PLC and Petri Net Training Based on a 3D Virtual Car Park Modelling and Control

José L. Lima[*], José C. Gonçalves[*], Paulo G. Costa[+] and A. Paulo Moreira[+]

[*] Polytechnic Institute of Bragança, Campus of Santa Apolónia, 5301-857 Bragança, Portugal
[+] Department of Electrical Engineering and Computers in Faculty of Engineering of University of Porto, 4200-465 Porto, Portugal

Email: jllima@ipb.pt, goncalves@ipb. pt, paco@fe.up.pt, amoreira@fe.up.pt

Abstract

This paper describes a PC application, behaving as a virtual maquette, that emulates a three-dimensional real world where an external PLC controls its behaviour receiving feedback from the environment. The developed interface connects the PLC to the PC. The virtual maquette is composed by a car parking where user requests gate open order while the PLC supervises gate open/close command, parking availability lights and free parking places. The virtual maquette also supports a Petri net viewer where the car park behaviour is modelled. State changes and resources allocation can be seen in real time. The virtual maquette supports different ways of control, allowing several simultaneous student groups to work on different solutions.

Keywords: Computer Aided Engineering, Control Engineering, Laboratories and Petri Nets.

1 Introduction

The Bologna's declaration introduces, in the European Union Education Space, a significative change in the learning process, by changing the focus from a paradigm oriented to the transmission to a paradigm oriented to the learning process. However, it is necessary to go further by introducing new tools that motivate and involve teachers and students in the learning process (Bourne 1995). Individual learning is the base concept and practice is nowadays a common answer to industry requirements (Kroumov 2003).

One of the graduate automation lessons skills is to prepare students to develop PLC (Programmable Logic Controller) programs. The real PLC applications are applied in several industrial designs (Culley 2001). The student PLC programs can usually be tested in laboratories resorting to maquettes. A classical 2D maquette connects to the PLC through wires and basically contains several switches and lights. A 3D maquette, based on hardware, is easier to perceive but harder to control and prototype. The movement of objects that emphasize the model and capture the attention of students is a difficult task in a hardware based solution. Even more, usually there are several groups, forcing the use of more than one maquette at the same time.

A virtual maquette is a tridimensional reconstruction of a model (industrial parts, buildings, cars , etc.) inside a computer containing realistic textures, lightning, different views stretching the distance from virtual to real (Foley 1995).

The developed virtual maquette is composed by a PC application, reducing some hardware problems and allowing an attractive visualization. The student programming task is done resorting to development tools provided by the PLC manufacturers. This approach is better than a stand-alone simulation because allows students to practice with a real PLC, accessing to real hardware.

The developed car parking maquette scene is presented in Figure 1.

This paper is organized as follows: Initially, the virtual maquette, its world behavior based on a Petri net model and hardware interface are described. In the next section, the PLC programming issues are presented, where a first approach and a sample solution are shown. Finally, last section rounds up with conclusions and future work.

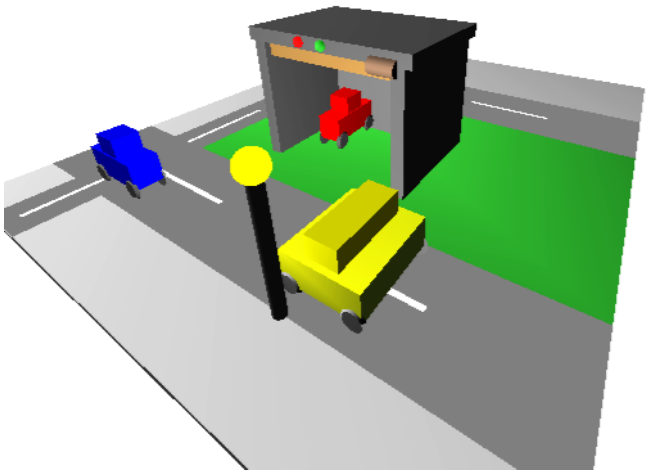

Figure 1: Developed virtual maquette

2 Virtual Maquette Description

The developed virtual maquette gathers the advantages of a classical maquette and some extra features supporting students to test their work. The disadvantages presented in a 3D classical maquette such as hardness to prototype and the hard control due to some hardware limitations are not presented in a 3D virtual maquette. It is also a low cost solution because the software can be easily changed to different requisites and distributed to several work groups. These advantages are not presented in solutions based only on hardware.

