Vallés Miquel, M.; Díaz-Rodríguez, M.; Valera Fernández, Á.; Mata Amela, V.; Page del Pozo, AF. (2012). Mechatronic development and dynamic control of a 3-DOF parallel manipulator. Mechanics Based Design of Structures and Machines: An International Journal. 40(4):434-452. doi:10.1080/15397734.2012.687292.

# MECHATRONIC DEVELOPMENT AND DYNAMIC CONTROL OF A 3 DOF PARALLEL MANIPULATOR

**Marina Vallés[*], Miguel Díaz-Rodríguez[**], Ángel Valera[*] , Vicente Mata[†], Álvaro Page[‡]**

[*]Instituto Universitario de Automática e Informática Industrial
Universitat Politècnica de València, Camino de Vera s/n, 46022 Valencia, Spain
e-mail: {mvalles, giuprog}@isa.upv.es

[**]Departamento de Tecnología y Diseño, Facultad de Ingeniería
Universidad de los Andes, La Hechicera, 5101 Mérida, Venezuela
e-mails: dmiguel@ula.ve

[†]Centro de Investigación en Tecnología de Vehículos
Universitat Politècnica de València, Camino de Vera s/n, 46022 Valencia, Spain
e-mails: vmata@mcm.upv.es

[‡]Dpto. Física Aplicada
Universitat Politècnica de València, Camino de Vera s/n, 46022 Valencia, Spain
e-mails: alvaro.page@ibv.upv.es

**Keywords:** Parallel Manipulator, Robot Control, Mechatronics, Kinematics, Dynamics.

**Abstract.** *The aim of this paper is to develop, from the mechatronic point of view,a low-cost parallel manipulator (PM) with 3-Degrees of Freedom (DOF). The robot has to be able to generate and control one translational motion (heave) and two rotary motions (rolling and pitching). Applications for this kind of parallel manipulator can be found at least in driving-motion simulation and in the biomechanical field. An open control architecture has been developed for this manipulator, which allows implementing and testing different dynamic control schemes for a PM with 3-DOF. Thus, the robot developed can be used as a test bench where control schemes can be tested. In this paper, several control schemes are proposed and the tracking control responses are compared. The schemes considered are based on passivity-based control and inverse dynamic control. The control algorithm considers point-to-point control or tracking control. When the controller considers the system dynamics, an identified model has been used. The control schemes have been tested on a virtual robot and on the actual prototype.*

## 1 INTRODUCTION

A Parallel Manipulator (PM) consists of a mobile platform connected to a fixed base by means of several kinematic chains. These manipulators have the end-effector attached to the mobile platform. PMs have advantages over their counterpart serial robots, essentially because the load is shared by the several links connecting the mobile platform to the base, thus giving PMs high stiffness, high load-carrying capacity, high speed, and high accuracy. However, PMs have small workspaces and singularity problems. In addition, the forward position solution, the system dynamics and the control of PMs are difficult to develop compared to a serial robot.

Due to their advantages, PMs have several applications. These manipulators have been implemented as motion simulators, tire-testing machines, flight simulators and medical applications. Various PM mechanical architectures and applications can be found in (Stewart, 1965), (Gough and Whitehall, 1962), (Merlet, 2000), (Tsai, 1999). Research on PMs was first focused on 6-DOFplatforms. However, 6-DOF is not always required for many applications. Hence, attention is been paid on the development of PMs with less than 6-DOF. The reason is that a PM with limited DOF maintains the inherent advantages of parallel mechanisms but offers additional benefits such as the reduction of total costs in manufacturing and operations. Examples of limited DOF PM are: the well-known Delta Robot with 3 Translational (T) DOF (Clavel, 1988), which is well-suitable for pick-and-place tasks, the 3Torthoglide robot for high-speed machining (Chablat, 2003). Translational3TPM have also been used in medical applications such as cardiopulmonary resuscitation equipment (Li and Xu, 2007). When combined translational and rotary motions are required, 3-R<u>P</u>S (Lee and Shah, 1988), (Kim and Tsai, 2003) and 3-<u>P</u>RS (Carretero *et al.*, 2000), (Merlet, 2002) architectures have been proposed, where R, P and S stand for the revolute, prismatic and spherical joints, respectively, and the underlined text (<u>P</u>) indicates the actuated joint.

The purpose of this paper is twofold. The first one is to develop, starting from scratch, the complete mechatronic design of a low-cost robot with 3-DOF. The manipulator has to be able to generate and control one translational motion (1T) and two rotary motions (2R) (rolling and pitching). With this type of motion (1T2R), driving motion can be simulated. The roll angle reproduces cornering while the pitch angle creates the illusion of acceleration and braking. The up and down motion can be reproduced by controlling the heave. Others application of this type of motion are: telescopic applications (Carretero *et al.*, 2000), fine positioning of a surgical tool (Merlet, 2002), and for ankle injury treatment in which the mobile platform simulates the foot trajectory during physiotherapy exercises (Syrseloudis and Emiris, 2008).

The second objective of the paper is to develop an open control architecture, which allows the implementation and testing of dynamic control schemes for PM with 3-DOF. The reason for building an open architecture is that the control is a field where there is still great potential for study in order to improve its accuracy (Paccot *et al.*, 2009). Thus, the robot developed can be used as a control scheme test bench. In this paper, several control schemes are proposed and the tracking of control responses is compared. The schemes considered are based on passivity-based control (Ortega and Spong, 1989) and inverse dynamic control (Rosillo *et al.*, 2011). The control algorithm considers point-to-point control or tracking control. When the controller considers the system dynamics, an identified model (Díaz-Rodriguez *et al.*, 2008) has been used. The control schemes have been tested over a virtual robot and an actual prototype.

The remainder of this paper is organized as follows. Section 2 concerns the kinematics of the 3-DOF parallel robot. Section 3 is devoted to the mechatronic design of the robot. Section

4 deals with the implemented control schemes. Section 5 presents the results while Section 6 summarizes the main conclusions.

## 2 PARALLEL ROBOT DESIGN

The choice of parallel robot architecture and movement is guided by the need to develop a low-cost robot capable of generating angular rotation in two axes (roll and pitch) and heave as a linear motion. Two alternative design architectures were considered: 3-RPS and 3-PRS. The 3-PRS architecture was selected after comparing the advantages and disadvantages of each one of the alternatives. For instance, one of the advantages of PRS architecture is that the actuators are located at the fixed base. In the case of 3-RPS architecture, the actuators move around the revolution joints. Another aspect for considering the PRS architecture is that the number of parameter for the dynamics model is lower than the RPS; see (Díaz-Rodriguez *et al.*, 2010) (25 rigid body parameters for the 3-RPS and 19 for the PRS), which is an important aspect when dealing with dynamic parameter identification and model-based control.

### 2.1 Physical description of the Designed Robot

Fig. 1 shows a CAD model of the robot designed. The physical system consists of three legs connecting the moving platform to the base. Each leg consists of: 1) a motor, which drives a ball screw, 2) a slider and 3) a connecting rod. The lower part of the ball screws are perpendicularly attached to the base platform. The positions of the ball screws at the base are in equilateral triangle configuration. The ball screw transforms the rotational movement of the motor into linear motion. The prismatic joint (P) is assumed to be between the sliders and the corresponding ball screw. The connecting rod is joined to the upper part of the ball screw by means of a revolution joint (R). The moving platform is joined to the connecting rod through spherical joints (S). In this way, each leg has PRS joints.

