Handbook of Control Science and Engineering

Marques Vang

Volume I

Handbook of Control Science and Engineering

Volume I

Edited by **Marques Vang**

CLANRYE INTERNATIONAL
New Jersey

Published by Clanrye International,
55 Van Reypen Street,
Jersey City, NJ 07306, USA
www.clanryeinternational.com

Handbook of Control Science and Engineering: Volume I
Edited by Marques Vang

International Standard Book Number: 978-1-63240-264-6 (Hardback)

Printed in China.

Handbook of Control Science and Engineering

Volume I

Preface

Control science or control systems or control engineering is a field of study and engineering science that integrates automata theory and control theory along with the system designing techniques to build systems with the desired functionality. These systems are designed for self-automation, and the sensors are used to initiate input processing, the measurement converted after the processing of input are further used to initiate output measurement and output controls to be achieved. When a designed system is used without human involvement, they are called automatic control systems. The basic approach to build such systems is mathematical modeling.

Control engineering is playing a large role in designing of control systems, and examples of these control systems can vary from microwave oven to high impact submarines. For such designing, the inputs, outputs, and other important functional components use mathematical modeling to develop controllers for complex systems, and then come the most important task of integrating these controllers with the physical systems using available technological tools. These systems can be of various ranges from mechanical to biological or even financial accounting, they all use control theory in time domains as well as frequency domains, depending on the designing and functional problem.

I especially wish to acknowledge the contributing authors, without whom a work of this magnitude would clearly not be realizable. We thank them for allocating much of their very scarce time to this project. Not only do I appreciate their participation but also their adherence as a group to the time parameters set for this publication. I also thank my publisher for considering me for this project and giving me this incredible opportunity.

Editor

Contents

Position Control of a 3-CPU Spherical Parallel Manipulator

Massimo Callegari,[1] Luca Carbonari,[1] Giacomo Palmieri,[2] Matteo-Claudio Palpacelli,[1] and Donatello Tina[1]

[1] *Department of Industrial Engineering & Mathematical Sciences, Polytechnic University of Marche, 60131 Ancona, Italy*
[2] *e-Campus University, Faculty of Engineering, 22060 Novedrate, Italy*

Correspondence should be addressed to Massimo Callegari; m.callegari@univpm.it

Academic Editor: Sabri Cetinkunt

The paper presents the first experimental results on the control of a prototypal robot designed for the orientation of parts or tools. The innovative machine is a spherical parallel manipulator actuated by 3 linear motors; several position control schemes have been tested and compared with the final aim of designing an interaction controller. The relative simplicity of machine kinematics allowed to test algorithms requiring the closed-loop evaluation of both inverse and direct kinematics; the compensation of gravitational terms has been experimented as well.

1. Introduction

Parallel kinematics machines, PKMs, are known to be characterized by many advantages like a lightweight construction and a high stiffness but also present some drawbacks, like the limited workspace, the great number of joints of the mechanical structure, and the complex kinematics, especially for 6-dof machines [1]. Therefore the A.'s proposed to decompose full-mobility operations into elemental subtasks, to be performed by separate reduced mobility machines, similarly to what is already done in conventional machining operations. They envisaged the architecture of a mechatronic system where two parallel robots cooperate in order to perform complex assembly tasks. The kinematics of both machines is based upon the same 3-CPU topology but the joints are differently assembled so as to obtain a translating parallel machine (TPM) with one mechanism and a spherical parallel machine (SPM) with the other.

This solution, at the cost of a more sophisticated controller, would lead to the design of simpler machines that could be used also stand-alone for 3-dof tasks and would increase the modularity and reconfigurability of the robotized industrial process. The two robots are now available at the prototypal stage, and the present paper reports the first experiments on the motion control of the orienting device (SPM).

2. Robot's Architecture and Kinematics

2.1. Mechanical Architecture. Since the detailed description of machine's kinematics and prototype design has been provided already in Callegari et al. [2], hereby only the most relevant aspects are recalled.