The developed virtual maquette is composed by three cars, two of them are small cars and another is a big one, a garage, a road, public artificial (which can be controlled) and day light (which can be sensed) providing a more realistic environment than a conventional maquette. The parking availability can be shown by a red and a green light presented in the garage top. The 3D scene, the zoom and the move around the world feature emphasizes the appearance (Tan 2006). It is desired to control the gate command (open and close), parking availability lights and street illumination by gathering information from the environment. The closed loop is achieved resorting to an external PLC (where students develop the Ladder Program) connected through the PC. The interface joints the PLC and the PC through the parallel port as presented in Figure 2.

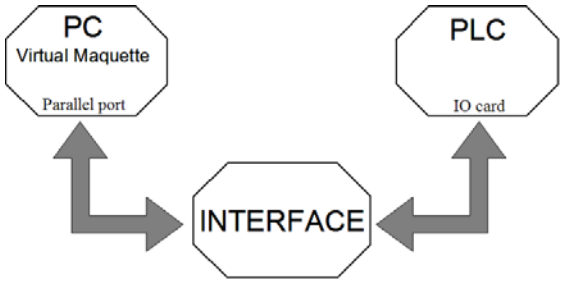

Figure 2: Virtual maquette connection

The developed interface, which connects the PC to the PLC, is presented in Figure 3. The PLC (at the left) has onboard inputs and outputs where the black wires are connected. Otherwise, an external Input/Output card should be attached to the CPU of the PLC. On the right side, the interface connects the PLC to the PC parallel port. The interface is further presented in subsection 2.3.

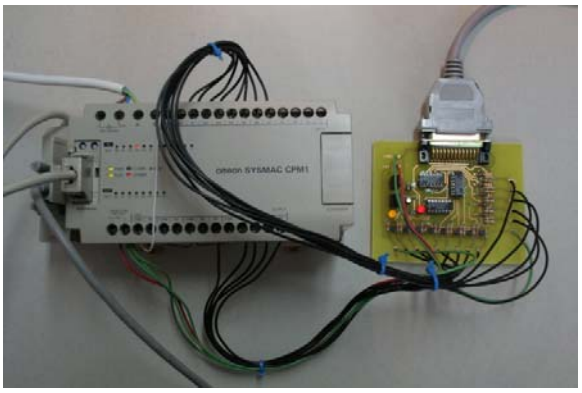

Figure 3: PLC - PC Interface

The virtual maquette provides to the PLC (CPM1 from Omron (Omron 2009)), in this example, the boolean status of the eight environment variables as presented in Table 1. The PLC Addresses with its symbolic names and the corresponding PC Parallel Port bits are also presented. Symbolic names allow programmers to easily decode the developed software as illustrated in the subsection 3.1 of PLC Programming Example. The same symbolic names are also used in the Petri net model and Grafcet example, further presented in this paper.

Table 1 Virtual maquette outupts map

Virtual Maquette Status	PLC Address	PC Parallel Port bit	Symbolic Name
Big car open door request	0.0	D0	BOP
Small car open door request	0.1	D1	SOP
Sensor of gate opened	0.2	D2	SGO
Sensor of gate closed	0.3	D3	SGO
Sunlight presence	0.4	D4	SLP
Small car 1 parked	0.5	D5	SC1P
Small car 2 parked	0.6	D6	SC2P
Big car parked	0.7	D7	BCP

The PLC must control the Boolean status of the five actuators in the virtual maquette as presented in Table 2. The PLC addresses with its symbolic names and the corresponding PC parallel port bits are also presented. The presented tables are included in the virtual maquette application, when requested by user.