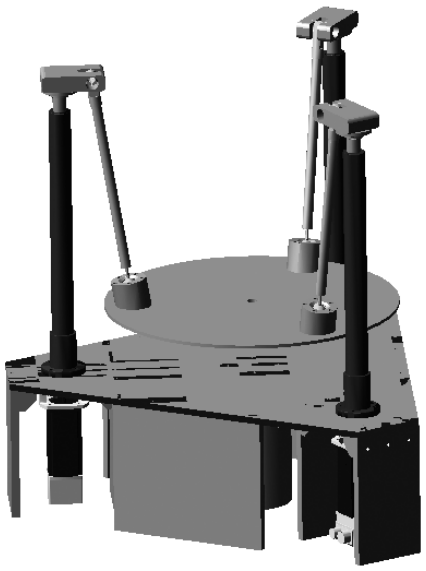

Figure 1: CAD model of the 3-PRS robot

## 2.2 Direct Kinematics

Given the actuators' linear motions, the direct kinematics of a PM consists of finding the roll ($\gamma$) and pitch ($\beta$) angles and the heave ($z$).The Denavit-Hartenbert (D-H) notation can be used to establish the generalized coordinates of the kinematic model. Table 1 shows the D-H parameters for the robot considered. From the table it can be seen that with 9 generalized coordinates, robot kinematics can be defined. The location of the coordinate systems for modeling the kinematics is shown in Fig. 2.

| $i$ | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 |
|---|---|---|---|---|---|---|---|---|---|
| $d_i$ | $q_1$ | 0 | 0 | 0 | 0 | $q_6$ | 0 | $q_8$ | 0 |
| $a_i$ | *0* | 0 | $l_a$ | 0 | 0 | 0 | 0 | 0 | 0 |
| $\theta_i$ | π/6 | $q_2$ | $q_3$ | $q_4$ | $q_5$ | 5π/2 | $q_7$ | -π/2 | $q_9$ |
| $\alpha_i$ | *0* | π/2 | 0 | π/2 | π/2 | 0 | π/2 | 0 | π/2 |

Table 1: D-H Parameters for the 3-DOF PM.

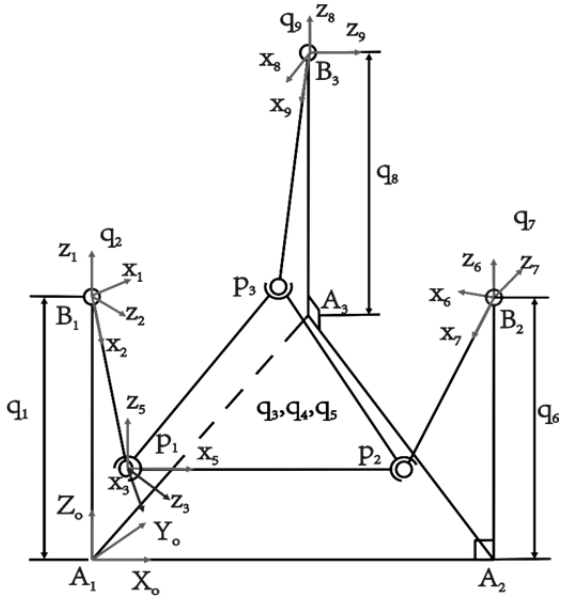


Figure 2: Location of the coordinate systems.

The robot has 3-DOF. Applying the geometric approach it can be seen that the length between $p_i$and $p_j$ is constant and equal to $l_m$. Thus,

$$f_1(q_1,q_2,q_6,q_7)=\left\|\left(\vec{r}_{A_1B_1}+\vec{r}_{B_1p_1}\right)-\left(\vec{r}_{A_1A_2}+\vec{r}_{A_2B_2}+\vec{r}_{B_2p_2}\right)\right\|-l_m=0 \quad (1)$$

$$f_2(q_1,q_2,q_8,q_9)=\left\|\left(\vec{r}_{A_1B_1}+\vec{r}_{B_1p_1}\right)-\left(\vec{r}_{A_1A_3}+\vec{r}_{A_3B_3}+\vec{r}_{B_3p_3}\right)\right\|-l_m=0 \quad (2)$$

$$f_3(q_6,q_7,q_8,q_9)=\left\|\left(\vec{r}_{A_1A_3}+\vec{r}_{A_3B_3}+\vec{r}_{B_3p_3}\right)-\left(\vec{r}_{A_1A_2}+\vec{r}_{A_2B_2}+\vec{r}_{B_2p_2}\right)\right\|-l_m=0 \quad (3)$$

In the forward kinematics the position of the actuators is known, thus the system of equations (1)-(3) can be seen as a nonlinear system with $q_2$ , $q_7$ and $q_9$ as unknown. The Newton-

Raphson (N-R) numerical method is chosen to solve the nonlinear system. The method converges rather quickly (quadratic convergence) when the initial guess is close to the desired solution (Rosillo *et al.*, 2011).The method is iterative and can be written as,

$$\begin{bmatrix} q_2 \\ q_7 \\ q_9 \end{bmatrix}^{i+1} = \begin{bmatrix} q_2 \\ q_7 \\ q_9 \end{bmatrix}^{i} - \mathbf{J}_i^{-1} \begin{bmatrix} f_1(q_2, q_7) \\ f_2(q_2, q_9) \\ f_3(q_7, q_9) \end{bmatrix}^{i} \tag{4}$$

In the equation, *i* means that the variables and functions are evaluated at the iteration *i.* The matrix $\mathbf{J}$ is the Jacobian matrix of $f_i$ with respect to the variables $[q_2, q_7, q_9]$.The iterative process ends when,

$$\sqrt{\left(f_1(q_2, q_7)^i\right)^2 + \left(f_1(q_2, q_9)^i\right)^2 + \left(f_1(q_7, q_9)^i\right)^2} < \varepsilon \tag{4}$$

The parameter $\varepsilon$ is a small positive quantity established by the user.

The Newton method requires an initial approximation as close as possible to the solution value. In this case it is not a problem since the initial pose of the links connecting the platform to the actuator ($q_2$, $q_7$ and $q_9$) has values between 0 to $-\pi/2$, but it moves round to $-2\pi/5$. For the subsequent initial guess, the values of the previous pose of the robot are considered. Several simulations have been conducted in which the direct kinematics is solved for a given trajectory. By using $-2\pi/5$ as an initial starting guess has been found that equations (1)-(3) can be solved in real time.

The location of the mobile platform is defined using a local coordinate system attached to it. Having found the generalized coordinates for the robot's legs, the position of points $p_i$ can be found. These three points share the plane of the platform. Based on these points, the rotational matrix of the platform with respect to the base can be built. A local axis $X_p$ is defined as a unit vector $\vec{u}$ with the direction given by $p_1p_2$. The axis $Z_p$ is defined by the vector $\vec{v}$ and is an axis perpendicular to the plane defined by points $p_1$, $p_2$ and $p_3$. Finally, the axis $Y_p$ is defined by the direction of the axis $\vec{w}$, which is determined by the vector product between the $\vec{u}$ and $\vec{v}$ axes. The rotation matrix of the moving platform is given by,

$${}^{O}R_p = \begin{bmatrix} \vec{u}^T & \vec{v}^T & \vec{z}^T \end{bmatrix} \tag{5}$$

The remaining generalized coordinates $q_3, q_4, q_5$ can be found from the rotation matrix.

