Development of a Multi-DOF Robotic Controller for Academic Purposes

Francisco Mota Muñoz and Sergio Martínez de la Piedra

Ingeniería Robótica, Universidad Politécnica de Guanajuato, Cortazar 38483, México

Received: April 21, 2015 / Accepted: May 06, 2015 / Published: May 25, 2015.

Abstract: This article presents the development of a robotic controller for technical training, academic teaching, and research. The controller was designed to interact from 1 to 6 DOF (degrees of freedom) serial robotic arms, actuated by brushed DC (direct current) servomotors equipped with incremental encoders. Controller architecture is based on four components: a processor, a reconfigurable FPGA (field-programmable gate array), measurement I/O hardware and software. Functionality of the robotic controller has been proved by means of the interaction with an SCARA (selective compliance assembly robot arm). The proposed controller presents the potential to teach technical courses (like robotics, control, electronics and programming) and to implement and validate advanced control algorithms.

Key words: Open architecture, robot control, FPGA control implementation, academic robot controller.

1. Introduction

This work responds to an increasing requirement of the didactical resources for the understanding of the design, function, performance, and operation of the industrial manipulators used at a university level.

There are a great number of didactical products working with mobile robotics. These products present a large variety of approaches and levels of interaction and abstraction, meaning that they can be used for the comprehension of robotics, from both the basic to university and engineering academic levels.

The platforms of companies as LEGO (with its Lego Mind Storm), Pitsco (with Tetrix), VEX robotics, and National Instruments (with NI DANI I and II), are well known by the majority of those working in robotics education activities. With such platforms, it is possible to teach and understand basic concepts of electronics, mechanics, programming and control, from basic to advanced levels. On the other hand, for industrial robotics, there are hardware-software platforms that work with robotic manipulators such as: Rhino, Brutus of Pitsco, Scorbot of Intelitek, KUKA and Jupiter of Amatrol. Due to these robotic arms, it is possible to understand the basic function of sensors, actuators, transmission systems, etc. Their programming is very simple and in some cases, they allow us to imitate and understand the real behavior of conventional industrial robots. However, they are useless when it is necessary to develop essential academic concepts such as: kinematic, dynamic or the proper control of the manipulator.

The main limitation of such equipment is its close-architecture, which does not allow modifying, adapting or increasing their functionality or performance of the robotic arms. Manufacturers keep the information related to the kinematic and dynamic model for themselves. It is not possible to know their control laws, to adjust gains, or make the trajectory planning. As a result, it is not possible to know the state of the different variables of the manipulator during its operation such as position, speed, energy consumption, etc.

There are open platforms as the robot PA-10 with 6
or 7 DOF (degrees of freedom) (GDL) working with open architecture. The high levels of flexibility and adaptability managed by these systems are the reasons why they have been widely adopted in research. Nevertheless, their use is limited due to their high acquisition costs that could be over 20 times higher than the average educational robot arms. There are some products dealing with educational software, such as the Robotic Toolbox developed by Matlab and LabVIEW, “MATLAB toolboxes: robotics and vision for students and teachers” [1]. This software is a completely free library with which students can simulate the kinematic and dynamic of robotic manipulators under a virtual environment. Another product, The MTIS (Matlab Toolbox for the Intelitek Scorbot) [2], presents the development of a free access library for Matlab to send commands from the USB port to the controller of the Intelitek’s Scorbot ER-4U robot. This product allows going from simulation to direct interaction with the manipulator, making it possible for teachers and students to enrich their practice with algorithms such as vision or artificial intelligence. Nevertheless, all this work is limited to the use of the Scorbot robot and its original controller, which does not allow modifications to its control laws.

For these reasons, there is a need to integrate a modular robot controller, which can be reconfigurable and reprogrammable, based on the concepts of open architecture and implemented over a last generation hardware-software platform.

Interest in the development of controllers known as “open” began to receive increased attention in the eighties. These controllers, designed to facilitate the implementation and integration of technology provided by different sources (control vendors (suppliers), machine tool builders and even end-users), were developed to work within advanced manufacturing systems. The increased interest in open controllers was due in large part to the increasing number of special purpose machines with a high level of automation, as well as by the increasing developing costs of software used by these machines [3].

