View metadata, citation and similar papers at core.ac.uk

brought to you by CORE

provided by IntechOpen

We are IntechOpen, the world's leading publisher of Open Access books Built by scientists, for scientists

4,800
Open access books available

122,000
International authors and editors

135M
Downloads

Our authors are among the

154
Countries delivered to

TOP 1%
most cited scientists

12.2%
Contributors from top 500 universities

CLARIVATE ANALYTICS BOOK CITATION INDEX INDEXED

WEB OF SCIENCE™

Selection of our books indexed in the Book Citation Index in Web of Science™ Core Collection (BKCI)

Interested in publishing with us?
Contact book.department@intechopen.com

Numbers displayed above are based on latest data collected.
For more information visit www.intechopen.com

21

Motion Control of Industrial Robots in Operational Space: Analysis and Experiments with the PA10 Arm

Ricardo Campa[1], César Ramírez[1], Karla Camarillo[2], Víctor Santibáñez[1] and Israel Soto[1]
[1]*Instituto Tecnológico de la Laguna*
[2]*Instituto Tecnológico de Celaya*
Mexico

1. Introduction

Since their appearance in the early 1960's, industrial robots have gained wide popularity as essential components in the construction of automated systems. Reduction of manufacturing costs, increase of productivity, improvement of product quality standards, and the possibility of eliminating harmful or repetitive tasks for human operators represent the main factors that have determined the spread of the robotics technology in the manufacturing industry. Industrial robots are suitable for applications where high precision, repeatability and tracking accuracy are required. These facts give a great importance to the analysis of the actual control schemes of industrial robots (Camarillo et al., 2008).
It is common to specify the robotic tasks in terms of the pose (position and orientation) of the robot's end–effector. In this sense, the *operational space*, introduced by O. Khatib (1987), considers the description of the end–effector's pose by means of a position vector, given in Cartesian coordinates, and an orientation vector, specified in terms of Euler angles. On the other hand, the motion of the robot is produced by control signals applied directly to the joint actuators; further, the robot configuration is usually measured through sensors located in the joints. These facts lead to consider two general control schemes:

- The *joint–space control* requires the use of inverse kinematics to convert the pose desired task to joint coordinates, and then a typical position controller using the joint feedback signals is employed.
- The *operational–space control* uses direct kinematics to transform the measured positions and velocities to operational coordinates, so that the control errors are directly computed in this space.

The analysis of joint–space controllers is simpler than that of operational–space controllers. However, the difficulty of computing the inverse kinematics, especially for robots with many degrees of freedom, is a disadvantage for the implementation of joint–space controllers in real–time applications.

Most of industrial robots are driven by *brushless DC* (BLDC) servoactuators. Typically, this kind of servos include electronic drives which have internal joint velocity and torque controllers (Campa et al., 2008). Thus, the drive's input signal can be either the desired joint velocity or torque, defining the so–called velocity or torque operation modes, respectively. In practice, these drive inner controllers are fixed (they are usually PI controllers tuned with a high proportional and integral gains), and an overall outer loop is necessary to achieve the control goal in operational space.

Aicardi et al. (1995) were the first in analyzing such a hierarchical structure to solve the problem of pose control, but considering an ad-hoc velocity inner loop. Kelly & Moreno (2005) used the same inner controller, together with the so–called *resolved motion rate controller* (RMRC) proposed by Whitney (1969), to solve the problem of operational space control. More recently, Camarillo et al. (2008) invoked some results from (Qu & Dorsey, 1991) to the analysis of a two–loop hierarchical controller using the RMRC as the outer loop and the typical PI velocity controller in the inner loop.

The Mitsubishi PA-10 arm is a general–purpose robotic manipulator with an open architecture, which makes it a suitable choice for both industrial and research applications (Oonishi, 1999). The PA-10 system has a hierarchical structure with several control levels and standard interfaces among them; the lowermost level is at the robot joint BLDC servoactuators, which can be configured either in velocity or torque mode; however, the original setup provided by the manufacturer allows only the operation in velocity mode.

The aim of this work is twofold. Firstly, we review the theoretical results that have been recently proposed in (Camarillo et al., 2008), regarding the stability of the operational space control of industrial robots, assuming inner joint velocity PI controllers, plus the RMRC as the outer loop. Secondly, as a practical contribution of this chapter, we describe the steps required to configure the PA-10 robot arm in torque mode, and include some experimental results obtained from the real–time implementation of two–loop operational space controllers in a PA-10 robot arm which is in our laboratory.

After stating the concerns of the chapter, and recalling the main aspects of modeling and controlling industrial robots (sections 1 and 2) we review the stability analysis of the two–loop operational–space motion–control scheme in Section 3. The description of the PA-10 open–architecture is given in Section 4, while Section 5 explains the hardware and software interfaces that were developed in order to make the PA-10 work in torque–mode. To show the feasibility of the operational–space control scheme in Section 3, we carried out some real–time experiments using the platform described before in a real PA-10 robot; this is explained in Section 6. Finally, concluding remarks are set in Section 7.

2. Modeling and Control of Industrial Robots

2.1 Kinematic and Dynamic Modeling

From a mechanical point of view, industrial robots are considered to have an open kinematic chain, conformed by rigid links and actuated joints. One end of the chain (the *base*) is fixed, while the other (usually called the *end–effector*) is supposed to have a tool or device for executing the task assigned to the robot. As this structure resembles a human arm, industrial robots are also known as robot manipulators.

2.1.1 Kinematics

In a typical industrial robot the number of degrees of freedom of the robot equals the number of joints in the kinematic chain; let n be that number. The configuration of the robot can then

be described by a set of n joint coordinates:

$$\boldsymbol{q} = \begin{bmatrix} q_1 & q_2 & \cdots & q_n \end{bmatrix}^T \in \mathbb{R}^n.$$

On the other hand, let m be the number of degrees of freedom required to specify the task to be executed by the robot. Notice that, in order to ensure the accomplishment of the desired task, then $n \geq m$. If $n > m$ we have a *redundant manipulator*, which has more degrees of freedom that those required to execute the given task.
It is common that the task is specified in terms of the pose (position and orientation) of the robot's end–effector (usually time–varying). In the general 3D–motion case, we require three degrees of freedom for the position and other three for the orientation, thus $m = 6$. Let $\boldsymbol{x}$ be the set of operational coordinates for describing the pose of the end–effector, then

$$\boldsymbol{x} = \begin{bmatrix} x_1 & x_2 & \cdots & x_6 \end{bmatrix}^T \in \mathbb{R}^6.$$

The direct kinematic model of the robot can be written as

$$\boldsymbol{x} = \boldsymbol{h}(\boldsymbol{q}), \tag{1}$$

where $\boldsymbol{h}(\boldsymbol{q}) : \mathbb{R}^n \rightarrow \mathbb{R}^6$ is a function describing the relation between the joint coordinates and the operational coordinates. Furthermore, the differential kinematics equation can be obtained as the time derivative of the direct kinematics equation (1), i.e. (Sciavicco & Siciliano, 2000):

$$\dot{\boldsymbol{x}} = J(\boldsymbol{q})\dot{\boldsymbol{q}}, \tag{2}$$

where $J(\boldsymbol{q}) \in \mathbb{R}^{6\times n}$ is the so–called analytic Jacobian matrix.
To obtain the inverse differential kinematics, we use the right pseudoinverse of $J(\boldsymbol{q})$, which is given by (Sciavicco & Siciliano, 2000):

$$J(\boldsymbol{q})^\dagger = J(\boldsymbol{q})^T \left[J(\boldsymbol{q}) J(\boldsymbol{q})^T \right]^{-1}, \tag{3}$$

in such a way that $J(\boldsymbol{q})J(\boldsymbol{q})^\dagger = I$, being $I \in \mathbb{R}^{6\times 6}$ the identity matrix. Notice that if $n = 6$ then $J(\boldsymbol{q})^\dagger$ reduces to the conventional inverse matrix $J(\boldsymbol{q})^{-1}$.
In the general case, the *inverse differential kinematics* is given by

$$\dot{\boldsymbol{q}} \quad = \quad J(\boldsymbol{q})^\dagger \dot{\boldsymbol{x}} + \left[I - J(\boldsymbol{q})^\dagger J(\boldsymbol{q}) \right] \dot{\boldsymbol{q}}_0 \ \in \mathbb{R}^m \tag{4}$$

where the first term is known as the minimal norm solution of (2), and matrix $\left[\boldsymbol{I} - \boldsymbol{J}(\boldsymbol{q})^\dagger \boldsymbol{J}(\boldsymbol{q}) \right]$ is the orthogonal projector to the null space of $J(\boldsymbol{q})$, $\mathcal{N}(J)$, defined as

$$\mathcal{N}(J) = \{ \boldsymbol{w} \in \mathbb{R}^n : J\boldsymbol{w} = \boldsymbol{0} \}.$$

Any vector belonging to $\mathcal{N}(J)$ has no effect in the motion of the robot's end–effector, but in the *self–motion* of its joints (Nakamura, 1991). And as the second term of (4) always belongs to $\mathcal{N}(\mathcal{J})$, independently of the vector $\dot{\boldsymbol{q}}_0$ (that can be easily shown by left multiplying (4) by $J(\boldsymbol{q})$), then $\dot{\boldsymbol{q}}_0$ can be considered as a joint velocity vector chosen in a way that allows performing a secondary task, without affecting the motion of the robot's end–effector.
There exist different approaches for choosing vector $\dot{\boldsymbol{q}}_0$, most of them considering that it is the gradient of a suitable cost function. See (Sciavicco & Siciliano, 2000) for more information on this.