### 2.3 Inverse Kinematics

Given the roll ($\gamma$) and pitch ($\beta$) angle and the heave ($z$), the inverse kinematics consists of finding the actuators' linear motion. Using an X-Y-Z fixed-angle system; the rotational matrix can be defined as,

$$ {}^{O}R_{p} = \begin{bmatrix} c_\alpha c_\beta & c_\alpha s_\beta s_\gamma - s_\alpha c_\gamma & c_\alpha s_\beta c_\gamma - s_\alpha s_\gamma \\ s_\alpha c_\beta & s_\alpha s_\beta s_\gamma - c_\alpha c_\gamma & s_\alpha s_\beta c_\gamma - c_\alpha s_\gamma \\ -s_\beta & c_\beta s_\gamma & c_\beta c_\gamma \end{bmatrix} \tag{6} $$

In the above equation, $c_*$ and $s_*$ stand for *cos(*)* and *sin(*)*, respectively. Given $\gamma$ and $\beta$ the yaw angle ($\alpha$) can be found as follows,

$$ \alpha = \text{atan2}\left(s_\beta s_\gamma, \left(c_\gamma + c_\beta\right)\right) \tag{7} $$

Having found the angle $\alpha$, the remaining terms of the rotational matrix can be found. The actuator positions can be found by the following expressions (Merlet, 2000),

$$ q_1 = {p_x}^2 + {p_y}^2 + {p_z}^2 + 2h\left(p_x u_x + p_y u_y + p_z u_z\right) - 2g\,p_x - 2g\,h u_x + g^2 + h^2 \tag{8} $$

$$ \begin{aligned} q_6 = {} & {p_x}^2 + {p_y}^2 + {p_z}^2 - h\left(p_x u_x + p_y u_y + p_z u_z\right) + \sqrt{3}h\left(p_x v_x + p_y v_y + p_z v_z\right) \\ & + g\left(p_x - \sqrt{3}p_y\right) + g\,h\left(u_x - \sqrt{3}u_y\right)/2 + g\,h\left(v_x - \sqrt{3}v_y\right)/2 + g^2 + h^2 \end{aligned} \tag{9} $$

$$ \begin{aligned} q_8 = {} & {p_x}^2 + {p_y}^2 + {p_z}^2 - h\left(p_x u_x + p_y u_y + p_z u_z\right) - \sqrt{3}h\left(p_x v_x + p_y v_y + p_z v_z\right) \\ & + g\left(p_x - \sqrt{3}p_y\right) - g\,h\left(u_x - \sqrt{3}u_y\right)/2 + g\,h\left(v_x - \sqrt{3}v_y\right)/2 + g^2 + h^2 \end{aligned} \tag{10} $$

In the equation, $h = l_m/\sqrt{3}$, $g = l_b/\sqrt{3}$, $p_x = -h u_y$, $p_y = -h\left(u_x - v_y\right)$, $p_z = z$ and $l_b$ are the lengths between $A_i A_j$.

### 2.4 Dynamic Model

A mechatronic design is a combination of mechanical engineering, electronics, control systems and computers (Awtar *et al.*, 2002). Thus, an important aspect when developing a mechatronic device is the control system. One of the objectives of this paper is to develop an open control architecture allowing the implementation and testing of dynamic control schemes. Schemes such as inverse dynamics' control require writing the differential equations describing the equation of motion as follows,

$$ \mathrm{M}\left(\vec{q}, \vec{\Phi}\right) \cdot \ddot{\vec{q}} + \vec{C}\left(\vec{q}, \dot{\vec{q}}, \vec{\Phi}\right) + \vec{G}\left(\vec{q}, \vec{\Phi}\right) = \vec{\tau} \tag{11} $$

From equation (11) it can be seen that the mass matrix $\mathrm{M}$, the vectors corresponding to the centrifugal and Coriolis forces $\vec{C}$, and the gravitational forces depend on the dynamic parameters $\vec{\Phi}$. However, in order to identify the dynamic parameters, the model in linear parameter form has to be build first as follows as in (Díaz-Rodriguez *et al.,* 2010), (Awtar *et al.*, 2002),

$$ \mathrm{K}\left(\vec{q}, \dot{\vec{q}}, \ddot{\vec{q}}\right) \cdot \vec{\Phi} = \vec{\tau} \tag{12} $$

In equation (12) only one set of parameters can be identified because some of them make small or even insignificant contributions to the system's dynamic behavior. Moreover, they are prone to noise in the measurements and the unmodeled dynamics, see (García de Jalón and Bayo, 1994). Thus, a methodology previously developed by the authors (Awtar *et al.*, 2002) has been used in order to identify the dynamic model.

Once the identification process is carried out, the terms of the equation (11) can be found. First, the gravitational vector forces are obtained by zeroing the generalized velocities and accelerations in equation (12),

$$\mathrm{K}\left(\vec{q},\dot{\vec{q}}=0,\ddot{\vec{q}}=0\right)\cdot\vec{\Phi}=\vec{G}\left(\vec{q},\vec{\Phi}\right) \tag{13}$$

The mass matrix can be determined as follows,

$$\mathrm{K}\left(\vec{q},\dot{\vec{q}}=0,\vec{e}_i\right)\cdot\vec{\Phi}-\vec{G}\left(\vec{q},\vec{\Phi}\right)=\mathrm{M}_i\left(\vec{q},\vec{\Phi}\right) \tag{14}$$

In the equation, $\mathrm{M}_i\left(\vec{q},\vec{\Phi}\right)$ is the *i-th* column of the mass matrix and $\vec{e}_i=\left[0\ \ \ldots\ \ 1\ \ \ldots\ \ 0\right]^T$ is a column vector of value 1 in the *i-th* position. Due to constraints, $\mathrm{M}_j\left(\vec{q},\vec{\Phi}\right)$can be expressed in dependent and independent generalized coordinates,

$$\mathrm{M}^{ind}\left(\vec{q}_i,\vec{\Phi}\right)\cdot\ddot{\vec{q}}_{ind}+\mathrm{M}^{dep}\left(\vec{q}_i,\vec{\Phi}\right)\cdot\ddot{\vec{q}}_{dep} \tag{15}$$

Considering that the dependent accelerations can rewritten in terms of the independent accelerations ( $\ddot{\vec{q}}_{dep}=\mathrm{A}_{dep}{}^{-1}\cdot\vec{b}-\mathrm{X}\cdot\ddot{\vec{q}}_{ind}$ ), the following equation can be obtained,

$$\mathrm{M}^{ind}\left(\vec{q}_i,\vec{\Phi}\right)\cdot\ddot{\vec{q}}_{ind}+\mathrm{M}^{dep}\left(\vec{q}_{ind},\vec{\Phi}\right)\cdot\left(\mathrm{A}_{dep}{}^{-1}\cdot\vec{b}-\mathrm{X}\cdot\ddot{\vec{q}}_{ind}\right) \tag{16}$$

where $\mathrm{X}=\mathrm{A}_{dep}{}^{-1}\mathrm{A}_{ind}$ and $\mathrm{A}_{dep}$, $\mathrm{A}_{ind}$ are the components of the Jacobian matrix corresponding to the dependent and independent generalized coordinates. The vector $\vec{b}$ stands for the bias vector. As can be seen, when replacing the dependent accelerations, a vector depending on the velocities appears. Thus, this vector has to be added with the vector corresponding to centrifugal and Coriolis forces $\vec{C}$ . Finally, $\mathrm{M}\left(\vec{q},\vec{\Phi}\right)$ is built as follows,

$$\mathrm{M}\left(\vec{q},\vec{\Phi}\right)=\mathrm{M}^{ind}\left(\vec{q},\vec{\Phi}\right)-\mathrm{M}^{dep}\left(\vec{q},\vec{\Phi}\right)\mathrm{X}\left(\vec{q}\right) \tag{17}$$

The centrifugal and Coriolis terms depend on the generalized coordinates and velocities. By zeroing generalized accelerations in equation (12) again, it can be established that,

$$\mathrm{K}\left(\vec{q},\dot{\vec{q}},\ddot{\vec{q}}=0\right)\cdot\vec{\Phi}-\vec{G}\left(\vec{q},\vec{\Phi}\right)+\mathrm{M}^{dep}\left(\vec{q},\vec{\Phi}\right)\mathrm{A}_d{}^{-1}\vec{b}=\vec{C}\left(\vec{q},\dot{\vec{q}},\vec{\Phi}\right) \tag{18}$$

In summary, the differential equations describing the equation of motion (11) can be built using equations (12)-(18).