In literature, there are many studies dealing with open platforms. In 1992, important advances were made in the subject within the work carried out under the European project called OSACA. Other important advances in this field were the results derived from the work done by the OSEC and OMAC projects, carried out by American and Japanese researchers, respectively. In 1998, the University of New York created MOSAIC, the first open architecture system ever developed. One country that is making an important effort to develop this concept is China. Its experts have devised several controllers working under different software [4].

Despite all these studies, there is still no precise definition of this concept, which is commonly understood as a synonym for modularity, portability, extendibility, scalability and interoperability [4].

A key element in the development of advanced manufacturing systems is the industrial robot manipulator. In Ref. [5], a study of the PA10-7C manipulator is presented. These findings highlighted how this manipulator completely changed the view of conventional manipulators due to its operation under open architecture.

This project started at the beginning of 2013 and was designed to be done in three different stages. In the first one, an independent control system was implemented using a PAC (programmable automatic control) of National Instruments, which is used to make the interface between the sensors and the encoders. In the second stage, the fundamental work required to do a real-time control of the manipulator, was done. The work made in this stage, mainly included the analysis of forward kinematic as well as the point to point trajectory interpolation. Finally, in the third stage, the analysis of inverse kinematic was designed and programmed as well as the lineal interpolation for the manipulator.

The main idea is that this platform can be seen as a transparent box to facilitate the teaching and
developing of industrial robotics projects, and making the equipment available to universities and research centers all over the country.

2. Robotic Controller Design

The controller design requirements are based on the concepts of open architecture. Under these concepts, the user not only has the opportunity to make changes to the original functions of the system, but also to develop new ones, thus allowing the controller to be used with both teaching and research purposes. Another key requirement for the design of the controller is that it could interact with other serial robot manipulators put into action by PMDC motors. All of this was done with the idea of allowing universities and research centers to use their existing robots manipulators or to develop their own at a low cost. A brief synthesis of the main features to be implemented in this work is listed below.

2.1 Features of a Robotic Industrial Controller

For the design and development of the controller, the features listed below are considered the basic functions and features of any robotic controller:

- Manipulator’s power supply and components (sensors, drivers, actuators, etc.);
- Processing of joint variables (position, speed and acceleration) through incremental quadrature encoders;
- Independent joint closed loop control;
- User interface;
- Programming by teaching points;
- Point to point, linear and spline interpolation;
- Manual and automatic operation modes;
- Use of Joint and Cartesian coordinates.

2.2 Features of an Open Architecture Controller

To allow the control being used for educational and research purposes, it is necessary to define the following characteristics typical of systems that open architecture has:

- Have a hierarchical structure with several levels of control;
- Have a general purpose programmable interface;
- Perform the communication between the different levels of control through a standard interface;
- Have the ability to operate under different control laws;
- Be integrated with modular, inter-changeable and standards elements;
- Be flexible to be adapted to future applications and requirements;
- Be reconfigurable, allowing the development of new functions;
- Be scalable to increase both its period of life and its applications number.

3. Proposed Control Architecture

The objective of the work presented in this article is the development of a platform that can be seen a transparent box to facilitate the teaching and development of industrial robotics projects, by the update the existing equipment at the universities and research centers over the country.

The proposed control architecture is structured at three levels:

- User interface;
- Operation control;
- Movement control.

3.1 Hardware/Software Architecture Approach

After reviewing several articles related to the rehabilitation of obsolete industrial robots, different configurations of HW/SW were found used by other authors in the development of controllers [6-16].

Considering the descriptions of the configurations types used in the development of the robot controller, ten different alternatives for the development of the open architecture controller were raised, where the main differences between these alternatives lie in the software used for such development. In order to select the best alternatives, seven different criterions were
defined: programmable interface, development complexity, ability to operate under different modes and control laws, system performance, reliability, and scalability. After analyzing the different solution alternatives, the NI platform was selected. This platform is integrated by a Compact RIO which is a reconfigurable hardware for control and acquisition of data, and by a LabVIEW high level graphic programming language for the development of analysis applications, simulation and control.

Relevant aspects and advantages presented by this platform are listed below:

- It is a robust platform orientated to the development of instruments and control systems;
- It incorporates a great number of tools, libraries and functions for the signal processing and the control of systems;
- Its graphic programming language is easy to understand, allowing the control system to be a user-friendly system for different engineering purposes;
- There is a great amount of information in the web where it is possible download tutorials, manuals and real applications examples useful as a base for the development of new applications;
- Its programming language helps engineers of any discipline with basic knowledge of structure programming and digital electronics to develop powerful applications.