In the analysis forthcoming, it is assumed that the robot's end–effector motion makes $J(\boldsymbol{q})$ to be full rank during the whole task. Also, it is assumed that the following inequalities stand, for some positive constants k_J, k_{Jp}, and k_{dJp}:

$$\|J(\boldsymbol{q})\| \leq k_J \quad \forall\, \boldsymbol{q} \in \mathbb{R}^n, \tag{5}$$

$$\|J(\boldsymbol{q})^\dagger\| \leq k_{Jp}, \quad \forall\, \boldsymbol{q} \in \mathbb{R}^n, \tag{6}$$

$$\|\dot{J}(\boldsymbol{q})^\dagger\| = \left\| \frac{d}{dt}\left\{ J(\boldsymbol{q})^\dagger \right\} \right\| \leq k_{dJp}, \quad \forall\, \boldsymbol{q} \in \mathbb{R}^n. \tag{7}$$

2.1.2 Dynamics

The dynamic model of a serial n–joint robot free of friction is written (Kelly et al., 2005):

$$M(\boldsymbol{q})\ddot{\boldsymbol{q}} + C(\boldsymbol{q}, \dot{\boldsymbol{q}})\dot{\boldsymbol{q}} + \boldsymbol{g}(\boldsymbol{q}) = \tau, \tag{8}$$

where $\boldsymbol{q} \in \mathbb{R}^n$ is the vector of joint coordinates, $\dot{\boldsymbol{q}} \in \mathbb{R}^n$ is the vector of joint velocities, $\ddot{\boldsymbol{q}} \in \mathbb{R}^n$ is the vector of joint accelerations, $\boldsymbol{\tau} \in \mathbb{R}^n$ is the vector of applied torque inputs, $M(\boldsymbol{q}) \in \mathbb{R}^{n\times n}$ is the symmetric positive definite manipulator inertia matrix, $C(\boldsymbol{q}, \dot{\boldsymbol{q}}) \in \mathbb{R}^{n\times n}$ is the matrix of centripetal and Coriolis torques, and $\boldsymbol{g}(\boldsymbol{q}) \in \mathbb{R}^n$ is the vector of gravitational torques.
Some useful properties of the dynamic model (8) that will be used in this document, are the following (Kelly et al., 2005):
Property 1: $C(\boldsymbol{q}, \dot{\boldsymbol{q}})$ can be chosen so that matrix $\frac{1}{2}\dot{M}(\boldsymbol{q}) - C(\boldsymbol{q}, \dot{\boldsymbol{q}})$ is skew–symmetric, i.e.,

$$\boldsymbol{y}^T \left[\frac{1}{2}\dot{M}(\boldsymbol{q}) - C(\boldsymbol{q}, \dot{\boldsymbol{q}}) \right] \boldsymbol{y} = 0, \quad \forall\, \boldsymbol{y} \in \mathbb{R}^n. \tag{9}$$

Property 2: There exist positive constants k_M, k_C and k_g such that, for all $\boldsymbol{u}, \boldsymbol{w}, \boldsymbol{y} \in \mathbb{R}^n$, we have

$$\|M(\boldsymbol{u})\boldsymbol{y}\| \leq k_M\|\boldsymbol{y}\|, \tag{10}$$

$$\|C(\boldsymbol{u}, \boldsymbol{w})\boldsymbol{y}\| \leq k_C\|\boldsymbol{w}\|\|\boldsymbol{y}\|, \tag{11}$$

$$\|\boldsymbol{g}(\boldsymbol{u})\| \leq k_g. \tag{12}$$

2.2 Hierarchical control

It is well–known that most of industrial robots have internal joint velocity PI controllers, which are usually tuned with very high proportional and integral gains. In practice, these internal controllers are fixed (they belong to each of the joint actuator's servodrives), and an outer loop is necessary to achieve the control goal in operational space. This two–loop hierarchical structure is shown in Figure 1.

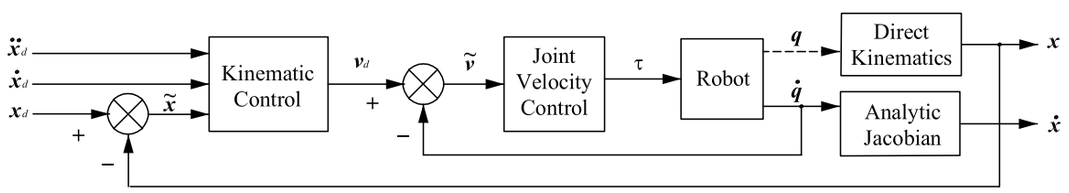

Fig. 1. Typical scheme of a hierarchical structure for operational space control.

2.2.1 Kinematic controller

By kinematic control we refer to any scheme that uses an inverse Jacobian algorithm to resolve the desired joint velocities directly from the pose variables of the desired task. Thus, a kinematic controller is used as the outer loop in the hierarchical controller of Figure 1, which provides the desired joint velocities for the inner velocity controller.
In this chapter we use as kinematic controller the so–called resolved motion rate control (RMRC), which was first proposed by (Whitney, 1969). Using this scheme, the desired joint velocity for the inner loop, can be written as

$$\boldsymbol{v}_d = J(\boldsymbol{q})^{\dagger}\left[\dot{\boldsymbol{x}}_d + K\tilde{\boldsymbol{x}}\right] + \left[I - J(\boldsymbol{q})^{\dagger}J(\boldsymbol{q})\right]\dot{\boldsymbol{q}}_0, \tag{13}$$

where $\boldsymbol{v}_d \in \mathbb{R}^n$ is the desired joint velocity vector, $\dot{\boldsymbol{x}}_d \in \mathbb{R}^6$ is the time derivative of the desired pose vector $\boldsymbol{x}_d$ in operational space, $\tilde{\boldsymbol{x}} = \boldsymbol{x}_d - \boldsymbol{x} \in \mathbb{R}^6$ is the pose error vector in operational space, and $K \in \mathbb{R}^{6\times 6}$ is a symmetric positive definite matrix of control gains. The second term in (13) is added to include the possibility of controlling a secondary task in redundant robots.

2.2.2 Joint velocity PI controller

For the inner loop in Figure 1, we consider the classical joint velocity proportional integral PI controller, commonly used in industrial robots, which can be written as

$$\boldsymbol{\tau} = K_p\tilde{\boldsymbol{v}} + K_i\boldsymbol{\xi}, \tag{14}$$

$$\dot{\boldsymbol{\xi}} = \tilde{\boldsymbol{v}}, \tag{15}$$

where

$$\tilde{\boldsymbol{v}} = \boldsymbol{v}_d - \dot{\boldsymbol{q}} \in \mathbb{R}^n, \tag{16}$$

$\boldsymbol{\xi} = \int_0^t \tilde{\boldsymbol{v}}(\sigma)\, d\sigma$, and $K_p, K_i \in \mathbb{R}^{n\times n}$ are diagonal positive definite matrices.
Without loss of generality, let us take

$$K_p = \left[k_p + \gamma\right] I, \tag{17}$$

$$K_i = \alpha K_p, \tag{18}$$

where k_p, γ and α are strictly positive constants. Notice that in such case (18) can be written as

$$K_i = \left[k_i + \alpha\gamma\right] I, \tag{19}$$

with $k_i = \alpha k_p$, or

$$\alpha = \frac{k_i}{k_p}. \tag{20}$$

3. Stability Analysis

The stability analysis presented here is taken from (Camarillo et al., 2008). More detail can be found in that reference.
First, some remarks on notation: We use $\lambda_{\min}\{A\}$ and $\lambda_{\max}\{A\}$ to represent, respectively, the smallest and the largest eigenvalues of a symmetric positive definite matrix $A(\boldsymbol{y})$, for any $\boldsymbol{y} \in \mathbb{R}^n$. Given $\boldsymbol{y} \in \mathbb{R}^n$ and a matrix $A(\boldsymbol{y})$ the Euclidean norm of $\boldsymbol{y}$ is defined as $\|\boldsymbol{y}\| = \sqrt{\boldsymbol{y}^T\boldsymbol{y}}$, and the induced norm of $A(\boldsymbol{y})$ is defined as $\|A(\boldsymbol{y})\| = \sqrt{\lambda_{\max}\{A^TA\}}$.

Note in Figure 1 that the closed–loop system is formed by the RMRC controller (13), the joint velocity PI controller (14)–(15), and the robot model, given by (8), (1) and (2).
Left multiplying (13) by $J(\boldsymbol{q})$, using (2), and (16), we get

$$\dot{\tilde{\boldsymbol{x}}} = -K\tilde{\boldsymbol{x}} + J(\boldsymbol{q})\tilde{\boldsymbol{v}}. \tag{21}$$

Substituting (14) into the robot dynamics (8) we have

$$M(\boldsymbol{q})\ddot{\boldsymbol{q}} + C(\boldsymbol{q},\dot{\boldsymbol{q}})\dot{\boldsymbol{q}} + \boldsymbol{g}(\boldsymbol{q}) = K_p\tilde{\boldsymbol{v}} + K_i\boldsymbol{\xi}. \tag{22}$$

Now, taking into account (16) and its derivative, we get

$$M(\boldsymbol{q})\dot{\tilde{\boldsymbol{v}}} + C(\boldsymbol{q},\dot{\boldsymbol{q}})\tilde{\boldsymbol{v}} + K_p\tilde{\boldsymbol{v}} + K_i\boldsymbol{\xi} = M(\boldsymbol{q})\dot{\boldsymbol{v}}_d + C(\boldsymbol{q},\dot{\boldsymbol{q}})\boldsymbol{v}_d + \boldsymbol{g}(\boldsymbol{q}).$$

We can add the terms $\alpha M(\boldsymbol{q})\tilde{\boldsymbol{v}} + \alpha C(\boldsymbol{q},\dot{\boldsymbol{q}})\boldsymbol{\xi}$ to both sides of the last equation, to obtain

$$\begin{aligned} M(\boldsymbol{q})\left[\dot{\tilde{\boldsymbol{v}}} + \alpha\tilde{\boldsymbol{v}}\right] + C(\boldsymbol{q},\dot{\boldsymbol{q}})\left[\tilde{\boldsymbol{v}} + \alpha\boldsymbol{\xi}\right] + K_p\tilde{\boldsymbol{v}} + K_i\boldsymbol{\xi} = \\ M(\boldsymbol{q})\left[\dot{\boldsymbol{v}}_d + \alpha\tilde{\boldsymbol{v}}\right] + C(\boldsymbol{q},\dot{\boldsymbol{q}})\left[\boldsymbol{v}_d + \alpha\boldsymbol{\xi}\right] + \boldsymbol{g}(\boldsymbol{q}). \end{aligned} \tag{23}$$

Finally, from (21), (15) and (23), we get the closed–loop system in terms of the state vector $\boldsymbol{z} = \begin{bmatrix}\tilde{\boldsymbol{x}}^T & \boldsymbol{\xi}^T & \tilde{\boldsymbol{v}}^T\end{bmatrix}^T \in \mathbb{R}^{6+2n}$:

$$\frac{d}{dt}\begin{bmatrix}\tilde{\boldsymbol{x}}\\ \boldsymbol{\xi}\\ \tilde{\boldsymbol{v}}\end{bmatrix} = \begin{bmatrix} J(\boldsymbol{q})\tilde{\boldsymbol{v}} - K\tilde{\boldsymbol{x}} \\ \tilde{\boldsymbol{v}} \\ -\alpha\tilde{\boldsymbol{v}} - M(\boldsymbol{q})^{-1}\left[C(\boldsymbol{q},\dot{\boldsymbol{q}})\left[\tilde{\boldsymbol{v}} + \alpha\boldsymbol{\xi}\right] + K_p\tilde{\boldsymbol{v}} + K_i\boldsymbol{\xi} - \boldsymbol{d}(t,\boldsymbol{z})\right]\end{bmatrix}, \tag{24}$$

where the term

$$\boldsymbol{d}(t,\boldsymbol{z}) = M(\boldsymbol{q})\left[\dot{\boldsymbol{v}}_d + \alpha\tilde{\boldsymbol{v}}\right] + C(\boldsymbol{q},\dot{\boldsymbol{q}})\left[\boldsymbol{v}_d + \alpha\boldsymbol{\xi}\right] + \boldsymbol{g}(\boldsymbol{q}) \tag{25}$$

is considered as a disturbance and, as it is shown in (Camarillo et al., 2008), is upper bounded for all $t \geq t_0$ by a quadratic polynomial:

$$\|\boldsymbol{d}(t,\boldsymbol{z})\| \leq \varsigma_2\|\boldsymbol{z}(t)\|^2 + \varsigma_1\|\boldsymbol{z}(t)\| + \varsigma_0, \tag{26}$$

where ς_0, ς_1, ς_2 are positive constants, assuming that $\|\boldsymbol{v}_d\|$ and $\|\dot{\boldsymbol{v}}_d\|$ are also bounded.
Substituting the definitions (17) and (19) in (24), we get

$$\frac{d}{dt}\begin{bmatrix}\tilde{\boldsymbol{x}}\\ \boldsymbol{\xi}\\ \tilde{\boldsymbol{v}}\end{bmatrix} = \begin{bmatrix} J(\boldsymbol{q})\tilde{\boldsymbol{v}} - K\tilde{\boldsymbol{x}} \\ \tilde{\boldsymbol{v}} \\ -\alpha\tilde{\boldsymbol{v}} - M(\boldsymbol{q})^{-1}\left[C(\boldsymbol{q},\dot{\boldsymbol{q}})\left[\tilde{\boldsymbol{v}} + \alpha\boldsymbol{\xi}\right] + k_p\left[\tilde{\boldsymbol{v}} + \alpha\boldsymbol{\xi}\right] + \gamma\left[\tilde{\boldsymbol{v}} + \alpha\boldsymbol{\xi}\right] - \boldsymbol{d}(t,\boldsymbol{z})\right]\end{bmatrix}. \tag{27}$$