## 3 MECHATRONIC ROBOT DEVELOPMENT

The actual parallel robot developed for this paper is depicted in Fig.3.Each leg of the parallel manipulator is driven by a brushless DC servomotor equipped with power amplifiers. The actuators are AEROTECH BMS465 AH brushless servomotors. The motors are operated by AEROTECH BA10power amplifiers.

The AEROTECH BMS465 brushless, slot-less servomotor provides a very high torque, acceleration and smoothness. They can be equipped with a variety of encoder resolution options. In addition to the standard RS-422 line driver output, an optional amplified sine-wave encoder can be used to provide ultra-high resolution. The performance specifications of BMS465 motors are 2.86N-m stall torque (continuous), 11.43N-m peak torque and 2,000 rpm.

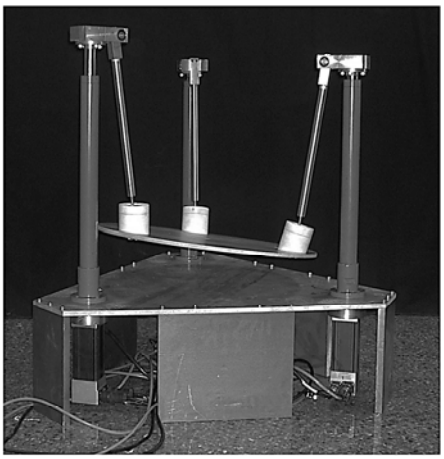

Figure 3: 3-PRS parallel robot implemented.

The AEROTECH BA10 amplifier is Aerotech's stand-alone drive for brushless or single-phase DC brush motors. This amplifier can run in velocity mode or torque mode using a self-commutating, low ripple, modified six-step algorithm. It accepts a standard ±10V DC as a velocity or torque (current) command from any motion controller. The continuous output current is 5A, with a peak output current of 10A. The BA10 amplifier is based on a 20 kHz IGBT for reliable operation in a compact package. It is completely self-contained, requiring only AC line power, and the amplifier is fully protected. The DC-isolated power stage minimizes loop noise. The amplifier accepts a quadrature encoder or tachometer input for velocity feedback. The encoder signal is converted to a voltage representing speed. The motors are operated in torque (current) mode for easy modeling and control of the robot. Experiments were conducted to find the relationship between current and torque. In Fig. 4 the experimental setup is depicted, which consists of a shaft, instrumented with strain gauges, and a flywheel representing the equivalent inertia of the actual system. This relationship was found to be linear for the prescribed operating range of the motors.

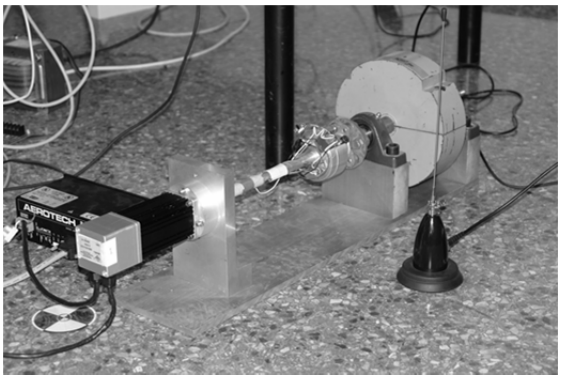

Figure 4: Experimental setup for finding the relationship between current and torque in motors

In order to implement the control architecture for the parallel robot, an industrial PC has been used. It is based on a high performance 4U Rackmount industrial system with 7 PCI slots and 7 ISA slots. It has a 3,06GHz Intel® Pentium® 4 processor and two GB DDR 400 SDRAM. The industrial PC is equipped with 2 Advantech™ data acquisition cards: a PCI-1720 and a PCL-833.

The PCI-1720 card has been used for supplying the control actions for each parallel robot actuator. It provides four 12-bit isolated digital-to-analog outputs for the Universal PCI 2.2 bus. It has multiple output ranges (0~5V, 0~10V, ±5V, ±10V), programmable software and an isolation protection of 2500 VDC between the outputs and the PCI bus. The PCL-833 card is a 4-axis quadrature encoder and counter add-on card for an ISA bus. The card includes four 32-bit quadruple AB phase encoder counters, an onboard 8-bit timer with a wide range time-based selector and it is optically isolated up to 2500V. Fig. 5 shows the control architecture based on an industrial PC developed for this study.

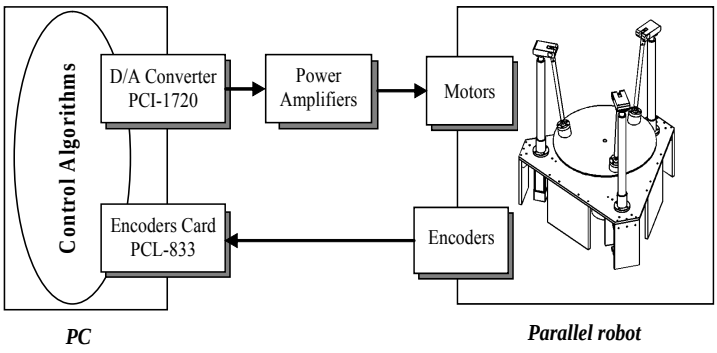


Figure 5: Robot control architecture.

## 4 DYNAMIC-BASED CONTROL SCHEMES

### 4.1 Passivity-based Control

In recent years, the passivity-based approach to robot control has gained a lot of attention. This approach solves the robot control problem by exploiting the robot system's physical structure, and specifically its passivity property. The design philosophy of these controllers is to reshape the system's natural energy in such a way that the tracking control objective is achieved (Ortega and Spong, 1989).

Table 2 shows the point-to-point (regulation) controllers based on passivity. They could be viewed as particular cases of the next general control law:

$$\tau_e = -K_p e - K_d v - u \tag{19}$$

where $e = q - q_d$ and $u$ and $v$ vary according to the kind of controller:

| **Controller (point-to-point problem)** | ***u*** | ***v*** |
|---|---|---|
| PD+G | $-G(q)$ | $\dot{q}$ |
| PD+G0 | $-G(q_d)$ | $\dot{q}$ |
| PID | $K_i \int_0^t e \, \mathrm{dt}$ | $\dot{q}$ |

Table 2: Passivity-based point-to-point controllers.

The first controller implemented was the PD with gravity term compensation. This controller is composed of two parts: the first part is a lineal feedback of the state and the second part is the gravity forces compensation.

The second control strategy was also proposed by the same authors. It is a variation on the first controller where gravity compensation is carried out in the desired final position.