Table 2 Virtual maquette inputs map

Virtual Maquette Actuator	PLC Address	PC Parallel Port bit	Symbolic Name
Red availability light	10.3	S3	RAL
Green availability light	10.4	S4	GAL
Street illumination switch	10.5	S5	SL
Close door command	10.6	S6	CDC
Open door command	10.7	S7	ODC

The virtual maquette application software is presented in Figure 4. It was developed in Object Pascal language resorting to GLScene, an OpenGL 3D library. It provides visual components and objects allowing description and rendering of 3D scenes (Glscene 2009).

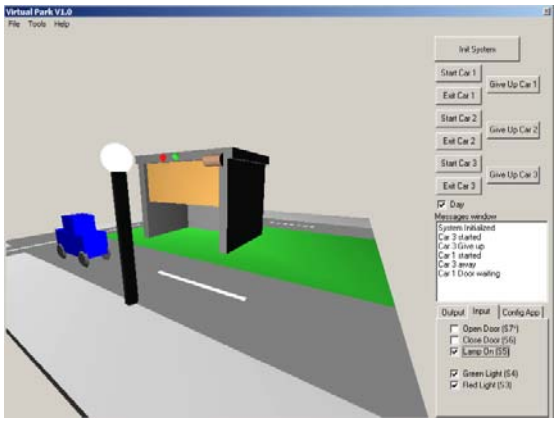

Figure 4: Virtual maquette application

2.1 World Behavior

The virtual maquette control must obey to some requisites. The garage capacity allows either two small cars or just a big one. The parking availability signal (red light) must be turned on in the following circumstances: A small car is parked and the big one requests the open door command or the big car is parked and a small one requests the open door command. Otherwise, the parking availability signal (green light) should be turned on when a car can be parked and the gate should be opened. The gate status is closed by default. If a car is getting in or leaving the park, the gate must be completely opened. When the gate is opened, the open command should be turned off, avoiding motor damage, just as well in closing operation. The street light must be turned on at nightfall. This example is important since allocation of shared resources (park places) is an essential issue in automation systems.

The cars movement is autonomous. User starts-up the desired car, which moves until the garage entrance and remains static waiting for the gate to be opened. Once opened, the car gets into the garage with an autonomous movement too. Otherwise, user can cancel the parking operation guiding the car to move away.

These actions must be provided by the PLC, based on the developed software as a student challenge task. If a PLC program malfunction occurs, the virtual maquette remains still and presents a visual warning informing the fault to the student.

2.2 Petri Net Viewer

A Petri net is a graphical and mathematical modeling tool. It consists of places, transitions, and arcs that connect them. Input arcs connect places with transitions, while output arcs start at a transition and end at a place. Petri nets have been originated from Carl Adam Petri's doctoral dissertation "Communication with automata" in 1962 (Petri 1962). He described, using a net, the casual relationships between events in a computer system. Further developments by A. W. Holt and others illustrate how Petri nets could be used to model and analyze systems of many concurrent components (Zhou 1999).

A reference tutorial paper was given by Professor T. Murata in 1989, which comprehensively presented properties, analysis, and applications of Petri nets and a list of references of significance (Murata 2009).

The virtual maquette contains a Petri net viewer, where the car park behavior is modeled, supporting students a better understanding of what is intended for their PLC control software in an early stage. It is also important, in a final stage, to monitor if their software is working as requested. The Petri net viewer was developed specifically for this software, having in mind the proposed requisites in previous subsection.

Students can monitor, in real-time, the several states, having special attention to the allocation of the shared resources. The initial state is presented in Figure 5, where the shared resources are shown as available represented by two tokens. The big car, in order to park, consumes two tokens in opposite of the small ones.