In accordance with the theory of perturbed systems (Khalil, 2005), we define

$$\frac{d}{dt}\begin{bmatrix}\tilde{\boldsymbol{x}}\\ \boldsymbol{\xi}\\ \tilde{\boldsymbol{v}}\end{bmatrix} = \begin{bmatrix} J(\boldsymbol{q})\tilde{\boldsymbol{v}} - K\tilde{\boldsymbol{x}} \\ \tilde{\boldsymbol{v}} \\ -\alpha\tilde{\boldsymbol{v}} - M(\boldsymbol{q})^{-1}\left[C(\boldsymbol{q},\dot{\boldsymbol{q}})\left[\tilde{\boldsymbol{v}} + \alpha\boldsymbol{\xi}\right] + k_p\left[\tilde{\boldsymbol{v}} + \alpha\boldsymbol{\xi}\right]\right]\end{bmatrix} \tag{28}$$

as the nominal system from the closed–loop system (27). As we can see, (28) is free of disturbances, and the origin of the state space $\boldsymbol{z} = \boldsymbol{0}$ is an equilibrium of (28).
In order to show the stability of the overall closed–loop system (27), we use the methodology proposed in (Khalil, 2005), that is:

(a) First, to prove the asymptotic stability of the nominal system (28).

(b) Then, to prove the uniform ultimate boundedness of the overall closed–loop system (27).

3.1 Stability of the nominal system

To prove that the origin of the nominal system (28) is exponentially stable, we propose the following Lyapunov function candidate, inspired from (Qu & Dorsey, 1991):

$$V(t,\boldsymbol{z}) = \frac{1}{2}\tilde{\boldsymbol{x}}^T\tilde{\boldsymbol{x}} + \alpha\boldsymbol{\xi}^T\left[k_p I + \frac{1}{2}\alpha M(\boldsymbol{q})\right]\boldsymbol{\xi} + \alpha\boldsymbol{\xi}^T M(\boldsymbol{q})\tilde{\boldsymbol{v}} + \frac{1}{2}\tilde{\boldsymbol{v}}^T M(\boldsymbol{q})\tilde{\boldsymbol{v}}. \tag{29}$$

It is easy to show that

$$\begin{bmatrix}\|\tilde{\boldsymbol{x}}\|\\ \|\boldsymbol{\xi}\|\\ \|\tilde{\boldsymbol{v}}\|\end{bmatrix}^T P_1 \begin{bmatrix}\|\tilde{\boldsymbol{x}}\|\\ \|\boldsymbol{\xi}\|\\ \|\tilde{\boldsymbol{v}}\|\end{bmatrix} \leq V(t,\boldsymbol{z}) \leq \begin{bmatrix}\|\tilde{\boldsymbol{x}}\|\\ \|\boldsymbol{\xi}\|\\ \|\tilde{\boldsymbol{v}}\|\end{bmatrix}^T P_2 \begin{bmatrix}\|\tilde{\boldsymbol{x}}\|\\ \|\boldsymbol{\xi}\|\\ \|\tilde{\boldsymbol{v}}\|\end{bmatrix},$$

with

$$\begin{aligned} P_1 &= \begin{bmatrix}\frac{1}{2} & 0 & 0\\ 0 & \alpha k_p + \frac{1}{2}\alpha^2\lambda_{\min}\{M\} & -\frac{1}{2}\alpha\lambda_{\max}\{M\}\\ 0 & -\frac{1}{2}\alpha\lambda_{\max}\{M\} & \frac{1}{2}\lambda_{\min}\{M\}\end{bmatrix},\\ P_2 &= \begin{bmatrix}\frac{1}{2} & 0 & 0\\ 0 & \alpha k_p + \frac{1}{2}\alpha^2\lambda_{\max}\{M\} & \frac{1}{2}\alpha\lambda_{\max}\{M\}\\ 0 & \frac{1}{2}\alpha\lambda_{\max}\{M\} & \frac{1}{2}\lambda_{\max}\{M\}\end{bmatrix}, \end{aligned}$$

so that the Lyapunov function candidate (29) satisfies

$$\lambda_1\|\boldsymbol{z}\|^2 \leq V(t,\boldsymbol{z}) \leq \lambda_2\|\boldsymbol{z}\|^2, \tag{30}$$

for some positive constants λ_1, λ_2, given by

$$\lambda_1 = \lambda_{\min}\{P_1\},\ \lambda_2 = \lambda_{\max}\{P_2\}. \tag{31}$$

Notice that $V(t,\boldsymbol{z})$ is a globally positive definite function if

$$\alpha \leq \frac{2k_p\lambda_{\min}\{M\}}{\lambda^2_{\max}\{M\} - \lambda^2_{\min}\{M\}}, \tag{32}$$

or, using (20), if

$$k_i \leq \frac{2k_p^2\lambda_{\min}\{M\}}{\lambda^2_{\max}\{M\} - \lambda^2_{\min}\{M\}}. \tag{33}$$

On the other hand, the time derivative of the Lyapunov function candidate (29) is given by

$$\begin{aligned} \dot{V}(t,\boldsymbol{z}) &= \tilde{\boldsymbol{x}}^T\dot{\tilde{\boldsymbol{x}}} + \alpha k_p\boldsymbol{\xi}^T\dot{\boldsymbol{\xi}} + \alpha^2\boldsymbol{\xi}^T M(\boldsymbol{q})\dot{\boldsymbol{\xi}} + \frac{1}{2}\alpha^2\boldsymbol{\xi}^T\dot{M}(\boldsymbol{q})\boldsymbol{\xi} + \alpha\dot{\boldsymbol{\xi}}^T M(\boldsymbol{q})\tilde{\boldsymbol{v}} + \alpha\boldsymbol{\xi}^T\dot{M}(\boldsymbol{q})\tilde{\boldsymbol{v}}\\ &\quad + \alpha\boldsymbol{\xi}^T M(\boldsymbol{q})\dot{\tilde{\boldsymbol{v}}} + \tilde{\boldsymbol{v}}^T M(\boldsymbol{q})\dot{\tilde{\boldsymbol{v}}} + \frac{1}{2}\tilde{\boldsymbol{v}}^T\dot{M}(\boldsymbol{q})\tilde{\boldsymbol{v}}, \end{aligned} \tag{34}$$

which along the solutions of the nominal system (28) results in

$$\dot{V}(t,\boldsymbol{z}) = -\tilde{\boldsymbol{x}}^T K\tilde{\boldsymbol{x}} - \alpha k_i\boldsymbol{\xi}^T\boldsymbol{\xi} - k_p\tilde{\boldsymbol{v}}^T\tilde{\boldsymbol{v}} + \tilde{\boldsymbol{x}}^T J(\boldsymbol{q})\tilde{\boldsymbol{v}},$$

where we have used the definition of α in (20), and the skew–symmetry property (9) to eliminate some terms.

Then, under the assumption that the Jacobian matrix $J(\boldsymbol{q})$ is bounded, see (5), it is easy to show that

$$\dot{V}(t,\boldsymbol{z}) \leq -\begin{bmatrix} \|\tilde{\boldsymbol{x}}\| \\ \|\boldsymbol{\xi}\| \\ \|\tilde{\boldsymbol{v}}\| \end{bmatrix}^T Q \begin{bmatrix} \|\tilde{\boldsymbol{x}}\| \\ \|\boldsymbol{\xi}\| \\ \|\tilde{\boldsymbol{v}}\| \end{bmatrix}, \tag{35}$$

with

$$Q = \begin{bmatrix} \lambda_{\min}\{K\} & 0 & -\frac{1}{2}k_J \\ 0 & \alpha k_i & 0 \\ -\frac{1}{2}k_J & 0 & k_p \end{bmatrix}. \tag{36}$$

So that (35) can be written as

$$\dot{V}(t,\boldsymbol{z}) \leq -\lambda_3 \|\boldsymbol{z}\|^2, \tag{37}$$

where

$$\lambda_3 = \lambda_{\min}\{Q\}. \tag{38}$$

Notice that we can choose

$$\lambda_{\min}\{K\} > \frac{k_J^2}{4k_p}, \tag{39}$$

so that Q is positive definite and $\dot{V}(t,\boldsymbol{z})$ is globally negative definite.
Thus, provided that (33) and (39) are satisfied, and according to Lyapunov's direct method (Khalil, 2005), we conclude the exponential stability of the origin of the nominal system (28). This implies that the origin of the state space is attractive, i.e.

$$\lim_{t\to\infty} \boldsymbol{z}(t) = \lim_{t\to\infty} \begin{bmatrix} \tilde{\boldsymbol{x}}(t) \\ \boldsymbol{\xi}(t) \\ \tilde{\boldsymbol{v}}(t) \end{bmatrix} = \boldsymbol{0}.$$

3.2 Stability of the overall closed–loop system

We have shown in the previous subsection that the equilibrium of the nominal system (28) is exponentially stable by choosing the feedback gains k_i, and $\lambda_{\min}\{K\}$ according to (33) and (39), respectively. Now, we are able to show that, by properly choosing the feedback gain γ of the joint velocity PI controller, the solutions $\boldsymbol{z}(t)$ of the overall closed–loop system (27) are uniformly ultimately bounded.
First, we recall the following fact (Qu & Dorsey, 1991), that will be useful for our purposes.
Fact 1: *Let*

$$g(\|\boldsymbol{z}\|) = -\alpha_1\|\boldsymbol{z}\|^2 + \alpha_2\|\boldsymbol{z}\| + \alpha_3, \tag{40}$$

be a quadratic polynomial with $\alpha_1 > 0$, *and* $\alpha_2, \alpha_3 \geq 0$. *Given*

$$\eta = \frac{\alpha_2 + \sqrt{\alpha_2^2 + 4\alpha_1\alpha_3}}{2\alpha_1}, \tag{41}$$

then

- $g(\eta) = 0$,

- $g(\|\boldsymbol{z}\|) < 0$, *and strictly decreasing, for all* $\|\boldsymbol{z}\| > \eta$.