These controllers are very simple but they have two main drawbacks: the first one is the computational complexity of the gravity term. Depending on the robot and/or its dynamic modelling, it can be so high that it is impossible to calculate it in real-time. On the other hand, the dead-zone phenomenon or any error in the gravity term estimation can cause a variation in the equilibrium point and therefore a stationary position error. A practical solution to attempt to solve these problems is to insert an integral action into the control law. These laws are basically the same as the PD, but the gravity compensation has been changed by the error integral.

For the tracking problem, the kinetic and potential energy must be modified as required in passivity-based controllers. The general expression of the controllers that can be found in the literature is as follows:

$$\tau_e = M(q)a + C(q,\dot{q})v_1 + G(q) - K_p e - K_d v_2 \tag{20}$$

where $a$, $v_1$, $v_2$ and $e$ varies according to the kind of controller (see Table 3). In all these controllers the control law has two parts, robot dynamics compensation and a proportional and differential controller.

| **Tracking Controllers** | ***a*** | ***v₁*** | ***e*** | ***v₂*** |
|---|---|---|---|---|
| Paden, Panja | $\ddot{q}_d$ | $\dot{q}_d$ | $q - q_d$ | $\dot{e}$ |
| Slotine, Li | $\ddot{q}_r$ | $\dot{q}_r$ | $0$ | $\dot{e} + A_1 e$ |
| Sadegh, Horowitz | $\ddot{q}_r$ | $\dot{q}_r$ | $q - q_d$ | $\dot{e} + A_1 e$ |

Table 3: Passivity-based tracking controllers.

The first controller is a variation of the PD with gravity compensation. The second one is a tracking controller based on the sliding mode theory. In the last one, some modifications are

introduced into the control law and into the energy function. It allows for probing the system's asymptotic stability using the Lyapunov theory.

### 4.2 Inverse Dynamic Control

Some controllers have been implemented with the control architecture depicted previously. The first class of controllers is based on the inverse dynamic method. This control approach makes a regular static state feedback that transforms the nonlinear system into a linear one (this is known as the inverse dynamic or feedback linearization problem). Assuming the dynamic model as:

$$x^{(n)} = f(x) + b(x)u \tag{21}$$

where *f(x)* is a nonlinear state function and *u* is the control input. If you use the expression:

$$u = \frac{1}{b}[v - f] \tag{22}$$

The nonlinearities will be cancelling, and the simple input-output relationship will be obtained:

$$x^{(n)} = v \tag{23}$$

where *v* is a new lineal input vector to be defined below.

In the robot case, the controllers based on the inverse dynamics could be viewed as particular cases of the following general control law (Ortega and Spong, 1989):

$$\tau_c = M(q)v + C(q,\dot{q})\dot{q} + G(q) \tag{24}$$

Inverse dynamic control (24) shows how the nonlinearities such as Coriolis terms as well as gravity terms can be simply compensated for by adding these forces to the control input. As shown in Table 4, depending on the expression *v*, different controllers can be obtained:

| **Control algorithm** | ***v*** |
|---|---|
| Point-to-point control | $-K_d\dot{q} - K_p e$ |
| Tracking control | $\ddot{q}_d - K_d\dot{q} - K_p e$ |
| Tracking control with integral action | $\ddot{q}_d - K_d\dot{q} - K_p e - K_i \int_0^t e(u)du$ |

Table 4: Inverse dynamic controllers.

The first controller implemented was the point-to-point controller. In this case, proportional and derivative terms compose the linear auxiliary control input *v*, and the robot system is exponentially stable by a suitable choice of the matrices $K_d$ and $K_p$.

The second controller is very similar to the first, but in this case the robot must follow a given time-varying trajectory $q_d(t)$ and its successive derivatives $\dot{q}_d$ and $\ddot{q}_d$, which describe the desired velocity and acceleration respectively. This tracking control is very simple but it

has several drawbacks: any error in the estimation of the robot dynamics can cause a variation in the equilibrium point and therefore a position error. The second problem that can occur is related to the dead-zone phenomenon: in this case the static friction in the motor shafts can also provoke a position error. A practical solution to attempt to solve these problems is to insert an integral action into the control law. This is the case of the last controller, where the integral of the error has been added.

### 4.3 Real-Time Control Implementation

The control unit developed for this study is based on an industrial PC. It is a totally open system and it gives a powerful platform for programming high level tasks. Thus any controller and/or control technique can be programmed and implemented, such as automatic trajectory generation, control based on external sensing using a force sensor or artificial vision, etc. In this study, the PC runs on the Windows XP operating system and two development environments have been used: Matlab and Microsoft Visual C++.

For the rapid development of the parallel robot controllers, Simulink schemes have been used in this application. Therefore, the Matlab Real-Time Workshop (RTW) and Real-Time Windows Target (RTWin) toolboxes have been used, which produce codes directly from Simulink models and automatically generate programs that can be run in environments like Linux, VxWorks, DOS and Windows. These toolboxes feature a rapid and direct path from system design to hardware implementations, seamless integration with Matlab and Simulink, a simple and easy interface, an open and extendable architecture, a fully configurable code generator etc.

Fig. 6 shows a Simulink/RTW scheme used for parallel robot control. Measurements for the three angular positions of the robot joints of the robot are obtained by means of the Advantech encoder card. It must be taken into account that, because they are incremental encoders, it is necessary to program the initial values when the system starts.

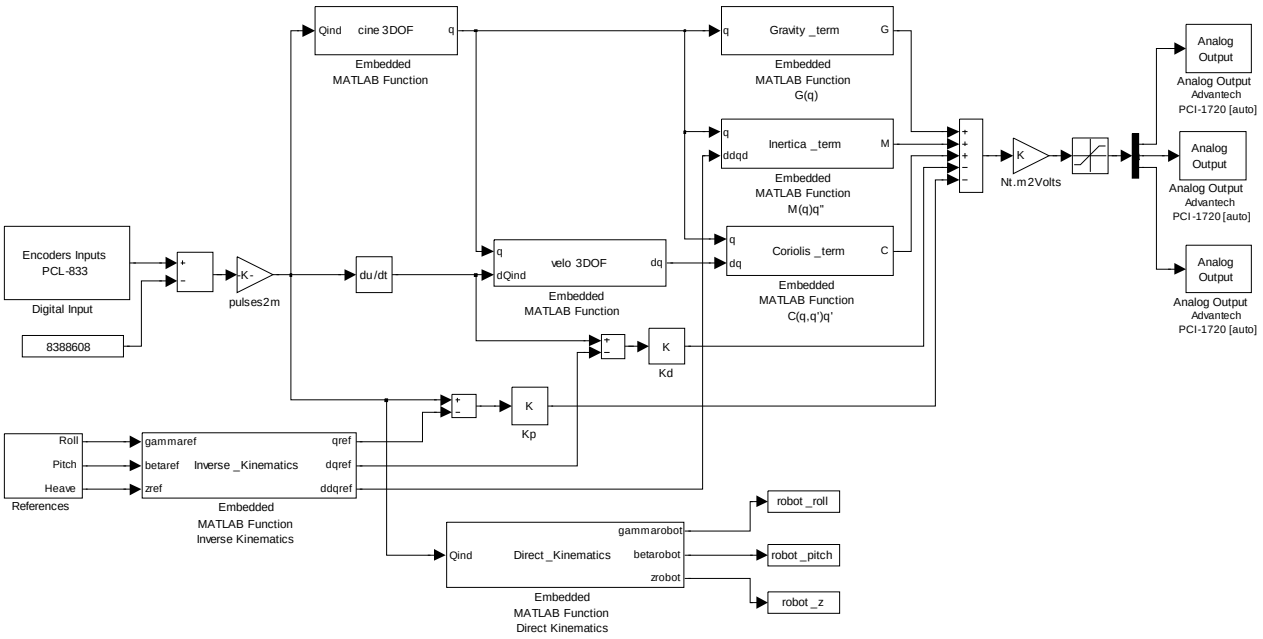


Figure 6: Paden passivity-based robot controller.

