not only by (x, y) coordinates but it is needs to include a third component for the robot orientation (heading), so (x, y, θ) is the full state. Relations are shown in (2), (3) and (4)

$$\theta_{(t)} = \int_0^t \omega_{(t)} dt \tag{2}$$

$$x_{(t)} = \int_0^t v_{(t)} \cos\theta_{(t)} dt \tag{3}$$

$$y_{(t)} = \int_0^t v_{(t)} \sin\theta_{(t)} dt \tag{4}$$

c) Vehicle dynamic model: The dynamic model of the system allows relating the generated forces –dc motors torque– with the change in the actual position. This model is used in the simulator in order to allow isolate testing of each block in the project. A diagram for the dynamic model is shown in Fig. 8. For the proposed model, and using Newton's second law of dynamics, is possible to simulate the reaction of the vehicle to changes in the torque reference to the motors, as shown in (5) y (6)

$$\sum F = ma = m\frac{dv}{dt} \tag{5}$$

$$\sum \tau = J\alpha = J\frac{d\omega}{dt} \tag{6}$$

Using the equations from the steering cinematic model:

$$v = \frac{R}{2}(\omega_R + \omega_L) \Rightarrow \frac{dv}{dt} = \frac{R}{2}\left(\frac{d\omega_R}{dt} + \frac{d\omega_L}{dt}\right) \tag{7}$$

$$\omega = \frac{R}{T}(\omega_R - \omega_L) \Rightarrow \frac{d\omega}{dt} = \frac{R}{T}\left(\frac{d\omega_R}{dt} - \frac{d\omega_L}{dt}\right) \tag{8}$$

where wheel's radius has been assumed to be the same and equal to R for the sake of simplification.

By algebraic manipulation, and following reference [18], expressions (9) and (10) are obtained.

$$\tau_{MR} = A\frac{d\omega_R}{dt} + C\frac{d\omega_L}{dt} + E\omega_R \tag{9}$$

$$\tau_{ML} = B\frac{d\omega_L}{dt} + D\frac{d\omega_R}{dt} + F\omega_L \tag{10}$$

where,

$$A = \frac{1}{g_R}\left(g_R^2 J_{MR} + J_R + \frac{R_R^2}{2}\left(\frac{M}{2} + \frac{J}{T^2}\right)\right)$$

$$B = \frac{1}{g_R}\left(g_L^2 J_{ML} + J_L + \frac{R_L^2}{2}\left(\frac{M}{2} + \frac{J}{T^2}\right)\right)$$

$$C = \frac{1}{g_R}\left(\frac{R_R R_L}{2}\left(\frac{M}{2} - \frac{J}{T^2}\right)\right)$$

$$D = \frac{1}{g_L}\left(\frac{R_R R_L}{2}\left(\frac{M}{2} - \frac{J}{T^2}\right)\right)$$

$$E = \frac{1}{g_R} b_R$$

$$F = \frac{1}{g_L} b_L$$

2) Trajectory tracker: Trajectory tracker module's main objective is to generate commands for the velocity control loop of the motors driving the wheels. At each sample time, available information is the desired trajectory selected at the supervision layer and actual position estimation. Then, the trajectory tracker needs to implement an algorithm allowing the following transformation:

$$[\omega_r^*, \omega_l^*] = f\,(x^*, y^*, x, y)$$

where x^*, y^* are the coordinates of the commanded trajectory, x, y estimated actual position and ω_r^*, ω_l^* velocity commands to each motor.

As explained before, implementation of the trajectory tracker is made in Ada language. The module communicates with the inner control loop through the serial port. In addition, it has to receive the precalculate trajectory from the supervision layer. This communication is done by TCP sockets.

The algorithm to generate velocity commands is based on the cinematic model of the vehicle and in the measurements given by the velocity sensors at each motor.