$\diamondsuit$

Now, we recall the following theorem in (Camarillo et al., 2008), which is a variation of Theorem 4.18 in (Khalil, 2005).

Theorem 1: *Let*

$$\dot{\boldsymbol{z}} = \boldsymbol{f}(t, \boldsymbol{z}), \tag{42}$$

be a system that describes a first–order vector differential equation, where $\boldsymbol{f} : [0, \infty) \times D \rightarrow \mathbb{R}^m$ *and* $D \subset \mathbb{R}^m$*, is a domain that contains the origin. Let* $V(t, \boldsymbol{z})$*, with* $V : [0, \infty) \times D \rightarrow \mathbb{R}$*, be a Lyapunov–like function that satisfies*

$$\lambda_1 \|\boldsymbol{z}\|^2 \leq V(t, \boldsymbol{z}) \leq \lambda_2 \|\boldsymbol{z}\|^2, \tag{43}$$

$$\dot{V}(t, \boldsymbol{z}) \leq g(\|\boldsymbol{z}\|) < 0, \quad \forall \|\boldsymbol{z}\| > \eta, \tag{44}$$

for all $t \geq t_0$*, and for all* $\boldsymbol{z} \in D$*, with* $\lambda_1, \lambda_2 > 0$*,* $g(\|\boldsymbol{z}\|)$ *and* η *defined in Fact 1 by* (40) *and* (41)*, respectively. Then, there exists* $T \geq 0$ *(dependent on* $\boldsymbol{z}(t_0)$ *and* η*) such that the solution of* (42)*, satisfies the following properties, for any initial state* $\boldsymbol{z}_0 = \boldsymbol{z}(t_0)$*:*

A. *Uniform boundedness*

$$\|\boldsymbol{z}(t)\| \leq \sqrt{\frac{\lambda_2}{\lambda_1}} \max\{\|\boldsymbol{z}_0\|, \eta\}, \quad \forall\, t \geq t_0. \tag{45}$$

B. *Uniform ultimate boundedness*

$$\|\boldsymbol{z}(t)\| \leq \sqrt{\frac{\lambda_2}{\lambda_1}}\, \eta', \quad \forall\, t \geq t_0 + T, \tag{46}$$

for all $\eta' > \eta$.

$\triangledown$

The proof of Theorem 1 can be found in (Camarillo et al., 2008). Theorem 1 is the base for the following result:

Proposition 1: *Consider the overall closed–loop system* (27) *with the outer–loop kinematic controller*

$$\boldsymbol{v}_d = J(\boldsymbol{q})^{\dagger}[\dot{\boldsymbol{x}}_d + K\tilde{\boldsymbol{x}}],$$

and the inner–loop velocity controller

$$\boldsymbol{\tau} = K_p\tilde{\boldsymbol{v}} + K_i\boldsymbol{\xi} = [k_p + \gamma]\tilde{\boldsymbol{v}} + [k_i + \gamma\alpha]\boldsymbol{\xi}. \tag{47}$$

Suppose the control gains are chosen so that k_p*,* k_i *and* K *satisfy* (33) *and* (39)*. Moreover,* γ *is a positive scalar such that*

$$\gamma \geq \frac{1}{\epsilon}\left[\sigma^2\varsigma_2 + \sigma\varsigma_1 + \varsigma_0\right], \tag{48}$$

where ς_0*,* ς_1 *and* ς_2 *are the coefficients of the upper bound of* $\|\boldsymbol{d}(t, \boldsymbol{z})\|$*, given in* (26)*,* ϵ *is a positive constant such that*

$$0 < \epsilon < \frac{\lambda_3}{\varsigma_2}, \tag{49}$$

and

$$\sigma = \sqrt{\frac{\lambda_2}{\lambda_1}} \max\{\|\boldsymbol{z}(t_0)\|, \bar{\eta}\}, \tag{50}$$

with

$$\bar{\eta} = \frac{\epsilon\varsigma_1 + \sqrt{\epsilon^2\varsigma_1^2 + 4\epsilon\varsigma_0\left[\lambda_3 - \epsilon\varsigma_2\right]}}{2\left[\lambda_3 - \epsilon\varsigma_2\right]}, \tag{51}$$

where λ_1, λ_2, *are defined in* (31), *and* λ_3 *is in* (38).
Then, the system is stabilizable in the sense that exists $T \geq 0$ *such that the solution* $\boldsymbol{z}(t)$, *with initial state* $\boldsymbol{z}_0 = \boldsymbol{z}(t_0)$ *satisfies*

$$\|\boldsymbol{z}(t)\| \leq \sigma, \quad \forall\, t \geq t_0 \ \text{(uniform boundedness)}, \tag{52}$$

$$\|\boldsymbol{z}(t)\| \leq \sqrt{\frac{\lambda_2}{\lambda_1}}\bar{\eta}', \quad \forall\, t \geq t_0 + T \ \text{(uniform ultimate boundedness)}. \tag{53}$$

for all $\bar{\eta}' > \bar{\eta}$, *where* (52) *and* (53) *regard to the uniform and uniform ultimate boundedness, respectively.*

▽

The proof of Proposition 1 is also in (Camarillo et al., 2008); it employs (25) and Theorem 1. The following remarks can be set from this result:

Remark 1. *Notice that if* $k_p \to \infty$ *in the PI controller then, from* (20), $\alpha \to 0$ *and, considering* (36), *and* (49), *we get* $\lambda_3 = \lambda_{\min}\{Q\} \to 0$ *and* $\epsilon \to 0$. *Moreover, from* (51), $\epsilon \to 0$ *implies that* $\bar{\eta} \to 0$, *the uniform bound* σ *becomes* $\sigma = \sqrt{\frac{\lambda_2}{\lambda_1}}\|\boldsymbol{z}(t_0)\|$, *and the system tends to be exponentially stable.*

Remark 2. *It is noteworthy that the joint velocity PI controller*

$$\begin{aligned} \boldsymbol{\tau} &= [k_p + \gamma]\tilde{\boldsymbol{v}} + [k_i + \gamma\alpha]\boldsymbol{\xi} \\ \dot{\boldsymbol{\xi}} &= \tilde{\boldsymbol{v}} \end{aligned}$$

can be also written as

$$\begin{aligned} \boldsymbol{\tau} &= [k_p + \gamma]\tilde{\boldsymbol{v}} + \alpha\boldsymbol{y} \\ \frac{1}{k_p + \gamma}\dot{\boldsymbol{y}} &= \tilde{\boldsymbol{v}}. \end{aligned}$$

Thus, if $\gamma \to \infty$, *then,* $\tilde{\boldsymbol{v}} \to 0$, *and* $\dot{\boldsymbol{q}} \to \boldsymbol{v}_d$. *Also, considering* (17) *and* (19), $\gamma \to \infty$ *implies* $K_p, K_i \to \infty$.

From the previous remarks we can conclude that the common assumption of having high–value gains in the inner velocity PI controller of industrial robots leads to small tracking errors during the execution of a motion control task.

4. The PA10 Robot System

The Mitsubishi PA10 arm is an industrial robot manipulator which completely changes the vision of conventional industrial robots. Its name is an acronym of *Portable General-Purpose Intelligent Arm*. There exist two versions (Higuchi et al., 2003): the PA10-6C and the PA10-7C, where the suffix digit indicates the number of degrees of freedom of the arm. This work focuses on the study of the PA10-7CE model, which is the enhanced version of the PA10-7C. The PA10 arm is an open architecture robot; it means that it possesses (Oonishi, 1999):

- A hierarchical structure with several control levels.
- Communication between levels, via standard interfaces.
- An open general–purpose interface in the higher level.

This scheme allows the user to focus on the programming of the tasks at the PA10 system's higher level, without regarding on the operation of the lower levels. The programming can be performed using a high–level language, such as Visual BASIC or Visual C++, from a PC with Windows operating system. The PA10 robot is currently the open–architecture robot more employed for research (see, e.g., Kennedy & Desai (2003), Jamisola et al. (2004), Pholsiri (2004)).

4.1 Basic structure of the PA10 robot

The PA10 system is composed of four sections or levels, which conform a hierarchical structure (Mitsubishi, 2002a):

Level 4: Operation control section (OCS); formed by the PC and the teaching pendant.

Level 3: Motion control section (MCS); formed by the motion control and optical boards.

Level 2: Servo drives.

Level 1: Robot manipulator.

Figure 2 shows the relation existing among each of the mentioned components. The following subsections give a more detailed description of them.

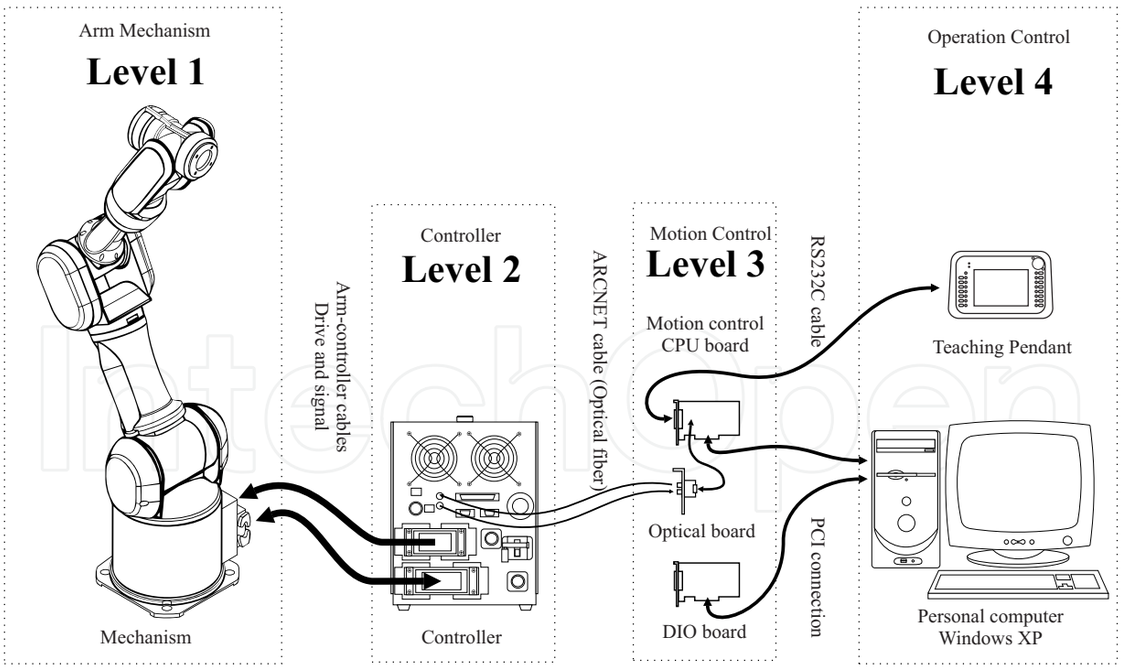

Fig. 2. Components of the PA10 system

4.1.1 Operation control section

This section is the higher level in the PA10 system's hierarchy; it receives its name because it is in this section in which the operator programs the tasks to be performed by the robot. This can be done by a control computer or a teaching pendant. The control computer is a PC with an MS-Windows operating system. For the programming of the robot tasks, the operator can employ the programs provided by the manufacturer, or he can develop his own applications using Visual C++ or Visual BASIC and some API functions from the so–called *PA library*.
It is through this functions that the user can select the control scheme he wants for the robot, being the most common the joint velocity control, the joint position control and the pose control. More information on how these schemes can be implemented, as well as an experimental comparison among them, can be found in (Camarillo et al., 2006). It is worth mentioning, however, that all of these schemes work with the servo–drives configured in velocity mode.
Independently on the way the tasks to be performed are specified in the operation control section, the information which is passed to the next level is always the same in this basic configuration. This allows to keep a hierarchical structure, but at the same time, it sets limitations to the versatility of the whole system.
For the experimental evaluation shown in Section 6, we decided to use the control computer and a real–time application, developed in Visual C++. See section 5.3 for a description of this.