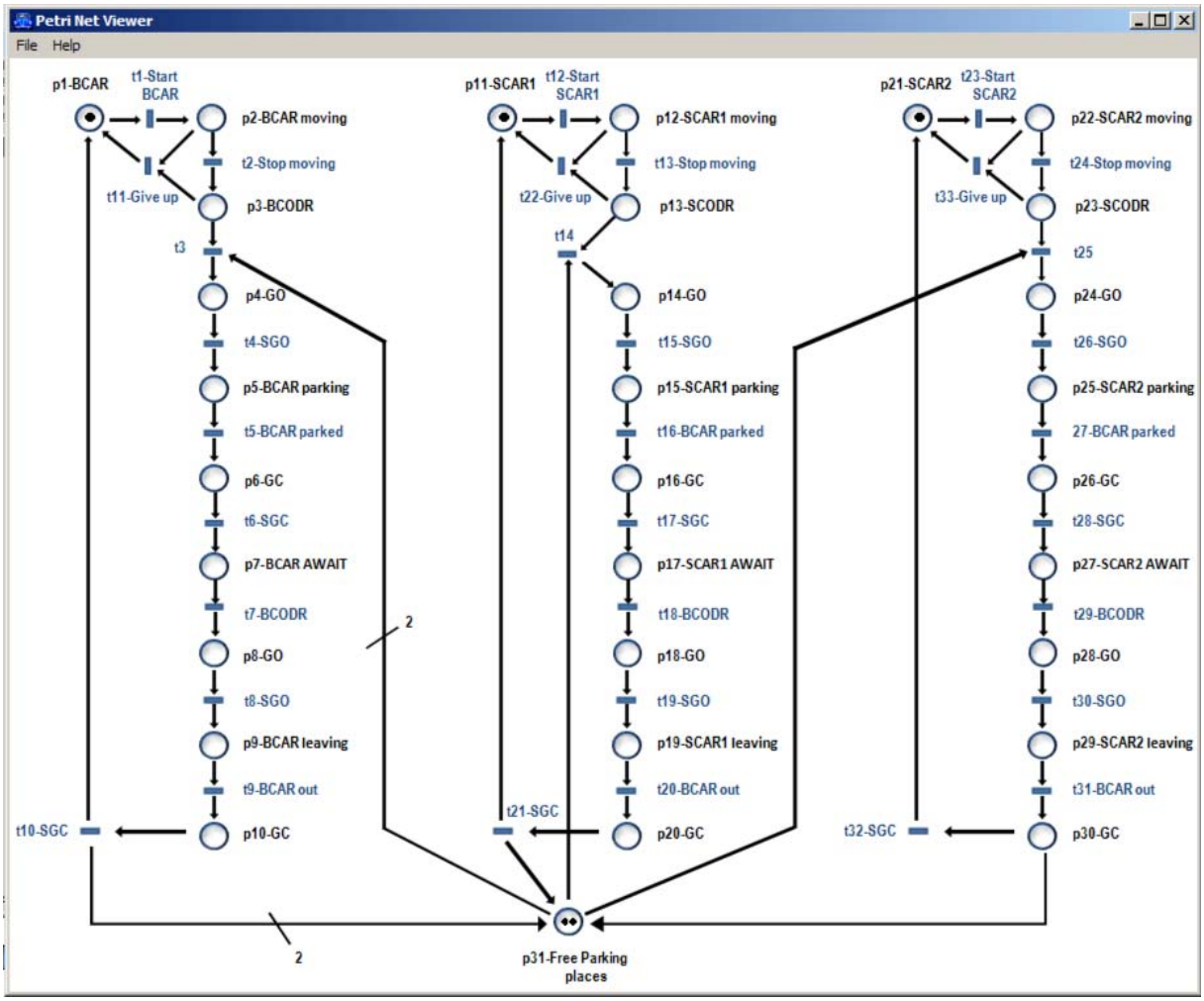

Figure 5: Car park Petri net viewer screenshot

This item is visible in the weights of some arches of Petri net when the shared resources are consumed or when they are reinstated. The used symbols in the Ladder program are the same as the automated system modeling in Petri net.

2.3 Hardware Interface

The basic objective of this interface is to adapt two different Voltage levels. The PLC standard Voltages are 24V whereas, the PC parallel port Voltages are 5V. It is basically composed by digital logic circuitry and voltage limiters as presented in Figure 6 and Figure 7.

The PC parallel port is protected by a 5V1 Zener diode. Then, a buffer gate is applied allowing by this way the conversion of 0V and 24V inputs (from the PLC) to 0V and 5V outputs to be applied in the status address of the PC parallel port (Gadre 1998).