The spherical parallel machine under study is made of three identical serial chains connecting the moving platform to the fixed base, as shown in Figure 1; each leg is composed by two links: the first one is connected to the frame by a cylindrical joint (C), while the second link is connected to the first one by a prismatic joint (P) and to the end-effector by a universal joint (U); for this reason its mechanical architecture is commonly called 3-CPU. A few *manufacturing conditions*, already investigated for a general pure rotational tripod by Karouia and Hervè [3], must be fulfilled in order to constraint the end-effector to a spherical motion:

(i) the axes of the cylindrical joints ($\mathbf{a}_i$, $i = 1, 2, 3$) are aligned along the x, y, z axes of the base frame and intersect at the center O of the spherical motion;

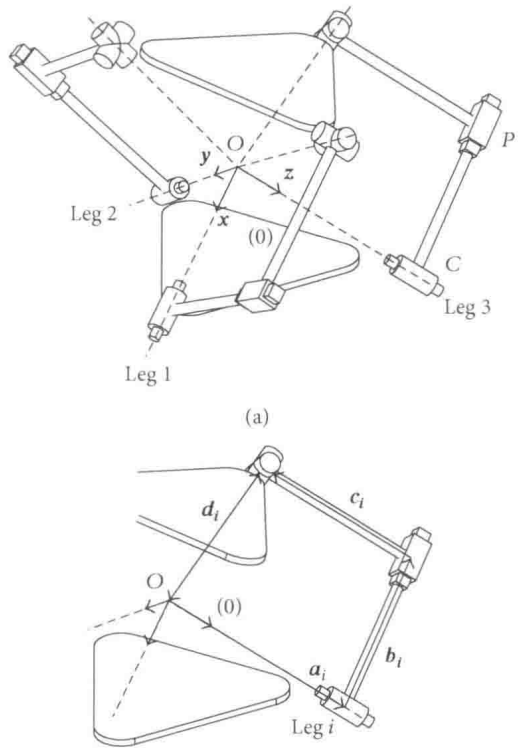

FIGURE 1: Kinematic schemes of the 3-CPU robot (a) and geometry of the legs (b).

(ii) the axis $\mathbf{b}_i$ of each prismatic pair is perpendicular to the axis of the respective cylindrical joint $\mathbf{a}_i$;

(iii) the first axis of each universal joint is perpendicular to the plane of the corresponding leg (plane identified by the axes $\mathbf{a}_i$ and $\mathbf{b}_i$);

(iv) the second axis of the 3 universal joints (resp., for the leg 1, 2, and 3) are aligned along the y_1, z_1, x_1 axes of a local frame centered in P (coincident with O) and attached to the mobile platform.

For a successful operation of the mechanism, one *mounting condition* must be satisfied too; assembly should be operated in such a way that the two frames $O(\mathbf{x_0}, \mathbf{y_0}, \mathbf{z_0})$ and $P(\mathbf{x_1}, \mathbf{y_1}, \mathbf{z_1})$ come to coincide when the robot is in its homing position. Such configuration is obtained when the three displacements a_i are equal to the length of the second link c and the displacements of the prismatic joints b_i are equal to the constant distance d. If the mounting conditions are verified, the points P and O remain fixed and coincident while the moving platform performs a spherical motion around them.

2.2. Kinematic Relations. The platform is actuated by driving the strokes of the 3 cylindrical joints; therefore joint space displacements are gathered into the following vector $\mathbf{q}$:

$$\mathbf{q} = \begin{bmatrix} a_1 \\ a_2 \\ a_3 \end{bmatrix}. \tag{1}$$

The position kinematics of the robot expresses the relation between the orientation of the mobile platform and the displacements of the actuators; the attitude of the machine in space is fully provided by the rotation matrix ${}^{O}_{P}\mathbf{R}$, that can also be conveniently expressed as a composition of elemental rotations. In the development of robot's kinematics, the following Cardan angles set is used:

$$\begin{aligned} {}^{O}_{P}\mathbf{R}(\alpha, \beta, \gamma) &= \mathbf{R}_x(\alpha)\,\mathbf{R}_y(\beta)\,\mathbf{R}_z(\gamma) \\ &= \begin{bmatrix} c\beta c\gamma & -c\beta s\gamma & s\beta \\ s\alpha s\beta c\gamma + c\alpha s\gamma & -s\alpha s\beta s\gamma + c\alpha c\gamma & -s\alpha c\beta \\ -c\alpha s\beta c\gamma + s\alpha s\gamma & c\alpha s\beta s\gamma + s\alpha c\gamma & c\alpha c\beta \end{bmatrix}. \end{aligned} \tag{2}$$

The position kinematics of the robot is simply expressed by

$$\begin{aligned} r_{12} &= -c\beta s\gamma = \frac{c - a_1}{d}, \\ r_{23} &= -s\alpha c\beta = \frac{c - a_2}{d}, \\ r_{31} &= -c\alpha s\beta c\gamma + s\alpha s\gamma = \frac{c - a_3}{d}, \end{aligned} \tag{3}$$

where r_{ij} is the element at the ith row and jth column of rotation matrix ${}^{O}_{P}\mathbf{R}$. The solution of the *direct position kinematics* (DPK) problem requires the computation of the rotation matrix ${}^{O}_{P}\mathbf{R}$ as a function of internal coordinates $\mathbf{q}$, which has been solved already by Carbonari et al. [4]. According to Innocenti and Parenti-Castelli [5], a maximum number of 8 different configurations can be worked out; however, a single feasible solution is found when the real workspace of the robot is considered; that is, the actual mobility of the joints is taken into consideration. *Inverse position kinematic* (IPK) problem admits just one solution and it is trivially solved by working out joint displacements $\mathbf{q}$ in (3).

Turning to *differential kinematics*, the expression of the analytic Jacobian $\mathbf{J}_A$ is immediately obtained as a function of the Cardan angles and their rates:

$$\begin{bmatrix} \dot{a}_1 \\ \dot{a}_2 \\ \dot{a}_3 \end{bmatrix} = \mathbf{J}_A \begin{bmatrix} \dot{\alpha} \\ \dot{\beta} \\ \dot{\gamma} \end{bmatrix},$$

$$\mathbf{J}_A = d \begin{bmatrix} 0 & -s\beta s\gamma & c\beta c\gamma \\ c\alpha c\beta & -s\alpha s\beta & 0 \\ -s\alpha s\beta c\gamma - c\alpha s\gamma & c\alpha c\beta c\gamma & -c\alpha s\beta s\gamma - s\alpha c\gamma \end{bmatrix}. \tag{4}$$

By taking into account the relation between the derivatives of the Cardan angles and the angular velocity $\boldsymbol{\omega}$:

$$\begin{bmatrix} \omega_x \\ \omega_y \\ \omega_z \end{bmatrix} = \begin{bmatrix} 1 & 0 & s\beta \\ 0 & c\alpha & -s\alpha c\beta \\ 0 & s\alpha & c\alpha c\beta \end{bmatrix} \begin{bmatrix} \dot{\alpha} \\ \dot{\beta} \\ \dot{\gamma} \end{bmatrix} = \mathbf{T} \begin{bmatrix} \dot{\alpha} \\ \dot{\beta} \\ \dot{\gamma} \end{bmatrix}, \tag{5}$$

the geometric Jacobian $\mathbf{J}_G$ is easily obtained too:

$$\begin{bmatrix} \dot{a}_1 \\ \dot{a}_2 \\ \dot{a}_3 \end{bmatrix} = \mathbf{J}_A \mathbf{T}^{-1} \begin{bmatrix} \omega_x \\ \omega_y \\ \omega_z \end{bmatrix} = \mathbf{J}_G \begin{bmatrix} \omega_x \\ \omega_y \\ \omega_z \end{bmatrix}, \tag{6}$$

with

$$\mathbf{J}_G = d \begin{bmatrix} 0 & -c\alpha s\beta s\gamma - s\alpha c\gamma & c\alpha c\gamma - s\alpha s\beta s\gamma \\ c\alpha c\beta & 0 & -s\beta \\ -s\alpha s\beta c\gamma - c\alpha s\gamma & c\beta c\gamma & 0 \end{bmatrix}. \tag{7}$$