4.1.2 Motion control section

It is in this section where the joint velocity commands for the servo–drives (Level 2) are computed, using the information programmed by the user in Level 4. To do so, the original PA10 system uses the MHI-D7281 motion control board, from Mitsubishi Heavy Industries, which is plugged in the PCI bus of the control computer. It is in this board where, among other things, the joint position and pose kinematic controllers, together with the interpolation algorithms for generating smooth trajectories, are implemented.
Communication among the control computer and the driver cabinet is done via optical fibers, using the ARCNET protocol. The motion control board counts on a converter to encode signals to the ARCNET format. More information on this can be found in Section 5.1.
A board model MHI-D7210 is dedicated to convert the electrical signals from the motion control board into optical signals. This board is called the optical board in Figure 2. For more information on the motion control board, and the optical board, see (Mitsubishi, 2002a).

4.1.3 Servo–drives

The seven drives of the PA10 servomotors are contained in a cabinet (driver) which connects to the optical–interface board via optical fiber. In addition, two cables connect the cabinet to the robot; one contains the power signals for the servomotors, while the other transmits the feedback signals from the joint position sensors.
PA10's servomotors are brushless DC type, and they are coupled to the links via harmonic drives. The servo–drives can be configured in torque or velocity modes. However, if the motion control board in the MCS is used, then it is only possible to configure the drives in velocity mode. In order to use the PA10 robot in torque mode, it is necessary to disable the MHI-D7281 board, and replace it by one which allows the operator to have direct access to the drive signals from the PC. This is explained in Section 5.2.
In velocity mode, the drives include an inner velocity loop which, according to the documentation provided by Mitsubishi (Mitsubishi, 2002b) is a digital PI controller with a 400 μs sampling period. Drives also include a current (torque) PI controller, with a 100 μs sampling

period, which, as indicated in (Campa et al., 2008) has an insignificant effect in the servomotor's dynamics at low velocities.

4.1.4 Robot manipulator

The PA10 robot is a 7–dof redundant manipulator with revolute joints. Figure 3 shows a diagram of the PA10 arm, indicating the positive rotation direction and the respective names of each of the joints.

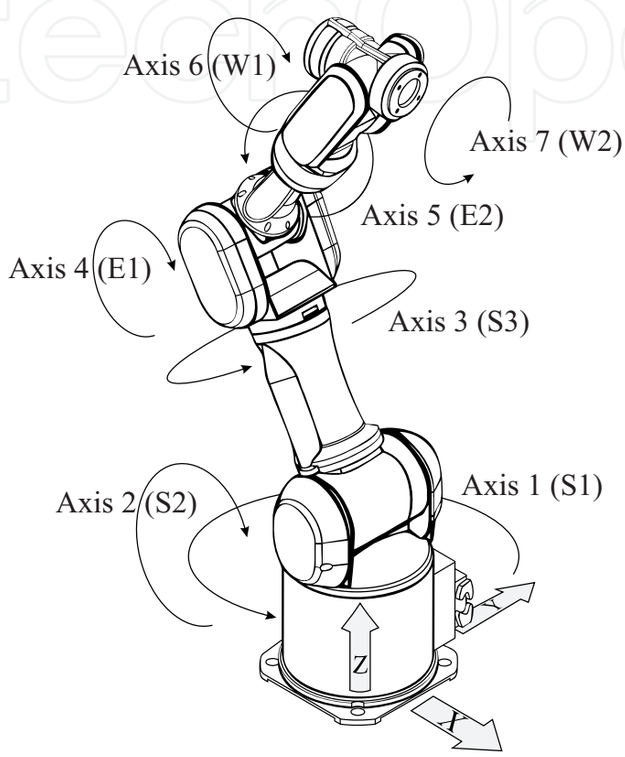

Fig. 3. Mitsubishi PA10-7CE robot

4.2 Modeling of the PA10 arm

As the PA10-7CE is a robot with seven degrees of freedom, then $n = 7$, and the direct kinematics function $\boldsymbol{f} : \mathbb{R}^7 \rightarrow \mathbb{R}^6$ has the form $\boldsymbol{x} = \boldsymbol{f}(\boldsymbol{q})$.
In this work, we consider the vector of operational coordinates $\boldsymbol{x}$ is formed by three Cartesian coordinates (x,y,z) and three Euler angles (α,β,γ), given by the ZYX convention (Craig, 2004). Using the traditional methodology, based on the Denavit-Hartenberg convention (Denavit & Hartenberg, 1955), we obtain for the PA10 arm the following expressions:

$$
\begin{aligned}
x &= [-([(c_1c_2c_3 - s_1s_3)c_4 - c_1s_2s_4]c_5 - [-(-c_1c_2s_3 - s_1c_3)]s_5)s_6 \\
&\quad +(-[c_1c_2c_3 - s_1s_3]s_4 - c_1s_2c_4)c_6]\, d_7 \\
&\quad +[-(c_1c_2c_3 - s_1s_3)s_4 - c_1s_2c_4]\, d_5 - c_1s_2d_3, \\
y &= [-([(s_1c_2c_3 + c_1s_3)c_4 - s_1s_2s_4]c_5 - (-(-s_1c_2s_3 + c_1c_3))s_5)s_6 \\
&\quad +(-[s_1c_2c_3 + c_1s_3]s_4 - s_1s_2c_4)c_6]\, d_7 \\
&\quad +[-(s_1c_2c_3 + c_1s_3)s_4 - s_1s_2c_4]\, d_5 - s_1s_2d_3, \\
z &= [-([s_2c_3c_4 + c_2s_4]c_5 - s_2s_3s_5)s_6 + (-s_2c_3s_4 + c_2c_4)c_6]\, d_7 \\
&\quad +[-s_2c_3s_4 + c_2c_4]\, d_5 + c_2d_3, \\
\alpha &= \text{atan2}\,(b, a)\,, \\
\beta &= \text{atan2}\left(-c, \frac{a}{\cos(\alpha)}\right), \\
\gamma &= \text{atan2}\,(d, e)\,.
\end{aligned}
$$

where the terms c_i y s_i correspond to $\cos(q_i)$ y $\text{sen}(q_i)$, respectively, and d_i is the length of the i-th link, with $i = 1, 2, \ldots, 7$. The required values for this robot are $d_1 = 0.317$ m, $d_3 = 0.450$ m, $d_5 = 0.480$ m, and $d_7 = 0.070$ m. For computing the Euler angles it is employed the inverse tangent function with two arguments (atan2) and the following expressions:

$$
\begin{aligned}
a &= [([(c_1c_2c_3 - s_1s_3)c_4 - c_1s_2s_4]c_5 + [-c_1c_2s_3 - s_1c_3]s_5)c_6 \\
&\quad -(-[-(c_1c_2c_3 - s_1s_3)s_4 - c_1s_2c_4])s_6]\, c_7 \\
&\quad +[-([c_1c_2c_3 - s_1s_3]c_4 - c_1s_2s_4)s_5 + (-c_1c_2s_3 - s_1c_3)c_5]\, s_7 \\
b &= [([(s_1c_2c_3 + c_1s_3)c_4 - s_1s_2s_4]c_5 + [-s_1c_2s_3 + c_1c_3]s_5)c_6 \\
&\quad -(-(-(s_1c_2c_3 + c_1s_3)s_4 - s_1s_2c_4))s_6]\, c_7 \\
&\quad +[-([s_1c_2c_3 + c_1s_3]c_4 - s_1s_2s_4)s_5 + (-s_1c_2s_3 + c_1c_3)c_5]\, s_7 \\
c &= [([-s_2c_3c_4 - c_2s_4]c_5 + s_2s_3s_5)c_6 - (-[s_2c_3s_4 - c_2c_4])s_6]\, c_7 \\
&\quad +[-(-s_2c_3c_4 - c_2s_4)s_5 + s_2s_3c_5]s_7 \\
d &= -[([-s_2c_3c_4 - c_2s_4]c_5 + s_2s_3s_5)c_6 - (-[s_2c_3s_4 - c_2c_4])s_6]\, s_7 \\
&\quad +[-(-s_2c_3c_4 - c_2s_4)s_5 + s_2s_3c_5]c_7 \\
e &= -[-([-s_2c_3c_4 - c_2s_4]c_5 + s_2s_3s_5)s_6 - (-[s_2c_3s_4 - c_2c_4])c_6]
\end{aligned}
$$

By reasons of space, we do not include here the analytic Jacobian of the PA10 robot; however, it is possible to obtain it from the direct kinematics, since

$$J(\boldsymbol{q}) = \frac{\partial \boldsymbol{f}(\boldsymbol{q})}{\partial \boldsymbol{q}}.$$

Other way of computing the analytic Jacobian is via the geometric Jacobian, $J_G(\boldsymbol{q})$, which relates the joint velocities with the linear and angular velocities, $\boldsymbol{v}$ and $\boldsymbol{\omega}$, respectively, according to (Sciavicco & Siciliano, 2000)

$$\begin{bmatrix} \boldsymbol{v} \\ \boldsymbol{\omega} \end{bmatrix} = J_G(\boldsymbol{q})\dot{\boldsymbol{q}}.$$

After computing the geometric Jacobian, using standard algorithms (Sciavicco & Siciliano, 2000), we can use the following expression:

$$J(\boldsymbol{q}) = \begin{bmatrix} I & 0 \\ 0 & T(\alpha, \beta, \gamma) \end{bmatrix} J_G(\boldsymbol{q})$$

where $T(\alpha, \beta, \gamma)$ is a representation transformation matrix which depends on the Euler angle convention employed. In the case of the ZYX convention, we have that

$$T(\alpha, \beta, \gamma) = \begin{bmatrix} 0 & -\sin(\alpha) & \cos(\alpha)\cos(\beta) \\ 0 & \cos(\alpha) & \sin(\alpha)\cos(\beta) \\ 1 & 0 & -\sin(\beta) \end{bmatrix}.$$

Regarding the dynamic model of the PA10-7CE robot, it was not required for the experiments described in Section 6, although it can be found in (Ramírez, 2008).

5. Real–time Control Platform

In this section we first describe the general characteristics of the ARCNET protocol, and give the required specifications for its use in the PA10 system. After that, we describe the modifications to the hardware, and the software application we have developed for controlling the PA10 robot in real–time.

5.1 The ARCNET protocol

ARCNET (acronym of Attached Resource Computer Network) is a protocol for data transmission in a local area network (LAN). It was developed by Datapoint Corporation and it became very popular in the 80's. In the current days, even though it has been relegated by Ethernet in commercial and academic networks, it is still used in the industrial field, since its deterministic behavior and robustness are appropriate for the control of devices.