The PC parallel port output data address bits connect to a buffer to achieve the base current of the NPN transistor, working as an open-collector switch. In both situations a pull-up resistor is used.

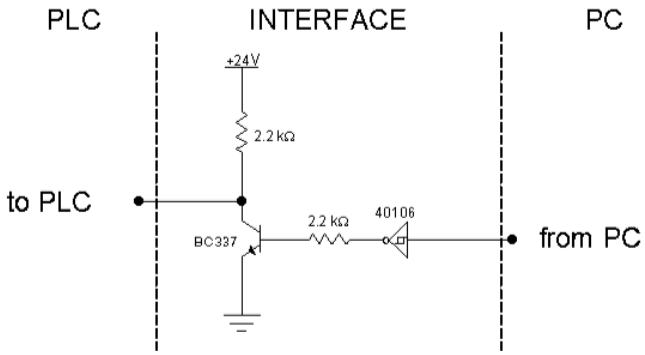

Figure 6: PC to PLC interface

On the one hand, for the interface output, the pull-up resistor places 24V to the PLC when the transistor is switched off and approximately 0V when in saturation.

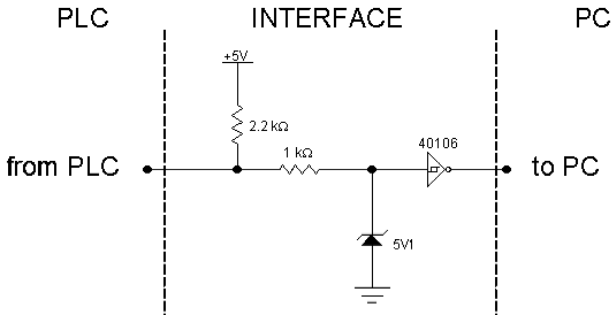

Figure 7: PC to PLC interface

On the other hand, a pull-up resistor defines the default state supporting the PLC relay and open collector types (Pallas-Areny 2001). The impedance adaptation is guaranteed applying a NOT gate for both input and output signals.

The developed interface circuitry is shown in Figure 8 that replicates eight outputs and five inputs. The parallel port output signals are implemented in the data register whereas the input signals are implemented in the status register, allowing a standard parallel port protocol (Gadre 1998).

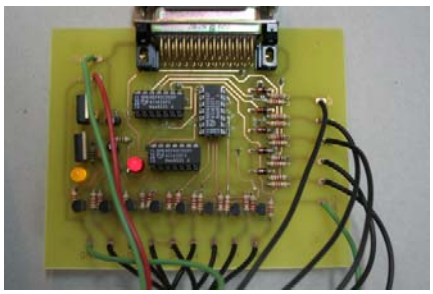

Figure 8: Interface Printed Circuit Board

This card can also be used for different applications as a standard I/O interface card through parallel port.

3 PLC PROGRAMMING EXAMPLE

3.1 First PLC Programming Approach

Ladder logic is a method of drawing electrical logic schematics. It was originally applied to describe logic made from relays. Ladder logic is widely used to program PLCs, where sequential control of a process or manufacturing operation is required.

Ladder logic is useful for simple but critical control systems, or for reworking old hardwired relay circuits. For complex systems, Petri net or Grafcet tools should be used to develop and to translate to ladder logic or structured text in a systematic way once programmable logic controllers became more sophisticated.

An introductory example of a ladder logic program that controls one of the features of the virtual maquette is presented in Figure 9 (Dunning 2006).

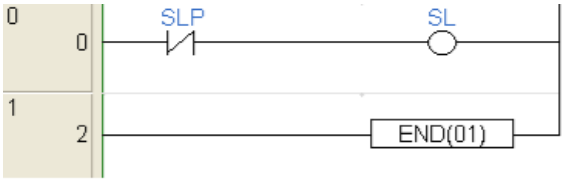

Figure 9: Introductory Ladder Programming Example

If Sunlight presence (SLP) is detected then the Street light (SL) is turned off. This simple Ladder program includes only two *rungs* - a Ladder logic line. Nowadays, in real PLC industrial applications, there may be hundreds of them.