4. Hardware Selection

The determination of the set of the hardware requirements to implement the system was based on the analysis of the features and the system requirements, as well as their development platform.

At the top level of the control architecture, a conventional PC (personal computer) was used as user interface. This PC uses a non-deterministic (Microsoft Windows 7 ©) operating system. This operating system was selected because it is not necessary for processing tasks, but only for programming, operational and monitoring tasks. The development software LabVIEW © together with Real-Time and FPGA (field-programmable gate array) libraries were installed to this PC. These libraries incorporate a complex set of procedures and functions to develop applications using LabVIEW © thus, a real-time hardware control and the incorporation of reconfigurable FPGA circuits or modules for signal acquisition.

For the second level of the architecture, it has opted to use a Compact RIO line controller ©. The operation of these controllers is based on an embedded PC (processor, volatile memory and data communication ports, etc.) which use the VxWorks operating system. With this operating system, the execution of programs is possible in a deterministic way, developed with LabVIEW ©.

The third level is integrated by an FPGA and data acquisition modules. The FPGA can be programmed with LabVIEW enabling the development of custom-made control and signal acquisition.

Acquisition modules of digital and analogical signals, as well as C series servo-drive modules, were selected to interact with the original sensors and actuators of the robot.

5. Implementation of the Universal Robotic Controller

5.1 First Level of Control: Movement Control

This is the stage where all the elementary tasks of the system are performed, such as signal conditioning, data acquisition, and the control of movements of the robot’s joints. This stage has been implemented in the FPGA of the 9074 NI Compact RIO. At this FPGA, decoding functions were programmed, as well as those of control loop and PWM (pulse width modulation).

The decoding function, takes the A and B encoder’s signals to obtain the angular position, speed and acceleration of the selected joint. The frequency used to take this information is set at the 40 MHz to prevent the loss of pulses. To implement this function,
Quadrature Encoder dX Method FPGE was used in the SCTL.vi” free distribution library. At this stage, the simplest strategy control was implemented, consisting in considering that the manipulator is integrated by “n” independent systems (n joints), where each of this systems is controlled as single input and output system (SISO) [17].

The loop control has been implemented by the library “PID (FPGA) Express”—an algorithm PID where the proportional, integral and derivative gains are expressed as a fixed point numbers, with a word size of 16 bits with an integer part of 8 bits [18].

Finally, a PWM was implemented to control the flux of current that is applied to the robot accumulators.

All functions and libraries mentioned here can be modified to be adapted to the particular requirements of the application. In Fig. 1, the control diagram of the movement implemented at the FPGA is showed.

5.2 Second Level of Control: “Operation Control”

This stage is responsible for interpreting, managing and executing commands from the “User Interface”. Other functions made at this stage are the calculus of direct, inverse, and differential kinematics, trajectory generation and control set-points for the first level of control is performed.

For the development of this stage, the machine control architecture was used. This architecture, based on standard OMAC [19], is a set of libraries created by National Instruments for performing machine control applications via the Compact RIO. This stage has been implemented by means of programming and editing libraries which are executed in real time by the processor of the Compact RIO NI 9074, under a master-slave configuration. At this configuration, the master is the user interface where commands to be executed are generated and the slave is the motion controller.

The main function within the second stage of control has been implemented by a state machine that manages the command from the user interface. This state machine consists of the states: Idle, Enabledrives, Getactposition, DeltaJtMove, DeltaWMove, JtMove, WMove and Busy. All of these were determined based on the work presented in Ref. [2].

In Fig. 2, the “Operating Control” module diagram is displayed. This kind of structure allows increasing or modifying the current controller functionality, adding, expanding or modifying statements. Gains, algorithms directly and inverse kinematics, the Jacobean Matrix and mechanical properties of the robot can be replaced to adapt the use of the controller to any configuration of serial manipulator robots. These functions have been implemented through a special node which can be programmed using ANSI C.

Table 1 specifies the functionality that each state has within the system.