2.3. User Frames. In order to better define the tasks to be commanded and visualize the obtained results, it is useful to choose a different set of reference frames, as shown in Figure 2. The fixed frame $O^*(\mathbf{x}_0^*, \mathbf{y}_0^*, \mathbf{z}_0^*)$ is defined as follows:

(i) the origin is located at the center of the moving platform when it assumes its initial configuration;

(ii) the $\mathbf{z}_0^*$ axis is aligned to the vector $\mathbf{g}$ of gravity acceleration;

(iii) the $\mathbf{x}_0^*$ axis lies on the upper plane of the platform and points toward the axis $\mathbf{a}_1$ of the cylindrical joint of the first leg;

(iv) the $\mathbf{y}_0^*$ axis is placed according to the right-hand rule.

The mobile frame $P^*(\mathbf{x}_1^*, \mathbf{y}_1^*, \mathbf{z}_1^*)$ is coincident with the fixed frame $O^*(\mathbf{x}_0^*, \mathbf{y}_0^*, \mathbf{z}_0^*)$ when the platform is in its initial configuration. Of course, since the frames are not placed at the center of the spherical motion, the two origins O^* and P^* will be coincident only in the home position.

Once the location of the new frame O^* has been defined by means of the ${}^{O}_{O^*}\mathbf{R}$ rotation matrix, the orientation of the mobile platform can be described in the new frames by

$${}^{O^*}_{P^*}\mathbf{R} = {}^{O}_{O^*}\mathbf{R}^T \, {}^{O}_{P}\mathbf{R} \, {}^{O}_{O^*}\mathbf{R}, \tag{8}$$

where it has been used the identity ${}^{O}_{O^*}\mathbf{R} = {}^{P}_{P^*}\mathbf{R}$. Of course, being the mobile and fixed frames modified, also the Cardan angles $\varphi_x, \varphi_y, \varphi_z$ that yield the rotation matrix ${}^{O^*}_{P^*}\mathbf{R}$ are different from the previously described set (α, β, γ):

$${}^{O^*}_{P^*}\mathbf{R}(\varphi_x, \varphi_y, \varphi_z) = \mathbf{R}_{x^*}(\varphi_x)\, \mathbf{R}_{y^*}(\varphi_y)\, \mathbf{R}_{z^*}(\varphi_z). \tag{9}$$

Henceforth these angles are used to describe the orientation of the manipulator and to assign the tasks of the mobile platform; since they are assumed as external coordinates for the computation of the differential kinematics, the analytic and the geometric Jacobians are worked out again as previously described, providing similar but more complex relations.

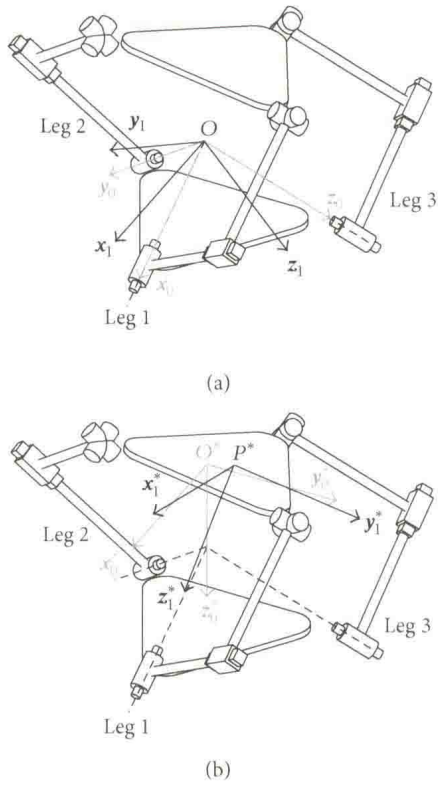

FIGURE 2: User-defined task frames.