3) Algorithm description: Mobile position in x, y plane (the movement is constricted to this plane) is determined by the estimated system state (x, y, θ), where x is the projection in the x axis, y the projection on the y one and θ robot heading. Comparison between commanded position (x^*, y^*) and estimated one (x, y) from the cinematic model, allows to generate velocity commands. Because the given trajectory does not include heading command –θ^*–, first step is to know if given the actual position and the reference, a change in the robot heading is needed in order to reach the next point of the desired trajectory. Heading command is calculated as (11):

$$\theta^*_{[k]} = atan\frac{y^*_{[k]} - y_{[k]}}{x^*_{[k]} - x_{[k]}} \tag{11}$$

Once heading command has been obtained, it is already possible to get linear and angular velocity commands using the discrete derivatives by Tustin approximation from the cinematic model of the vehicle (12) to (14).

$$\omega^*_{[k]} = 2\frac{\theta^*_{[k]} - \theta_{[k]}}{T_s} - \omega^*_{[k-1]} \tag{12}$$

$$v^*_{x[k]} = 2\frac{x^*_{[k]} - x_{[k]}}{T_s} - v^*_{x[k-1]} \tag{13}$$

$$v^*_{y[k]} = 2\frac{y^*_{[k]} - y_{[k]}}{T_s} - v^*_{y[k-1]} \tag{14}$$

$$v^*_{[k]} = \sqrt{{v^*_{x[k]}}^2 + {v^*_{y[k]}}^2} \tag{15}$$

where T_s is the sample time.

From linear and angular velocity commands v^* and ω^*, references to the inner velocity control loop ω_r^* and ω_l^*, are obtained from the inverse cinematic model of the steering system (16)

$$\begin{bmatrix} \omega_{rR} \\ \omega_{rL} \end{bmatrix} = \begin{bmatrix} \frac{1}{R_{rR}} & \frac{T}{2R_{rR}} \\ \frac{1}{R_{rL}} & -\frac{T}{2R_{rL}} \end{bmatrix} \begin{bmatrix} v \\ \omega \end{bmatrix} \tag{16}$$

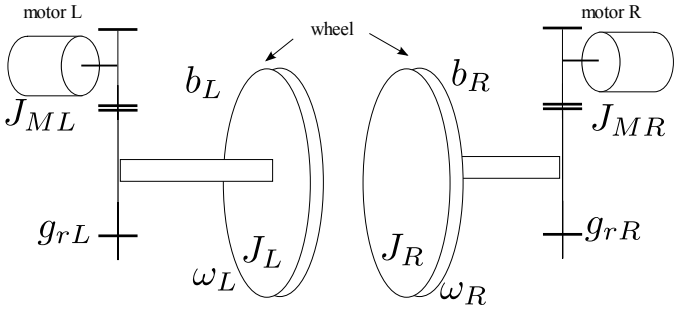


Fig. 8. Mechanical model for the two steering wheel mobile robot.

V. Students Response

Student's opinion about the presented courses has been obtained through the use of one questionnaire. The questionnaires were fulfilled by the students after the final assignment presentation was done and before the evaluation results were issued. The questions about the collaboration of both courses is shown in Table V.

All the students agreed that it has been a good experience. It is worth noticing that lower marks are in the questions related to the effort required to pass the courses as well as to the materials. About this last question, we are working on improving both the materials for the hands-on laboratories and also the documentation in the web page. The experience taught us that the question about the course difficulty is related to how easy the information needed to complete the project can be found. Therefore, we think this mark can also be raised with more documentation about some specific problems they encounter. In order to avoid returning to the previous teaching method, where all the materials were immediately available to the students, we will keep the information filtered. It will be shown to the students in fragments, and only after assuring that they have searched the information by their own.

Additionally, in order to avoid tendency of the students to delay the tasks until the last weeks of the course, more milestones are going to be established during the first bimester. This will help to minimize the overload sensation because of too much effort concentrated in the last weeks.

Finally, taking into account that this is the first year of the experience, we think the results are quite satisfactory.

VI. Conclusions

This paper has presented a collaborative experience between two courses related to the embedded systems engineering. A blended learning mode has been applied. PBL has been used as the learning methodology for the hands-on classes, and

TABLE V
STUDENTS RESPONSE QUESTIONNAIRE. MARKS RANGE IS FROM 0 TO 5.

Question	Result	
I think...	mean	std
...it has been important for my curriculum.	4,13	0,64
...it is positive to make common assignments.	4,63	0,52
...it is positive to blend different topics in the same assignment.	4,75	0,46
...the effort needed to pass the courses is adequate.	3,5	0,53
...the materials are appropriated.	3,63	0,52
...the common assignment is challenging and motivating.	4,63	0,74

a more traditional teaching for the lectures. Description on the evaluation method using a portfolio has been extensively explained. Common final assignment design as well as implementation details have been included. Students opinions about the experience has been reported and analyzed by means of one questionnaire.