An other solution can be done resorting to Grafcet language. The Grafcet standard is used as a mean to reduce the complexity of large programming tasks and to formalize the state-machine structure often used in the control of discrete-event dynamic systems (R. a. David 1992). A Grafcet is a graph having two types of nodes, i.e., steps and transitions (R. David 1995). The Grafcet, Petri net and structured text interconnection is possible and is discussed in (G. Frey 2000) and (G. a. Frey 2000).

The Grafcet solution for the same PLC program is presented in Figure 10, where the light problem is solved. Once again, the SLP sensor allows to change between state 1 and state 2 turning on and off the SL actuator (street illumination).

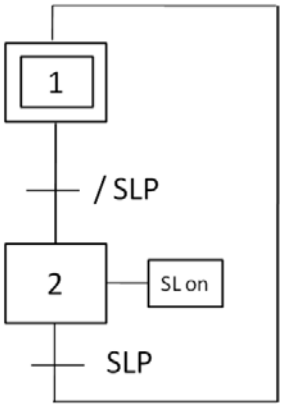

Figure 10: Introductory Grafcet Example

3.2 Programming Challenge

The forward program, presented in Figure 11 and its brief following explanation based on its symbolic names, is an example of one small car parking request (Rohner 1996).

- The SOP (small car open door request) is only attended if BCP (big car parked) is false.;
- If SOP is attended then the ODC (open door command) is activated;
- The gate is requested to open until it is completely open: SGO (sensor gate open);
- The green parking availability light (GAL) is turned on for 3 seconds when gate is opened.
- When the car is parked, the closed door command (CDC) is activated until it is completely closed (SGC)