3. Control Algorithms

3.1. Overview. Several kinds of control schemes have been tried on the 3-CPU SPM, with the immediate goal of testing the prototypal robot but aiming at the final design of an efficient co-operative environment for mechanical assembly. In the end, 3 different algorithms have been studied in simulation and then experimentally tested:

(i) a conventional joint resolved PID [6];

(ii) a joint resolved PID with the compensation of gravity forces [7];

(iii) a task-space PID with gravity compensation [8].

In all control schemes, the PID loop has been computed as usual in the following way:

$$\mathbf{u}(t) = \mathbf{K}_P \left(\mathbf{e}(t) + \frac{1}{\mathbf{T}_i} \int_0^t \mathbf{e}(\tau)\, d\tau + \mathbf{T}_D \frac{d}{dt} \mathbf{e}(t) \right), \tag{10}$$

with $\mathbf{u}(t)$ control action and $\mathbf{e}(t)$ input position error; $\mathbf{K}_P$, $\mathbf{T}_I$, and $\mathbf{T}_D$ are, respectively, the proportional gain, integral time, and derivative time matrices of the PID regulator.

3.2. Joint Resolved PID. First, a conventional joint resolved PID has been considered; see Figure 3. The error signal $\tilde{\mathbf{a}}$ is computed in the joint space as a difference between the desired position of the sliders $\mathbf{a}_D$ and their actual values $\mathbf{a}$:

$$\tilde{\mathbf{a}} = \mathbf{a}_D - \mathbf{a}. \tag{11}$$

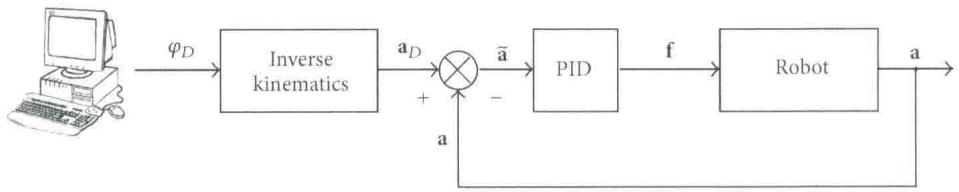

FIGURE 3: Joint space PID controller.

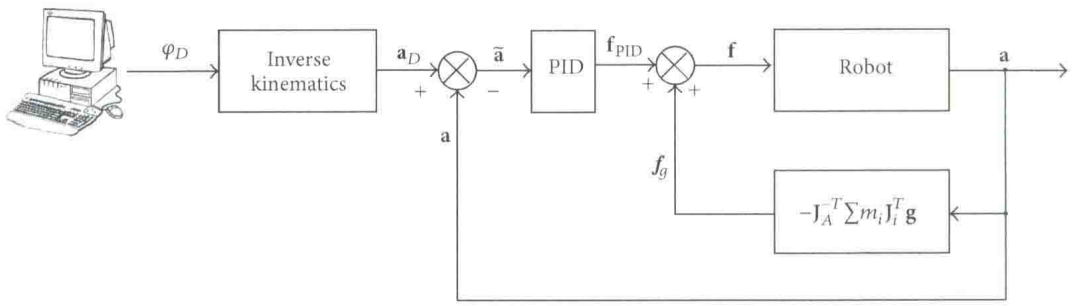

FIGURE 4: Joint space PID controller with gravity compensation.