Results demonstrate that the experience is positive for the develop of reasoning, critical thinking and use of acquired knowledge. This is an agreement with other authors [19], who conclude that the successful completion of an embedded design gives the student a sense of achievement which is lacking in more conventional engineering courses.

REFERENCES

[1] R. L. Traylor, D. Heer, and T. S. Fiez, "Using an integrated platform for learning to reinvent engineering education," *IEEE Transactions on Education*, vol. 46, pp. 409–419, Nov. 2003.

[2] R. Garcia-Robles, F. D. del Rio, S. Vicente-Diaz, and A. Linares-Barranco, "An elearning standard approach for supporting pbl in computer engineering," *IEEE Transactions on Education*, vol. 52, pp. 328–339, Aug. 2009.

[3] J. Moust, H. Berkel, and H. Schmidt, "Signs of erosion: Reflections on three decades of problem-based learning at maastricht university," *Higher Education*, vol. 50, pp. 665–683, Nov. 2005.

[4] R. V. Belhot, , J. H. L. Guerra, , and N. P. Kuri, "Problem-based learning in engineering education," Internation Conference on Engineering Education, ICEE, 1998.

[5] D. Rover, R. Mercado, Z. Zhang, M. Shelley, and D. Helvick, "Reflections on teaching and learning in an advanced undergraduate course in embedded systems," *IEEE Transactions on Education*, vol. 51, no. 3, pp. 400–412, 2008.

[6] M. A. Albanese and S. S. Mitchell, "Problem-based learning: A review of literature on its outcomes and implementation issues," *Acad. Med.*, vol. 68, pp. 52–81, Jan. 1993.

[7] B. S. Fagin, L. D. Merkle, and T. W. Eggers, "Teaching computer science with robotics using ada/mindstorms 2.0," in *2001 annual ACM SIGAda international conference on Ada*, pp. 73 – 78, 2001.

[8] J. W. McCormick, "Software engineering education: On the right track," *ACM SIGAda Ada Letters*, vol. 20, no. 3, pp. 41 – 49, 2000.

[9] V. L. Cohen, "Electronic-portfolios as cognitive tools in a teacher education program," (Lisbon, Portugal), International Conference on Multimedia and ICT in Education, ICTE 09, Apr. 2009.

[10] MediaWiki.org, *MediaWiki, http://www.mediawiki.org/wiki/MediaWiki*, Jan. 2007.

[11] D. A. Kolb, *Learning styles and disciplinary differences*, ch. The modern American College, pp. 232–255. San Francisco: Jossey-Bass: A. Chickering & R. Havighurst Eds, 1981.

[12] L. F. Paulson, P. P. R., and M. C., "What makes a portfolio a portfolio?," *Educational Leadership*, vol. 48, no. 5, pp. 60–63, 1991.

[13] C. Plumb and C. Scott, "Outcomes assessment of engineering writing at the university of washington," *Journal of Engineering*, vol. 91, no. 3, pp. 333–338, 2002.

[14] A. Turns, J., R. C., Adams, and T. Barker, "Research on engineering student knowing: Trends and opportunities," *Journal of Engineering Education*, pp. 27–40, Jan. 2005.

[15] G. R. Mayer, J. B. Weinberg, and X. Yu, "Teaching deliberative navigation using the lego rcx and standard lego components," *IEEE Robotics & Automation Magazine*, vol. 10, June 2003.

[16] J. B. Weinberg and X. Yu, "Robotics in education: Low-cost platforms for teaching integrated systems," *IEEE Robotics & Automation Magazine*, vol. 10, pp. 4–6, June 2003.

[17] S. H. Kim and J. W. Jeon, "Introduction for freshmen to embedded systems using LEGO mindstorms," *IEEE Transactions on Education*, vol. 52, no. 1, pp. 99–108, 2009.

[18] S. Iida and S. Yuta, "Control of vehicle with power wheeled steering using feedforward dynamics compensation," in *Industrial Electronics, Control and Instrumentation, 1991. Proceedings. IECON '91., 1991 International Conference on*, pp. 2264–2269 vol.3, 1991.

[19] D. L. Maskell and P. J. Grabau, "A multidisciplinary cooperative problem-based learning approach to embedded systems design," *IEEE Transactions on Education*, vol. 41, May 1998.