ARCNET handles data packets of variable length, from 0 to 507 bytes, and operates at velocities from 2.5 Mbps to 10 Mbps, so that it is possible to have a fast response for short messages. Another advantage is that it allows different types of transmission media, being the twisted cable, the coaxial cable and the optical fiber the most common. Moreover, the data interchange is implemented by hardware, in an ARCNET controller IC, so that the basic functions such as error checking, flux control and network configuration are performed directly in the IC.

Each device connected to an ARCNET network is known as a node, and it must contain a controller IC as well as an adapter for the type of cable employed. A MAC address identifier is assigned to each node. An ARCNET network can have up to 255 nodes with a unique identifier (or node number) assigned to each of them.

The ARCNET protocol employs an scheme known as token–passing, to manage the data stream among the active nodes in the network. When a node possesses the token, it can transmit data through the net or pass the token to the neighboring node, which, even though it can be located physically in any place, it has a consecutive node number. The header of each sent packet includes the address of the receiver node, in a way that all the nodes, with the exception of the pretended receiver, overlook those data. Once a message has been sent, the token passes to the next node, in a consecutive way, up to returning to the original node.

If the message is very long (more than 507 bytes) it is split in several packets. The packets can be short or long depending of the selected mode. Short packets are from 0 to 252 bytes, and

the long ones from 256 to 507 bytes. It is not possible to send packets of 253, 254 or 255 bytes; in those cases null data are added and they are sent either as two short packets or one long.
As each node can only send data when it has the token, no collisions occur when using the ARCNET protocol. Moreover, thanks to the token–passing scheme, the time a node lasts in sending a message can be computed; this is an important feature in industrial networks, where it is required that the control events occur in a precise moment.

5.1.1 Specifications for the PA10 system

Communication between the motion control board and the PA10's motor drives is carried out via an optical fiber, using the ARCNET protocol. In its original configuration a transmission rate of 10 Mbps is employed; the node number of the ARCNET controller IC in the driver cabinet is 254 (hexadecimal FE), while the one in the motion control board has number 255 (hexadecimal FF). As explained in (Mitsubishi, 2002b), the data packets are short, and they follow the format which is shown in Figure 4.

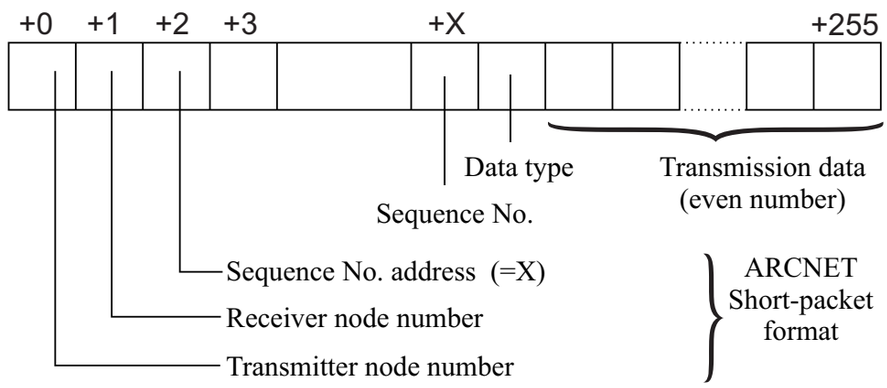

Fig. 4. ARCNET data transmission format

The first two bytes in each packet indicate the node numbers of the transmitter and receiver, respectively; next byte specifies the address (X) in which the data block start, in a way that the last byte always be the number 256.
The first byte of data (identified as "Data type" in Figure 4) can be either 'S', 'E' or 'C'. Commands 'S' and 'E' indicate, respectively, the start and end of a control sequence; when using these commands, no more data is expected. Command 'C' is used to transmit control data, which must be done periodically. When executing the 'S' command, a timer starts its operation, which checks that the time between each 'C' command does not exceeds a preestablished time limit; in case of occurring this, an alarm signal is activated, and the operation of the robot is blocked. The sequence of commands for normal operation is 'S'→'C'→ $\cdots$ →'C'→'E'.
The control computer (either using the motion control board or not) must send periodically the necessary signals to the servo-drives; these must acknowledge, returning to the computer the same command that they receive.
When executing the 'C' command, the control computer sends 44 bytes of data to the motor drives, being two bytes with general information regarding the communication and six bytes with particular information for each of the seven servos; among the latter, two bytes contain flags for turning on/off the servo, activating the brake, selecting the torque/velocity mode, etc.; the other four bytes indicate the desired torque and velocity for the corresponding actuator. On the other hand, 72 bytes are transmitted from the drives to the computer, being two

with general information and 10 for each servomotor, among which we find the status data (on/off, brake, etc.), and the actual joint position, velocity and torque of each motor. More details on this can be found in (Mitsubishi, 2002b).
It is important to highlight that, when using the motion control board and the PA library functions, the user has no access to all this information, and, in fact, he cannot configure the robot in torque mode. In the following subsections we describe the hardware and software interfaces we developed for overcoming such difficulties.

5.2 Hardware interface

In order to work with the PA10 arm in torque mode, it was necessary to have access to the signals commanded to the motor drives. For doing so, we decided to use another PC, equipped with an ARCNET board. This has been done by other authors, for example (Jamisola et al., 2004).
In (Contemporary, 2005) it is explained how a PCI22-48X ARCNET board from Contemporary Controls can be modified to make it work with the PA10 robot. In the same document it is mentioned that even though the PCI22-485X boards are now discontinued, the replacement parts are the PCI20U-485X and the PCI20U-400, from the same brand; all of them have the same ARCNET controller (COM20022); the only difference is the circuit for transmission/reception (known as *transceiver*), which is in the daughter board soldered to the base board.
For the application presented in this work, we acquired a PCI20U-485X board, which was installed in the PCI bus of a PC with a double-core processor, running at 2.4 GHz. This board was configured with the same ARCNET setup of the motion control board. On the other hand, for the sake of compatibility, we made an optical interface board, similar to the one in the original PA10 system.
Following the steps in (Contemporary, 2005), the daughter board with the transceiver was withdrawn from the PCI20U-485X and in its place we put a connector for the cable linking the signals from the main board with the optical interface; this latter plays the role of transceiver and converts the electric signals in optical signals to be transmitted by the optical fiber.
Figure 5 shows the connections required to use the PCI20U-485X board.

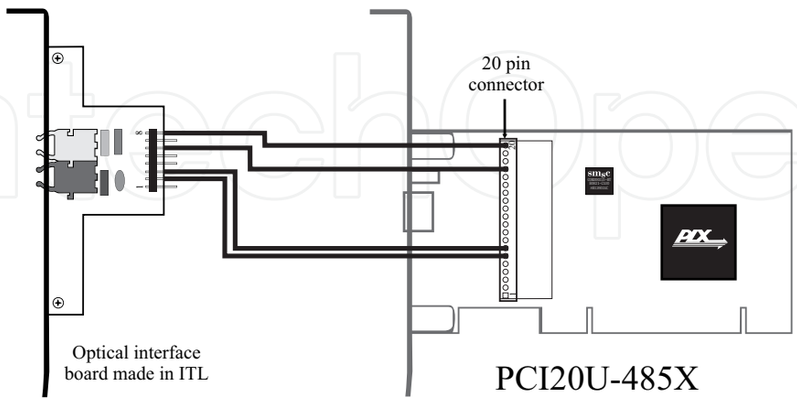

Fig. 5. Connection between the ARCNET board and the optical board

5.3 Real–time software

Real–time control of industrial processes has become an important topic in the recent years. Real–time systems are those in which it is imperative that their response occurs within a limit time. There are real–time operating systems that facilitate the development of this kind of applications. Microsoft Windows is not a real–time operating system, but it can behave as if one, by using an additional software, such as the RTX (real–time extension) from IntervalZero (formerly Ardence).

RTX allows Windows to handle two type of executable programs at the same time: (a) those with extension .RTSS, which are handled completely by the RTX real–time subsystem, thus being more predictable; and (b) those with extension .EXE, handled by the basic Windows subsystem (Win32), which are not so severe with time constraints, but they can include RTX functions and also use graphic interfaces. Thanks to RTX it is possible to simultaneously execute several real–time programs, and share information among them, by means of a shared memory.

To develop real–time applications with RTX, the Visual C++ IDE is required. It was with this platform that we designed the software interface to control the PA10 robot, without using the motion control board. Figure 6 shows a general diagram of the components of the developed software.

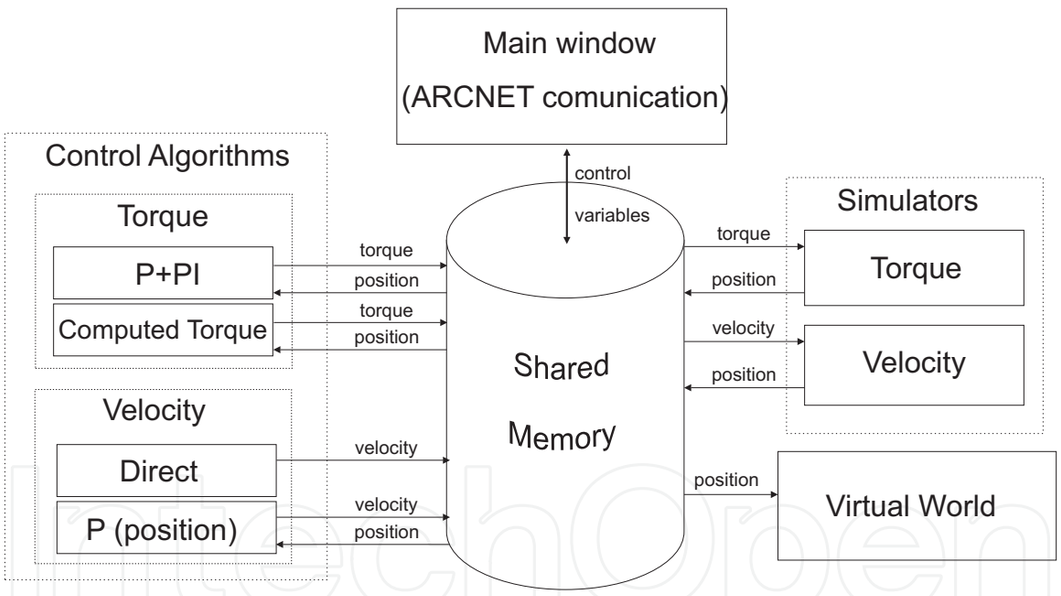

Fig. 6. Components of the control software

The central part of the system is the interchange of information among the different programs via shared memory. These programs are briefly described in the following subsections.

5.3.1 Main window

When starting the main program the user is being asked on what operation mode (velocity or torque) he wants to use for controlling the robot. After that, the main window of Figure 7 shows up; by using this window the user can select the control algorithm to execute, as well as enabling the simulator and the virtual world; further, the window is useful for monitoring and, if necessary, modifying the torque or velocity signals.