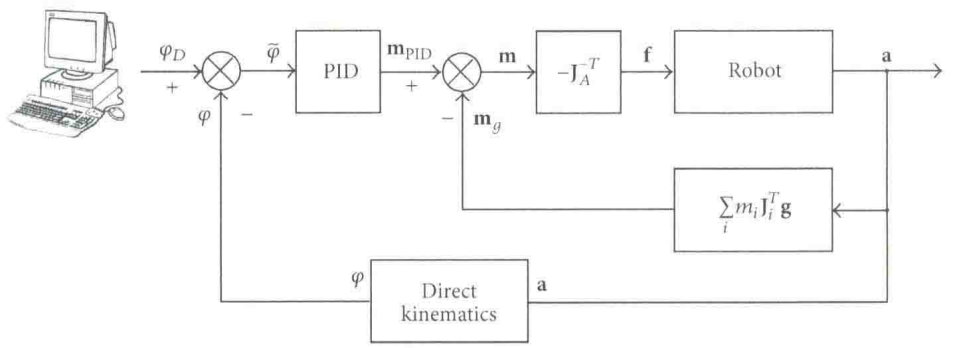

FIGURE 5: Task space PID controller with gravity compensation.

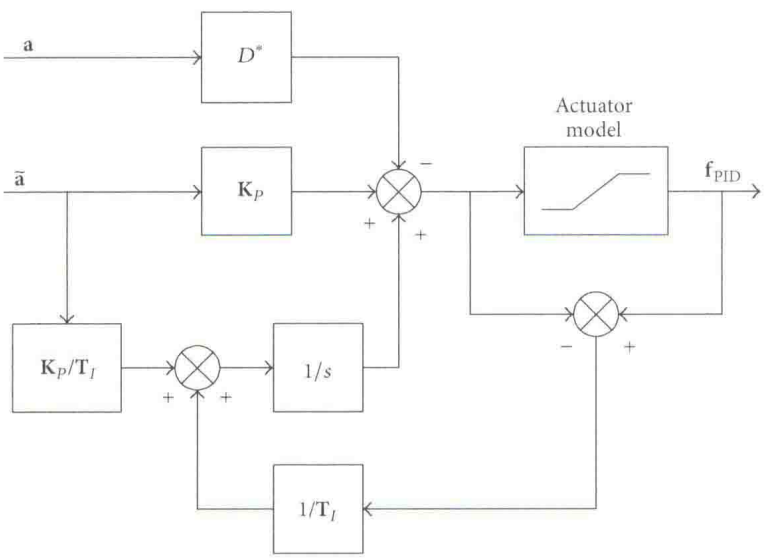

FIGURE 6: Anti-windup modified PID.

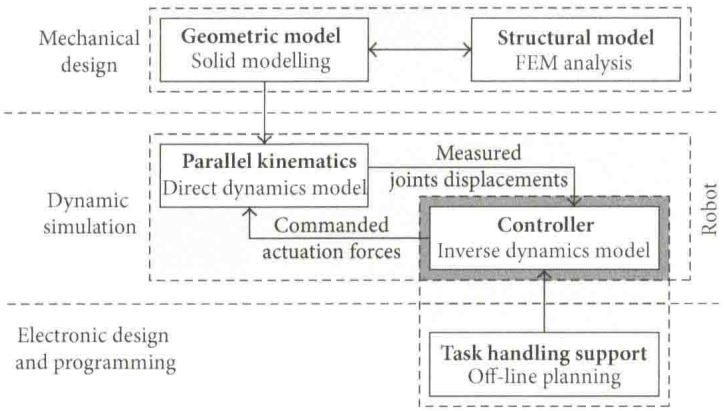

FIGURE 7: Integrated virtual prototyping environment for PKM's analysis and design.