With the real robot position and motion references, the PC computes the necessary control actions for the robot joints. These control actions are sent to the power amplifiers of the control unit by means of digital/analog converters.

In this system, the Embedded Matlab Functions have been used to implement the control algorithm: the Paden passivity-based tracking control. This controller calculates the action controls by means of the robot dynamic equation (gravity, Coriolis and Inertial terms) and a PD controller. Four additional blocks are needed. The first one implements a routine for solving the forward position problem of the 3 DOF PM with a PRS configuration (cine3DOF embedded function). It obtains the 9 generalized coordinates that define the robot kinematics by means of the concepts in (Merlet, 2000), (García de Jalón and Bayo, 1994), and the nonlinear position problem is solved by using a Newton-Raphson algorithm.

The second one implements a routine for solving the velocity problem of a 3 DOF parallel robot (vel3DOF embedded function). The procedure used in this routine is based on the concepts in (García de Jalón and Bayo, 1994), and uses the Denavit-Hartenbert notation.

Because this scheme establishes the robot control in the joint space, another embedded function is necessary. The third one (*Inverse Kinematics*) implements the inverse kinematic problem of the parallel robot. This block uses the roll-pitch and heave references desired for the robot, providing the references for the three actuators of the robot joints ($q_1$, $q_2$ and $q_3$ positions, velocities and accelerations).Finally, in order to compare the references and the real robot variables in the task space, the fourth embedded function block calculates the direct kinematic problem. With it, the roll, pitch and heave of the parallel robot is calculated.

Once the good functionality of the controller has been analyzed and the parameters of the controllers (gain matrices and constants, for example) have been tuned with Matlab, the development of the final control application can be performed. In order to do this, in this study Microsoft Visual C++ (MSVC) has been used, which is a very well-known development environment. MSCV is a commercial, integrated development environment (IDE) product from Microsoft for the C, C++, and C++/CLI programming languages. It has tools for developing

and debugging the C++ code, a code especially written for Microsoft Windows API, DirectX API, and Microsoft .NET Framework. For access to the data acquisition cards, the Advantech *PC-LabCard Software Driver* has been used, which provides the required drivers for the Advantech cards (A/D and D/A conversions, encoders, digital input/output, counters/timers, etc.) and allows control of the card's functions using high-level languages. In addition, these drivers make programming easier because each function pulls its parameters from a common parameter table.

The C++ application implemented establishes robot control using the passivity-based and inverse dynamic control algorithms presented in Sections 4.1 and 4.2. In order to program these algorithms, the NAG Numerical Library has been used.

The sampling period used to establish control of the parallel robot is 10ms. Tests developed have proved that this period is long enough, as the mean controller execution time obtained (which includes encoder readings, controller calculations and digital/analog conversions) is less than 4ms.

## 5 RESULTS

Using the parallel robot and its open control systems, different control algorithms have been developed and tested. The control strategies solved the point-to-point and tracking problems. Fig. 7 shows the reference and the real parallel robot values of heave, roll and pitch. As can be seen, the robot response follows the references with a very low level of error.

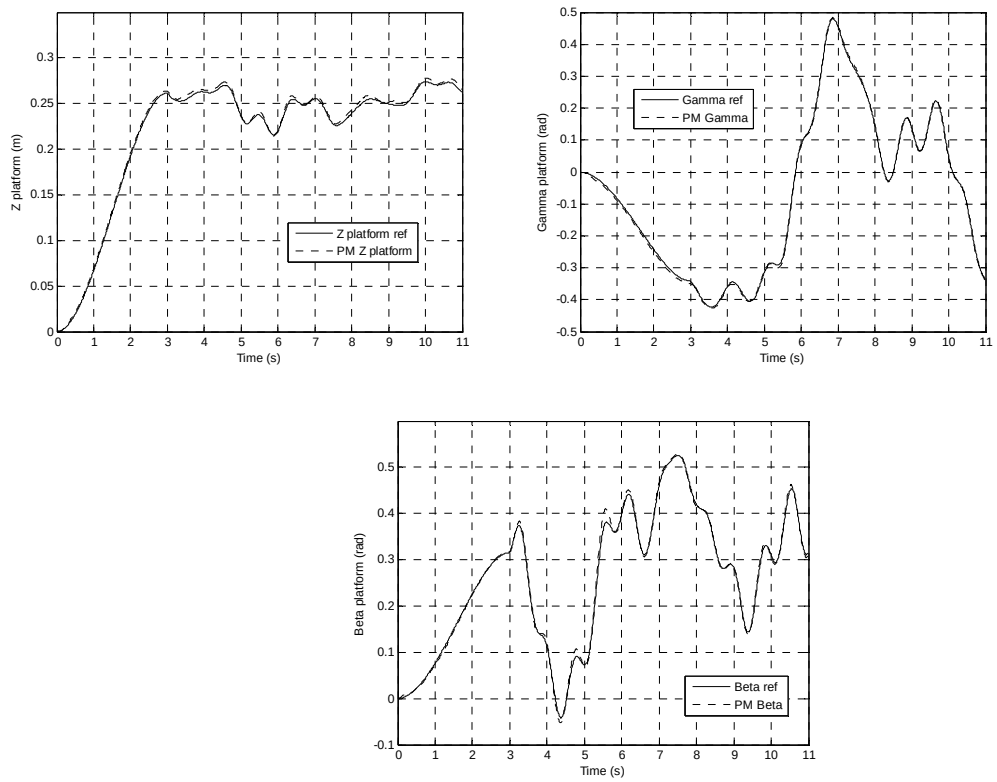


Figure 7: Heave, roll and pitch parallel robot response.

Figs. 8 and 9 show the joint space robot response. Fig. 8 shows the robot position ($q_1$, $q_6$ and $q_8$, in meters) and the action control (in volts) for the point-to-point control problem. The curves plotted in green are the robot references. The curves plotted in blue belong to the in-

verse dynamic controller. The black curves belong to the PID controller and the red curves belong to the PD with a gravity compensation controller.