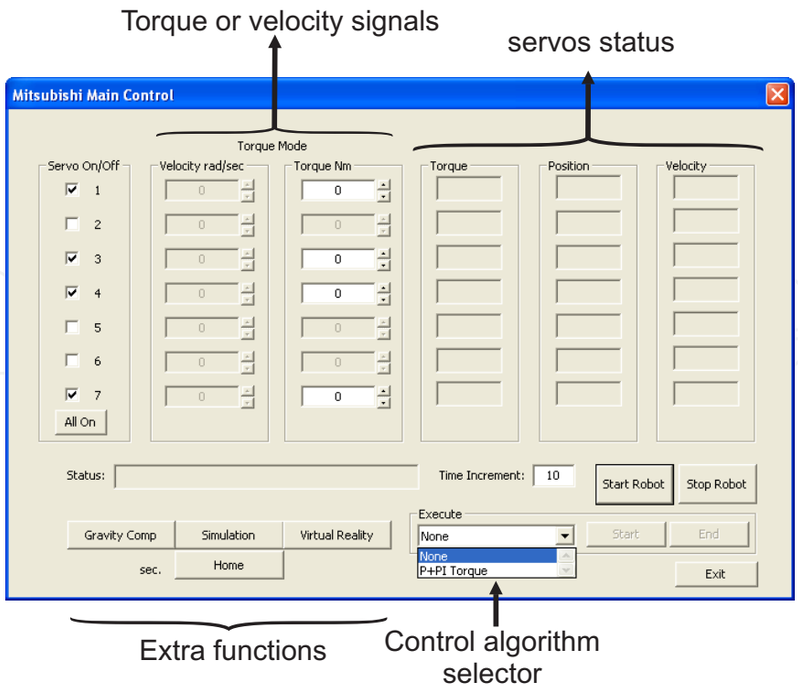

Fig. 7. Main window of the control software

It is important to mention that this program establishes the communication with the servo drives via ARCNET. This communication starts and stops by pressing the "Start Robot" and "Stop Robot" buttons in the window.

5.3.2 Simulators

Depending on the operation mode selected, a simulator of the robot can be executed, which resolves the corresponding model in order to obtain the joint coordinates of the robot, starting from the torque or velocity input.

5.3.3 Control algorithms

They are a series of Visual C++ programs which implement different control laws, which can be selected depending on the operation mode selected at the beginning. The number of programs increases when new controllers are added.

5.3.4 Virtual world

It shows, via a 3D animation, the behavior of the robot in real–time, either in simulation or in a real experiment (see Figure 8).

6. Experimental Results

In order to evaluate the performance of the developed real–time software for controlling the PA10-7CE robot arm, we carried out some experiments. The idea was to test the implementation of the two–loop controller studied in Section 2.2 in three different operation modes.

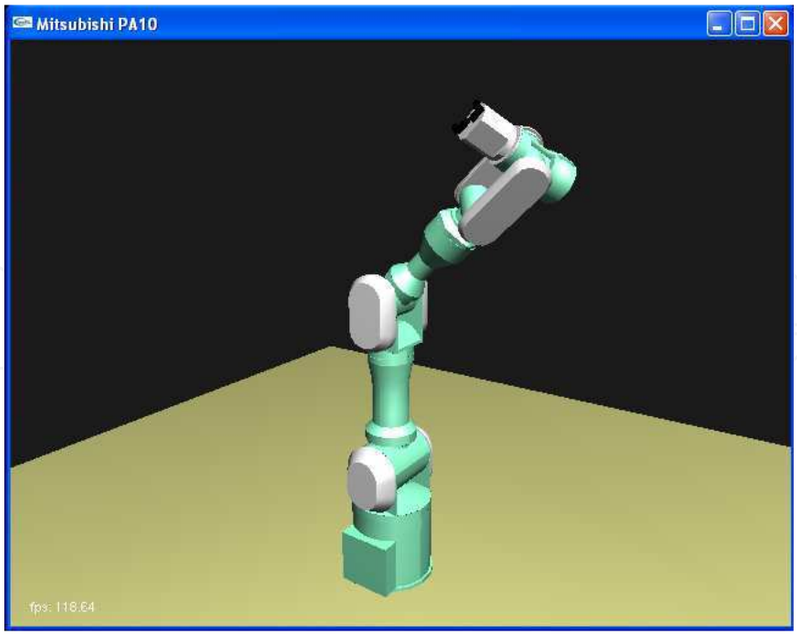

Fig. 8. Virtual world of the PA10 robot

Summing up, the following three cases were considered, being the difference how the inner and outer controllers were implemented (either in the control computer or in the servo–drives):

A. Both RMRC and PI controllers in the real–time software; drives in torque mode.
B. RMRC controller in the real–time software; PI in drives, configured in velocity mode.
C. Both RMRC and PI controllers in the original system, with the motion control board.

The control aim of the experiments was to track a desired time–varying trajectory. In order to show the benefits of redundancy in the PA10-7CE, a secondary task was designed, consisting in maximizing the distance of the joint positions to the actuator limits.

6.1 Desired trajectory

The primary task consisted in tracking a straight line in Cartesian coordinates, subject to the following constraints:

- The robot's end–effector should move only along the Y axis, with respect to the base coordinate frame; hence, x and z coordinates should remain constant.
- The end–effector should perform a turn around the Z axis of the base coordinate frame; hence, β and γ should remain constant.
- Variable operational coordinates (y and α) should perform a cycloidal–type motion, which is given by the following expression (Sciavicco & Siciliano, 2000):

$$s(t) = s_i + (s_f - s_i)\left[\frac{t}{T} - \frac{1}{2\pi}\text{sen}\left(\frac{2\pi\, t}{T}\right)\right] \tag{54}$$

where $s(t)$ is the corresponding operational coordinate (as a function of time t), s_i and s_f are, respectively, the initial and final values of such $s(t)$, and T indicates the duration of the cycloidal motion.

As explained in (Sciavicco & Siciliano, 2000), the cycloidal motion given by (54) produces smooth velocity and acceleration profiles, which is a convenient feature in robotic tasks.
For the experiments, we chose the parameters in Table 1.

Parameter	Value	Unit
x	-0.339	m
y_i	-0.339	m
y_f	0.339	m
z	0.837	m
α_i	0	rad
α_f	1.5708	rad
β	0	rad
γ	0	rad
T	20	s

Table 1. Parameters for the desired trajectory

6.2 Secondary (redundant) task

The PA10-7CE arm is a 7–dof redundant manipulator for pose control tasks (requiring only 6 dof). Thus, we decided to take advantage of the redundant degree of freedom to perform a secondary task, which should not affect the primary task (tracking of the desired trajectory) but only the self–motion of the robot joints.
According to the analysis in Section 2, the secondary task is added to the controller by means of vector $\dot{\boldsymbol{q}}_0$ in (13). Among the different methods of choosing vector $\dot{\boldsymbol{q}}_0$, those considering it as the gradient of a suitable cost function, $H(\boldsymbol{q})$, are the most common (Sciavicco & Siciliano, 2000). For the purpose of the experiments in this chapter, we chose the following cost function:

$$H(\boldsymbol{q}) = \sum_{i=1}^{7} \left(\frac{q_i - \bar{q}_i}{q_{i_{max}} - q_{i_{min}}} \right)^2 \tag{55}$$

where $q_{i_{max}} - q_{i_{min}}$ is the range of operation for the i-th joint, and $\bar{q}_i$ is the central value of this range. Minimizing this cost function implies keeping each of the joints near to their corresponding central values, that is, far from their joint limits.
In order to minimize (55) we should make $\dot{\boldsymbol{q}}_0$ equal to its negative gradient; in other words, the i-th element of $\dot{\boldsymbol{q}}_0$ should be

$$\dot{\boldsymbol{q}}_{0_i} = -\lambda \frac{2(q_i - \bar{q}_i)}{(q_{i_{max}} - q_{i_{min}})^2}$$

where λ is a weighting factor.
For the experiments we used the actual joint limits of the robot, shown in Table 2, with $q_{i_{max}} = -q_{i_{min}}$, and $\bar{q}_i = 0$ for each link i.

6.3 Results

We carried out three sets of experiments, each to test the performance during the execution of the two–loop controller shown in Figure 1, but using different operation modes. The same primary and secondary tasks (mentioned previously) were chosen for the three cases. The

Joint	Limit	Unit
1	180	deg
2	97	deg
3	180	deg
4	143	deg
5	270	rad
6	180	deg
7	270	deg

Table 2. Joint limits for the secondary task

gains of each controller were tuned so that a good performance of the tracking task could be perceived.
Prior to executing the desired task in operational space, a joint position controller was used to move the robot to the required initial pose.

6.3.1 Case A

In this case, the two controllers (outer RMRC and inner velocity PI) where implemented in the developed real–time application, using a sampling period of 10 ms. The control gains were chosen as follows,

- RMRC controller:

$$K = \text{diag}\{40, 50, 40, 12, 12, 12\}$$

- PI controller:

$$K_p = \text{diag}\{80, 120, 40, 48, 5, 8, 15\}$$
$$K_i = \text{diag}\{36, 50, 24, 20, 10, 10, 10\}$$

The weighting factor for the secondary task was chosen to be $\lambda = 2.5$. The servo–drives where configured in torque mode.

6.3.2 Case B

For this case, only the RMRC controller was implemented in the control computer, with the control gains:

$$K = \text{diag}\{24, 24, 24, 8, 8, 8\}$$

The servo–drives were configured in velocity mode, so that the inner PI velocity controller was implemented in them. Factor λ had to be reduced in order to obtain a better performance; its final value was $\lambda = 0.7$.

6.3.3 Case C

In this case we did not used the same software program in the control computer. Instead, we employed an application using a Windows multimedia timer and some API functions from the PA library, to send the desired operational coordinates to the motion control board. The control scheme was also chosen via API functions, in a way of implementing the RMRC in the control board and configuring the servo–drives in velocity mode so as to use their inner PI velocity loops.

6.3.4 Discussion

In order to compare the performance of the three cases, we decided to compute the Euclidean norm of both the position error $\|\tilde{\boldsymbol{p}}\|$ and the orientation error $\|\tilde{\boldsymbol{\phi}}\|$. Figures 9 and 10 show the evolution of such norms for the three cases.

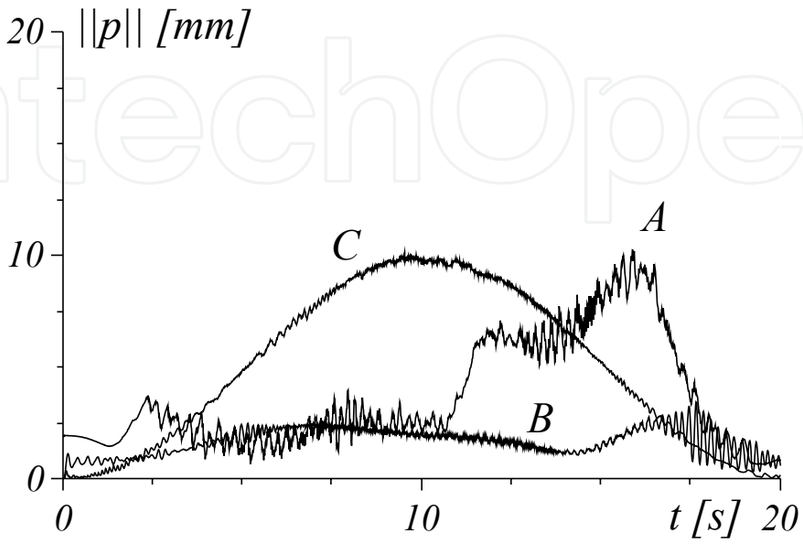

Fig. 9. Time evolution of the norm of the position error.