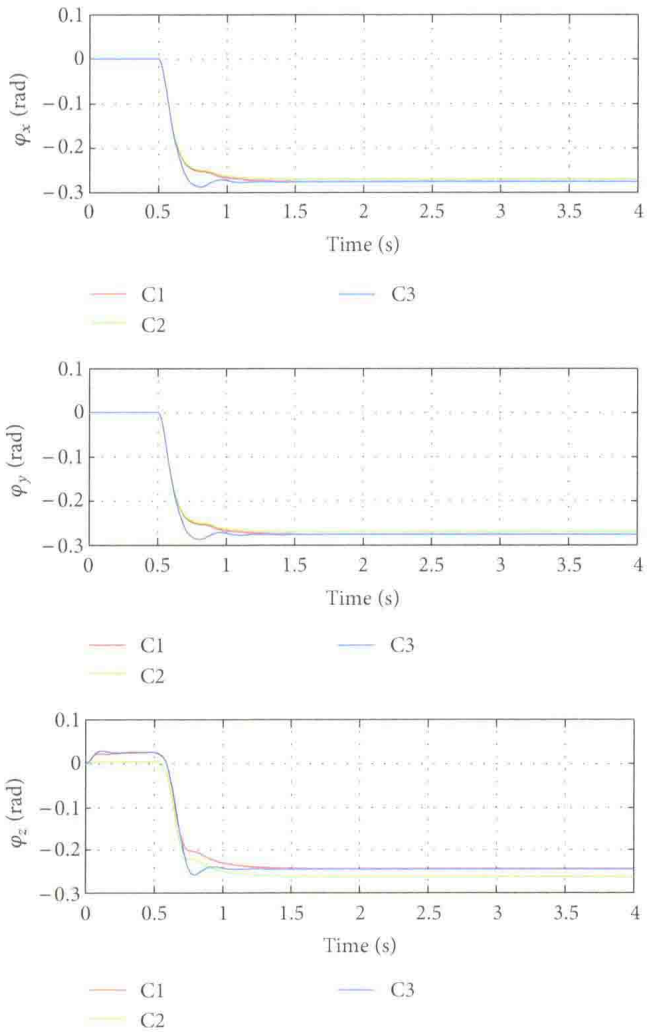

FIGURE 8: Response to step input: task space trajectories.

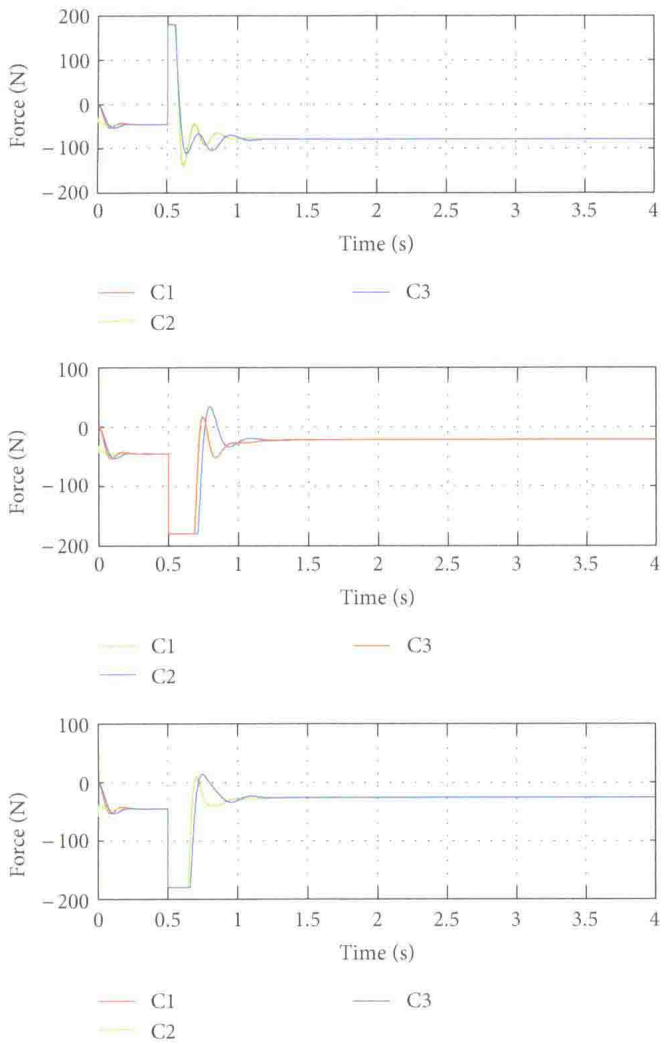

FIGURE 9: Response to step input: control efforts.