5.3 Third Stage of Control: “User Interface”

At the third stage of control, the “user interface” was made. This stage is in charge of the training of the robot, the program editing, the tasks specifying, the command execution, and the monitoring system. This interface was designed by emulating the characteristics and function observed in the teach pendant of a conventional industrial robot.

To implement the third level, the design pattern of control events was used. With this pattern, it is possible to make a very readable programming which is easy to interpret [20].

6. Experimental Results

To verify the correct functionality of the controller and the characteristics and properties of its open architecture, the controller was tested with three manipulators of different morphology; 3 DOF Micro XYZ arm (designed and manufactured by professors of the UPG), 4 DOF SCARA (Selective Compliance Robotic Arm) robotic arm (Amatrol Jupiter XL), and 6 DOF arm anthropomorphic (Intellitek Scorbot ER-4u).
Table 1  Functionality of each state in the system.

<table>
<thead>
<tr>
<th>State</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Idle</td>
<td>Not robot action it is required</td>
</tr>
<tr>
<td>Enabledrives</td>
<td>Turn on/off robot drives</td>
</tr>
<tr>
<td>Getactposition</td>
<td>Saves joint encoder actual position</td>
</tr>
<tr>
<td>DeltaJtMove</td>
<td>Incrementally moves the end effector from current joint position by an specific factor</td>
</tr>
<tr>
<td>DeltaWMove</td>
<td>Incrementally it moves the end effector from current workspace position by an specific factor</td>
</tr>
<tr>
<td>JtMove</td>
<td>Moves the end effector, from current point A to B in joint space</td>
</tr>
<tr>
<td>WMove</td>
<td>Moves the end effector from current point A to B in workspace</td>
</tr>
<tr>
<td>Busy</td>
<td>Robot manipulator moving</td>
</tr>
</tbody>
</table>
All the robot manipulators used in the tests have something in common: They are all activated by PMDC motors and all have quadratic encoders for joint, velocity and acceleration feedback.

6.1 Test Procedure Description

Independent PD workspace control laws were implemented, and the results obtained by working with the first joint of the SCARA robot are presented here. Tests and results of the other robots are similar and will be presented in future works.

6.2 Gain Tuning

For the first joint, the follow gains values of the joint position PID controller were calculated: proportional gain $K_p = 3 \ N\cdot m$, Integral gain $K_i = 0 \ s^{-1}$, and derivative gain $K_d = 16 \ s$.

After the manual tuning of the gains in the first joint of the robot Jupiter XL, a first performance test was realized. This test consisted in exciting this joint with a unitary step input, moving the joint from 0 rad to 0.175 rad, to evaluate the regulation control.

Fig. 3 shows the graph of response of the Jupiter XL robot joint articulation versus time, to a unit step input type of magnitude of 0.175 rad.

Fig. 4 shows the behavior of the position error of the joint during the development of the test.

Fig. 5 shows the joint energy consumption during the development of the test.

Fig. 6 Temporary joint position evolution of the first Jupiter XL robot joint during the cycloid path following from 0 rad to 0.175 rad in a second.

It is important to point out that the position error reached a zero value after about 250 ms, and it is stabilized after a period of 700 ms. This fact can be corroborated by using the energy graph where the amplitude of the first current pic shows the effort being carrying by the actuator to reduce the error and which decrease as the system stabilizes.
Development of a Multi-DOF Robotic Controller for Academic Purposes

7. Conclusions

The robots have become almost indispensable in industry and manufacturing processes and their lifetime is relatively large. However, its controllers have a lifetime much more inferior due to the technological advancement of communication. When the controller suffers some damage, very often all the mechanical system of the robot is removed at the industries. In the majority of the cases, the industrial robots in this situation are treated by the industry as “junk”, wasting the good condition of the mechanical system and its remaining useful period of life. Rehabilitation of industrial robots controller allows those robots to be used again, not only for industrial purposes, but also for educational or research ones.

This paper offers an alternative to the industry, as well as to the research institutions of using commercial and standardized technology (in this case, National Instruments technology), acquired at a low price in comparison with the price of purchase a new robotic unit. In this work, it has been experimentally proved that the implementation of open architecture controller guarantees tracking trajectories and robot positioning. Therefore, the use of the platform proposed in this paper allows for the industrial robot to conserve its basic functions (provided by the manufacturer) and in addition, to acquire the advantages provided by the open architecture controller.

References