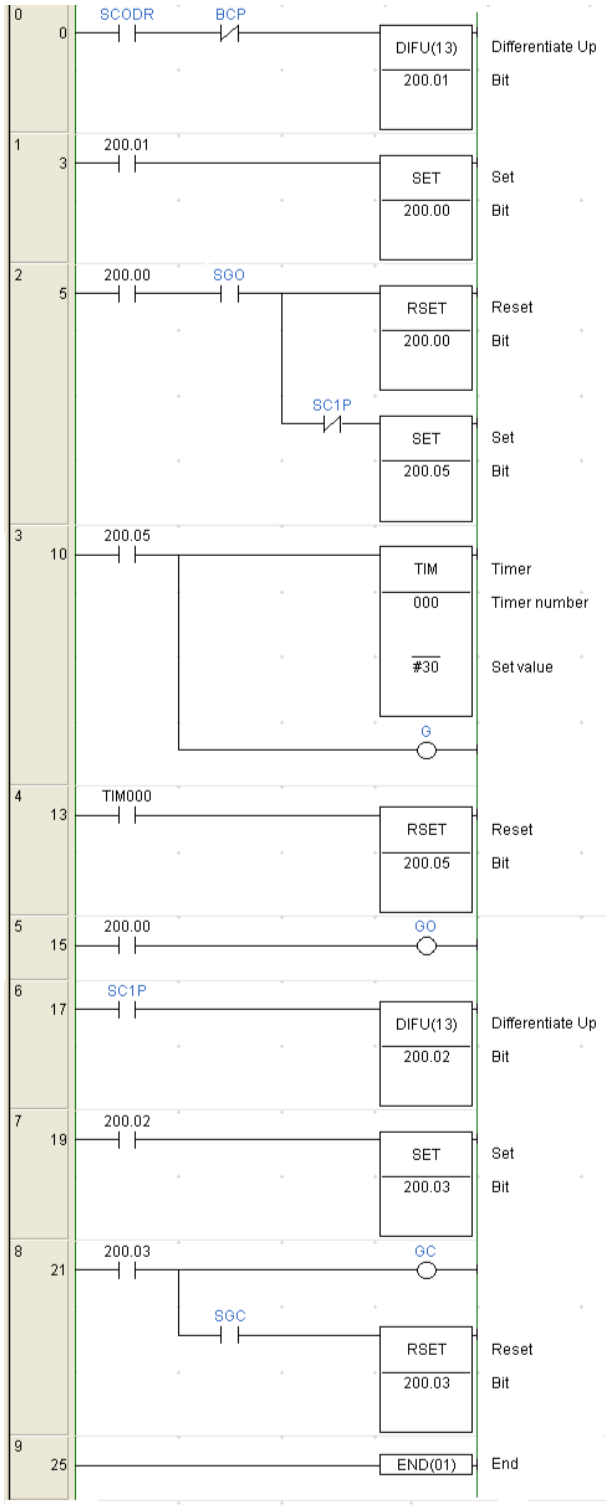

Figure 11: Challenge Ladder Diagram Solution

This simple program concerns only one small car and has different solutions for the same conduct. The red parking availability light should be turned on when parking places are unavailable. For three cars there must be a more complex Ladder diagram that remains to students having in mind important concepts such as shared resources.

4 CONCLUSION AND FUTURE WORK

The developed system, composed by a PC application and a hardware interface, allows students to develop their programs in the PLC, interacting with the virtual maquette while actions are displayed in the PC monitor, based on 3D realistic pictures. The cars movement, the zoom feature, the camera positioning freedom and the easy interact commands captivate students attention and increases their learning motivation. The Petri net viewer is an important issue, because it allows to enhance students perception of PLC program goal: mainly the initial and evolution states.

As future work, the presented maquette can be joined in other lectures such as embedded systems allowing students to control the virtual park using a microprocessor or microcontroller programmed in assembly or high level languages. In the PLC lessons context, an auto evaluation can also be created for this virtual maquette where each student has an established time to solve and experiment the PLC program and, at the end, the software application verifies the final behavior punishing each fault and giving a final classification.

References

Bourne, J. R., Brodersen, A., and Dawan, M. *The Influence of Technology on Engineering Education.* CRC Press Inc., 1995.

Culley, S. "Design applications in industry and education." *International Conference on Engineering Design.* 2001.

David, R. and Alla, H. *Petri Nets and Grafcet: Tools for Modelling Discrete Event Systems.* Prentice-Hall, Inc., 1992.

David, R. "Grafcet: A powerful tool for specification of logic controllers." *In IEEE Transactions on Control Systems Technology*, 1995.

Dunning, G. *Introduction to Programmable Logic Controllers.* Delmar Thomson Learning Inc., 2006.

Foley, J. D. *Computer graphics: Principles and Practices.* Addison Wesley Professional, 1995.

Frey, G. and Litz, L. "Formal methods in PLC programming." *Proceedings of the IEEE Conference on Systems Man and Cybernetics.* 2000.

Frey, G. "Automatic implementation of petri net based control algorithms on plc." *In Proceedings of the American Control Conference.* 2000.

Gadre, D. V. *Programming the Parallel Port: Interfacing the PC for Data Acquisition &Process Control.* CMP Books, 1998.

Glscene. *Glscene.* 2009. http://glscene.org/.

Kroumov, V., Shibayama, K., and Inoue, A. "Interactive learning tools for enhancing the education in contol systems." *33rd ASEE/IEEE Frontiers in Education Conference.* 2003.

Murata, T. *Petri nets: Properties, analysis and applications.* 2009. www.cs.unc.edu/ montek/teaching/spring-04/murata-petrinets.pdf.

Omron. *Omron automation products and services.* 2009. http://www.omron247.com.

Pallas-Areny, R. and Webster, J. G. *Sensors and Signal Conditioning.* John Wiley & Sons, 2001.

Petri, C. A. "Kommunikation mit Automaten." PhD. thesis, Natural Sciences, Technical University Germany, 1962.

Rohner, P. *Plc: Automation with Programmable Logic Controllers.* University of New South Wales Press, 1996.

Tan, K. C. and Wang, L. *Modern Industrial Automation Software Design.* Wiley-IEEE Press, 2006.

Zhou, M. and Venkatesh, K. *Modeling, Simulation and Control of Flexible Manufacturing Systems: A Petri Net Approach.* World Scientific Publishing Company, 1999.