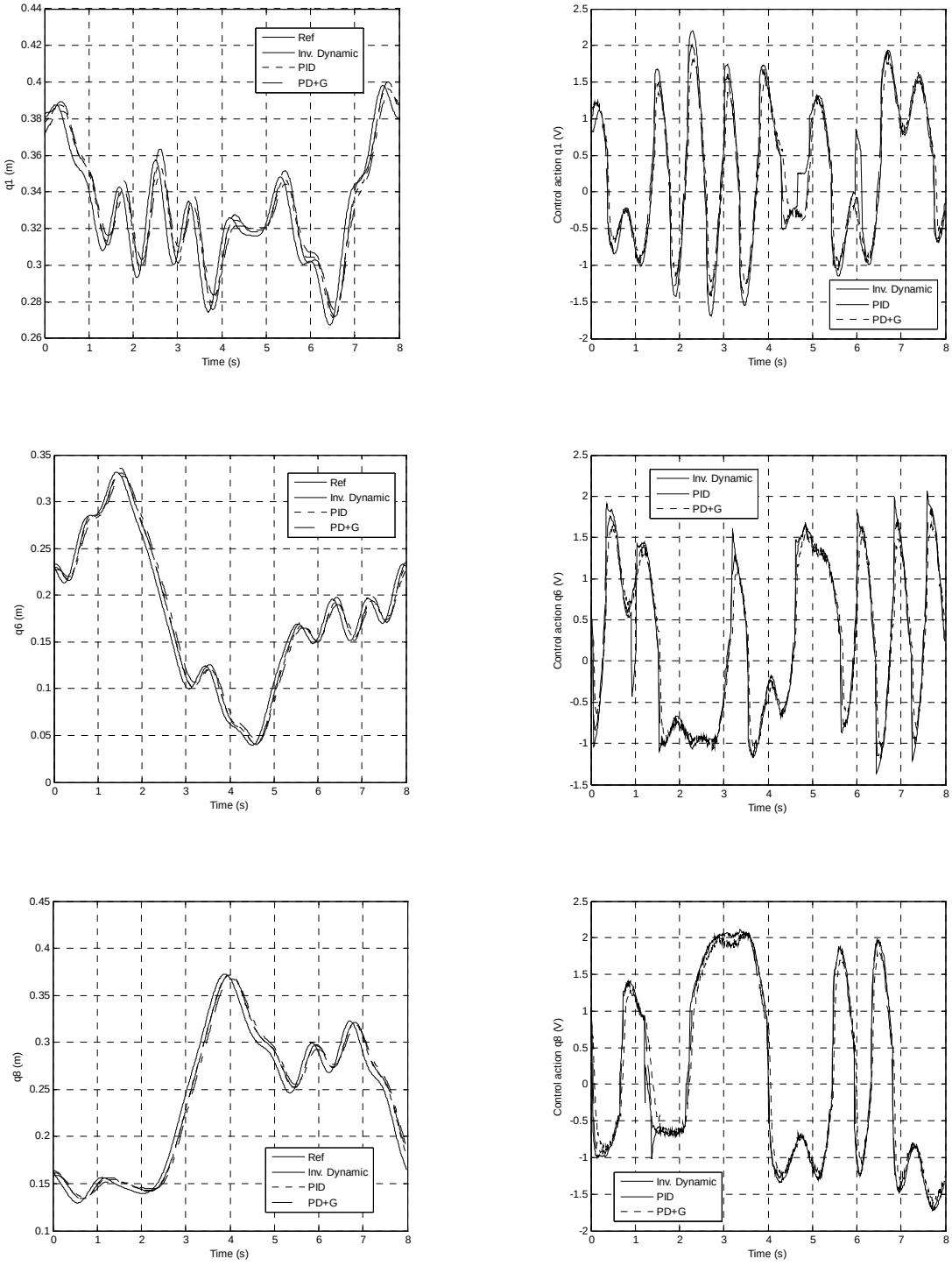


Figure 8: Parallel robot response for the point-to-point problem.

Fig. 9 shows the robot response for the tracking control problem. As in the last case, the references are plotted in green. The curves in blue belong to the inverse dynamic controller and the red curves belong to the Paden passivity-based controller.

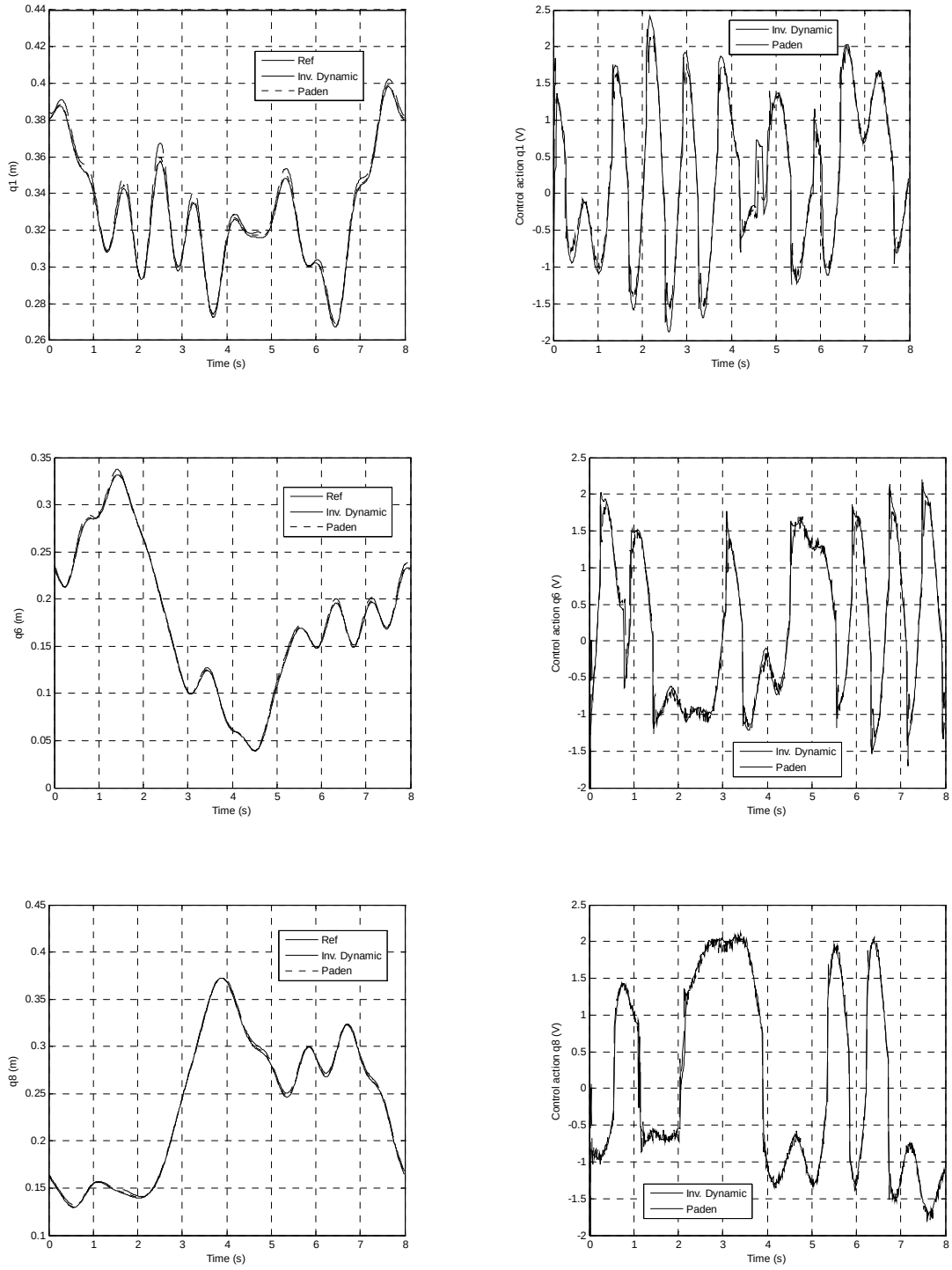


Figure 9: Parallel robot response for the tracking problem.

Table 5 shows the mean error and the mean square root error (RMS) between the references and the real positions of the parallel robot for the point-to-point problem.

| **Controller** | $\frac{\sum_i e_i}{n}$ | | | $\sqrt{\frac{\sum_i e_i^2}{n}}$ | | |
|---|---|---|---|---|---|---|
| | $q_1$ | $q_6$ | $q_8$ | $q_1$ | $q_6$ | $q_8$ |
| PD+G | -0.00111 | -0.00117 | -0.00108 | 0.01126 | 0.01325 | 0.01364 |
| PID | -0.00124 | -0.00046 | -0.00124 | 0.00856 | 0.00949 | 0.00965 |
| Inv. Dynamics | -0.00233 | -0.00142 | -0.00177 | 0.01016 | 0.00925 | 0.01030 |

Table 5: Robot position errors (mean and RMS) for point-to-point control.