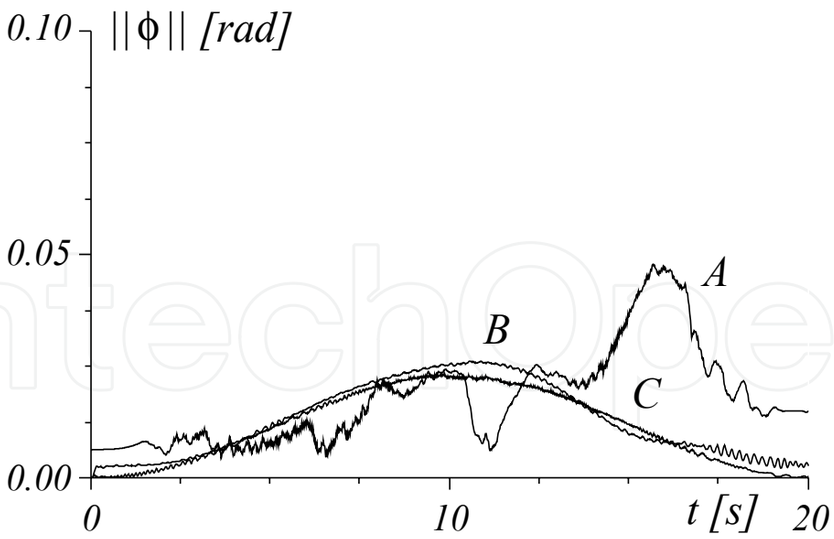

Fig. 10. Time evolution of the norm of the orientation error.

Figures show that even though Case C corresponds to the original setup for the PA10, in the case of the position error, we obtain a better performance when using the drives configured in velocity mode (Case B). As shown in Figure 10, the orientation errors for cases B and C are quite similar. Regarding the results of Case A (drives configured in torque mode) we notice a

more irregular shape of the norms. Some error peaks appear in the last half of the trajectory (when the velocity is decreasing); this is probably due to static friction, which appears at low velocities.
To have a better appreciation of the performance for the three cases, we decided to use a standard index: the root mean square (RMS) value of the norm of the corresponding error, computed in a trip of time T, that is

$$RMS(\tilde{\boldsymbol{u}}) \quad = \quad \sqrt{\frac{1}{T}\int_0^T \|\tilde{\boldsymbol{u}}(\sigma)\|^2 \, d\sigma}$$

where $\tilde{\boldsymbol{u}}$ can be either $\tilde{\boldsymbol{p}}$ or $\tilde{\boldsymbol{\phi}}$. Table 3 shows the performance indexes obtained during the execution of the trajectory ($T = 20$ s):

Index	Case A	Case B	Case C
$RMS(\tilde{\boldsymbol{p}})$	4.379	1.816	6.065
$RMS(\tilde{\boldsymbol{\phi}})$	0.021	0.015	0.014

Table 3. Performance indexes

It is worth noticing that the best performance is obtained with Case B (velocity mode). Further, even though Case A (torque mode) seems to produce more oscillations (see figures 9 and 10), the performance index for the position is lower for Case A than for Case C.

7. Conclusions

In this chapter, we have dealt with the motion control problem in operational space, using the resolved motion rate controller RMRC (kinematic control) plus the intrinsic joint velocity PI controller of the industrial robots, whose solutions of the overall closed–loop system have been proved to have uniformly ultimately boundedness.
Due to its open architecture system, the PA10 has become a suitable robot for both research and industrial applications. This leads to the need of studying the operation of each of the sections that compose its hierarchical structure.
Some experiments were carried out in a PA10-7CE arm, using a software program we have developed for control in real–time. The same hierarchical structure was tested in different operation modes. The results show good performance in all cases, even though the operation in torque mode is more affected by mechanical vibrations, perhaps due to friction and to the discretization of the controllers.

8. Acknowledgements

This work is partially supported by DGEST, PROMEP and CONACyT (grant 60230), Mexico.

9. References

Aicardi, M.; Caiti, A.; Cannata, G & Casalino, G. (1995). Stability and robustness analysis of a two layered hierarchical architecture for a closed loop control of robots in the operational space. *Proceedings of IEEE International Conference on Robotics and Automation*, Nagoya, Japan, 1995.

Camarillo, K.; Campa, R. & Santibáñez, V. (2006). Control of the Mitsubishi PA10-7CE robot using inner velocity PI loops (in Spanish). *Proceedings of 2006 Mexican Congress on Robotics (COMRob06)*, Mexico City, Mexico, October 2006.

Camarillo, K.; Campa, R.; Santibáñez, V. & Moreno-Valenzuela, J. (2008). Stability analysis of the operational space control for industrial robots using their own joint velocity PI controllers. *Robotica*, Vol. 26, No. 6, November 2008, 729-738.

Campa, R.; Torres, E.; Salas, F. & Santibáñez, V. (2008). On modeling and parameter estimation of brushless DC servoactuators for position control tasks. *Proceedings of IFAC World Congress*, Seoul, Korea, July 2008.

Craig, J. J. (2004). *Introduction to Robotics: Mechanics and Control*, Prentice–Hall, 2004.

Contemporary Control Systems (2005). Modification of a Contemporary Controls ARCNET card to control a Mitsubishi Heavy Industries, Inc. PA10 robot arm. Online technical document: www.ccontrols.com/pdf/PA10 procedure.pdf

Denavit, J. & Hartenberg, E. (1955). A kinematic notation for lower–pair mechanisms based on matrices. *Journal of Applied Mechanics*. Vol. 77, 1955, 215-221.

Higuchi, M., Kawamura, T.; Kaikogi, T.; Murata, T. & Kawaguchi, M. (2003) Mitsubishi clean room robot. Mitsubishi Heavy Industries, Ltd., Technical Review, Vol. 40, No. 5, 2003.

Jamisola, R. S.; Maciejewski, A. A. & Roberts, R. G. (2004). Failure–tolerant path planning for the PA-10 robot operating among obstacles. *Proceedings of IEEE International Conference on Robotics and Automation*, New Orleans, LA, April 2004.

Kelly, R. & Moreno, J. (2005). Manipulator motion control in operational space using joint velocity inner loops. *Automatica*, Vol. 41, 2005, 1423-1432.

Kelly, R.; Santibáñez, V. & Loría, A. (2005). *Control of Robot Manipulators in Joint Space*, Springer–Verlag, London, 2005.

Kennedy, C. & Desai, J. P. (2003). Force feedback using vision. *Proceedings of IEEE International Conference on Advanced Robotics*, Coimbra, Portugal, 2003.

Khalil, H. (2005). *Nonlinear Systems*, Prentice Hall, New York, 2005.

Khatib, O. (1987). A unified approach for motion and force control of robot manipulators. *IEEE Journal on Robotics and Automation*, Vol. 3, No. 1, 1987, 43-52.

Mitsubishi Heavy Industries (2002a). Instruction manual for installation, maintenance & safety. General Purpose Robot PA10 series, document 91-10023, 2002.

Mitsubishi Heavy Industries (2002b). Instruction manual for servo driver. General Purpose Robot PA10 series, document SKC-GC20004, 2002.

Nakamura, Y. *Advanced Robotics: Redundancy and Optimization*, Addison-Wesley, 1991.

Oonishi, K. (1999). The open manipulator system of the MHIPA-10 robot. *Proceedings of International Symposium on Robotics*, Tokio, Japan, October 1999.

Pholsiri, C. (2004). Task decision making and control of robotic manipulators. Ph.D. Thesis, The University of Texas at Austin, Austin, TX, 2004.

Qu, Z. & Dorsey, J. (1991). Robust tracking control of robots by a linear feedback law. *IEEE Transactions on Automatic Control*, Vol. 36, No. 9, 1991, 1081-1084.

Ramírez, C. (2008). Dynamic modeling and torque–mode control of the Mitsubishi PA10-7CE robot. Master's Thesis (in Spanish). Instituto Tecnológico de la Laguna, Torreón, Mexico, December 2008.

Ramírez, C. & Campa, R. (2008). Development of a system for real–time control of the Mitsubishi PA10 robot (in Spanish). *Proceedings of 2008 Mexican Congress on Robotics (COMRob08)*, Mexico City, Mexico, September 2008.

Sciavicco, L. & Siciliano, B. (2000). *Modelling and Control of Robot Manipulators*, Springer–Verlag, London, 2000.
Whitney, D. E. (1969). Resolved motion rate control of manipulators and human prostheses. *IEEE Transactions on Man-Machine Systems*, Vol. 10, No. 2, June 1969, 47-53.

IntechOpen

IntechOpen

Advances in Robot Manipulators

Edited by Ernest Hall

ISBN 978-953-307-070-4

Hard cover, 678 pages

Publisher InTech

Published online 01, April, 2010

Published in print edition April, 2010

The purpose of this volume is to encourage and inspire the continual invention of robot manipulators for science and the good of humanity. The concepts of artificial intelligence combined with the engineering and technology of feedback control, have great potential for new, useful and exciting machines. The concept of eclecticism for the design, development, simulation and implementation of a real time controller for an intelligent, vision guided robots is now being explored. The dream of an eclectic perceptual, creative controller that can select its own tasks and perform autonomous operations with reliability and dependability is starting to evolve. We have not yet reached this stage but a careful study of the contents will start one on the exciting journey that could lead to many inventions and successful solutions.

How to reference

In order to correctly reference this scholarly work, feel free to copy and paste the following:

Ricardo Campa, Cesar Ramirez, Karla Camarillo, Victor Santibanez and Israel Soto (2010). Motion Control of Industrial Robots in Operational Space: Analysis and Experiments with the PA10 Arm, Advances in Robot Manipulators, Ernest Hall (Ed.), ISBN: 978-953-307-070-4, InTech, Available from: http://www.intechopen.com/books/advances-in-robot-manipulators/motion-control-of-industrial-robots-in-operational-space-analysis-and-experiments-with-the-pa10-arm

INTECH
open science | open minds

InTech Europe

University Campus STeP Ri
Slavka Krautzeka 83/A
51000 Rijeka, Croatia
Phone: +385 (51) 770 447
Fax: +385 (51) 686 166
www.intechopen.com

InTech China

Unit 405, Office Block, Hotel Equatorial Shanghai
No.65, Yan An Road (West), Shanghai, 200040, China
中国上海市延安西路65号上海国际贵都大饭店办公楼405单元
Phone: +86-21-62489820
Fax: +86-21-62489821

© 2010 The Author(s). Licensee IntechOpen. This chapter is distributed under the terms of the Creative Commons Attribution-NonCommercial-ShareAlike-3.0 License, which permits use, distribution and reproduction for non-commercial purposes, provided the original is properly cited and derivative works building on this content are distributed under the same license.

IntechOpen

IntechOpen