Table 6 shows the error for the tracking problem.

| **Controller** | $\frac{\sum_i e_i}{n}$ | | | $\sqrt{\frac{\sum_i e_i^2}{n}}$ | | |
|---|---|---|---|---|---|---|
| | $q_1$ | $q_6$ | $q_8$ | $q_1$ | $q_6$ | $q_8$ |
| Inv. Dynamics | -0.00239 | -0.00145 | -0.00186 | 0.00343 | 0.00300 | 0.00234 |
| Passivity-based | -0.00061 | -0.00057 | -0.00059 | 0.00080 | 0.00099 | 0.00075 |

Table 6: Robot position errors (mean and RMS) for tracking control.

An additional experiment with the parallel robot was carried out. In this experiment, a sine function was used as a reference for the heave of the robot mobile platform. 10 different values of frequency of this sine reference were considered: from 1/1.3 sec. until 1/0.5 sec. For each of these frequencies, a point-to-point controller (PID) and a trajectory controller (Paden) were used. Fig. 10 shows the RMS error obtained with these controllers. It is clear that the point-to-point controller not only provides a worse error than the trajectory controller, but it also has a slope that is twice as big as the trajectory controller error slope.

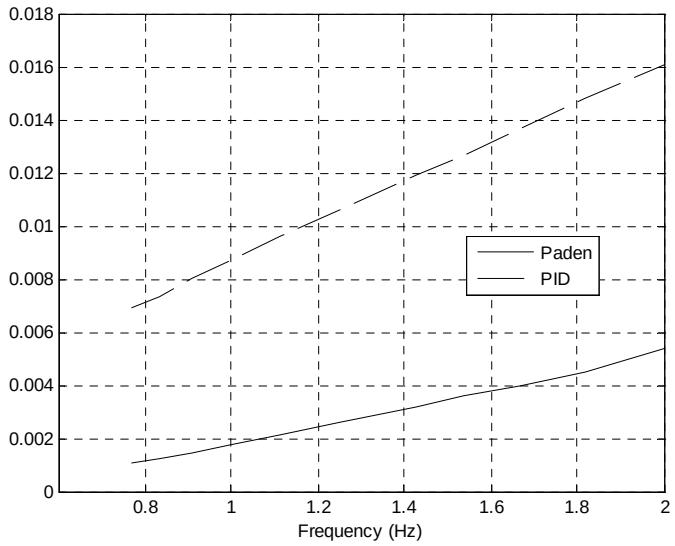


Figure 10: RMS error of point-to-point and trajectory controllers.

## 6 CONCLUSIONS

In this paper the mechatronic design, mechanical structure, electric actuators and control system of a low-cost 3-DOF PRS parallel manipulator have been fully developed. Open control architecture has been developed for this robot, and two control schemes have been proposed: Passivity-based control and Inverse dynamic control. The control algorithm considers point-to-point control or tracking control. Both direct and inverse kinematic equations for the PM have been obtained for application to the control system. When the controller considers the system dynamics, dynamic parameters obtained through an identification process have been used. The control schemes have been tested on a virtual robot and on the actual prototype. Different results showing the tracking accuracy of the proposed controllers are included.

## ACKNOWLEDGEMENTS

The authors wish to express their gratitude to the Plan Nacional de I+D, Comisión Interministerial de Ciencia y Tecnología (FEDER-CICYT) for the partial financing of this study under the projects DPI2009-13830-C02-01 and DPI2010-20814-C02-(01, 02). This work was also supported in part by the CDCHT-ULA Grant I-1286-11-02-B.

## REFERENCES

Awtar, S., Bernard, C., Boklund, N., Master, A., Ueda, D., Craig, K. (2002). Mechatronic design of a ball-on-plate balancing system. *Mechatronics*, vol. 12, n. 2, pp.217-228.

Carretero, J.A., Podhorodeski, R.O., Nahon, M.A. (2000). Kinematic analysis and optimization of a new three degree-of-freedom spatial parallel manipulator. *Journal of Mechanical Design,* vol. 122, nº1, 17-24.

Chablat, D., Wenger, P. (2003). Architecture optimization of a 3-DOF translational parallel mechanism for machining applications, the Orthoglide. *IEEE Transactions on Robotics and Automation,* vol. 19, nº 3, 403-410.

Clavel, R. (1988). DELTA, a fast robot with parallel geometry. *Proceedings of 18th International Symposium on Industrial Robot*, Lausanne, April, 91-100.

Díaz-Rodriguez, M., Mata, V., Farhat, N., Provenzano, S. (2008) Identifiability of the dynamic parameters of a class of parallel robots in the presence of measurement noise and modeling discrepancy. *Mechanics Based Design of Structures and Machines*, vol.36 , no. 4, pp. 478-498.

Díaz-Rodriguez, M., Mata, V., Valera, A., Page, A. (2010). A methodology for dynamic parameters identification of 3-DOF parallel robots in terms of relevant parameters. *Mechanism and Machine Theory*, vol. 45, no. 9, pp. 1337-1356.

García de Jalón, J., Bayo, E. (1994). *Kinematic and Dynamic Simulation of Multibody Systems: The Real-Time challenge.* Springer-Verlag, New-York.

Gough, V.E., Whitehall, S.G. (1962). Universal tire test machine. *Proceedings of 9th International Technical Congress FISITA, 117-135.*

Kim, H. S., Tsai, L.W. (2003). Kinematic Synthesis of a Spatial 3-RPS Parallel Manipulator. *Journal of Mechanical Design,* vol. 125, nº 1, 92-97.

Lee, K.M., Shah, D. (1988). Kinematic Analysis of a Three-Degrees-of-Freedom In-Parallel Actuated Manipulator. *IEEE Journal of robotics and automation*, vol.4, nº3, 354-360.

Li, Y., Xu, Q. (2007). Design and Development of a Medical Parallel Robot for Cardiopulmonary Resuscitation. *IEEE/ASME Transaction on Mechatronics*, vol. 12, nº 3, 265–273.

Merlet, J.P. (2000). *Parallel Robots*. Kluwer, London, U.K.

Merlet, J.P. (2002). Optimal design for the micro parallel robot MIPS. *Proceedings IEEE International Conference on Robotics and Automation*, 1149-1154.

Ortega, R., Spong, M. (1989). Adaptive Motion Control of Rigid Robots: a Tutorial, *Automatica*, vol. 25, pp. 877-888.

Paccot, F., Andreff, N., Martinet, P. (2009). A review on the dynamic control of parallel kinematic machines: Theory and Experiments. *The international Journal of Robotics Research,* vol. 28, nº 3, 395-416.

Rosillo, N., Valera, A., Benimeli, F., Mata, V., Valero, F. (2011). Real-time solving of Dynamic Problem in Industrial Robots, Industrial Robot, vol. 38, n. 2, pp. 119-129.

Stewart, D. A. (1965). A platform with 6 degree of freedom. *Proceedings of the Institution of mechanical engineers*. Part 1 vol. 15, 371-386.

Syrseloudis, C. E., Emiris, I. Z. (2008). A parallel robot for ankle rehabilitation-evaluation and its design specifications. *Proceeding of 8th IEEE International Conference on BioInformatics and BioEngineering*, 2008. Athens, 1-6, October.

Tsai, L. W. (1999). *Robot Analysis: The Mechanics of Serial and Parallel Manipulator*. Wiley Interscience, Canada.