Since planning is programmed in the orientation space by assigning the desired configuration of the robot $\boldsymbol{\varphi}_D$, the corresponding position of the actuated joints is computed by means of inverse kinematics relations. The actuation effort of the motors is computed as

$$\mathbf{f} = \mathbf{K}_P \left[\tilde{\mathbf{a}} + \frac{1}{\mathbf{T}_i} \int \tilde{\mathbf{a}}\, dt + \mathbf{T}_D \frac{d\tilde{\mathbf{a}}}{dt} \right], \tag{12}$$

where the diagonal matrices $\mathbf{K}_P$, $\mathbf{T}_I$, and $\mathbf{T}_D$ have been introduced already in the previous section.

3.3. Joint Resolved PID with Gravity Compensation. In robotics the effects of gravitational field are often much more relevant than the other dynamics terms, at least for the small/moderate velocities attained during assembly tasks; such terms can be easily evaluated by means of the virtual work principle, as worked out in Callegari et al. [2]. Thus, a compensation term can be introduced by adding the force vector:

$$\mathbf{f}_g = -\mathbf{J}_A^{-T} \sum_i m_i \mathbf{J}_i^T \mathbf{g}, \tag{13}$$

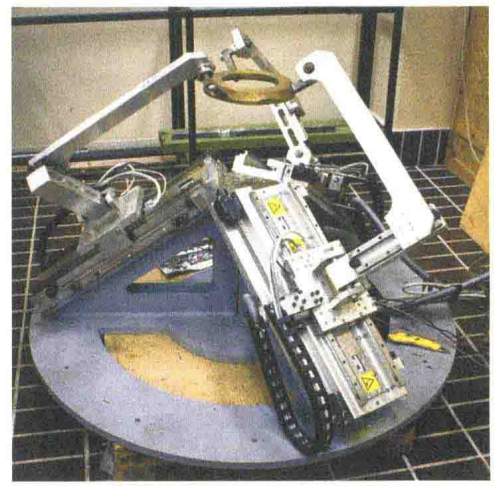

FIGURE 10: The prototype of the spherical parallel machine.

where $\mathbf{J}_A$ is the analytic Jacobian matrix, m_i is the mass of the ith member, $\mathbf{J}_i$ is the Jacobian that links the velocity of the centre of gravity of the ith member to the vector $\dot{\mathbf{a}}$, and $\mathbf{g}$ is the gravity acceleration. The resulting control scheme is shown in Figure 4.

3.4. Task Space PID with Gravity Compensation. The third control scheme that has been taken into consideration is a task space PID, with the compensation of the gravitational terms; see Figure 5:

$$\mathbf{f} = \mathbf{J}_A^{-T} \left(\mathbf{K}_P' \left[\tilde{\boldsymbol{\varphi}} + \frac{1}{\mathbf{T}_I'} \int \tilde{\boldsymbol{\varphi}}\, dt + \mathbf{T}_D' \frac{d\tilde{\boldsymbol{\varphi}}}{dt} \right] - \sum_i m_i \mathbf{J}_i^T \mathbf{g} \right), \tag{14}$$

where $\tilde{\boldsymbol{\varphi}}$ is the error signal in the task space and the PID gains $\mathbf{K}_P'$, $\mathbf{T}_I'$, and $\mathbf{T}_D'$ are diagonal matrices once again. This algorithm is computationally more expensive than the previous one, since it requires the evaluation of direct kinematics that for PKMs is more complex than inverse kinematics; on the other hand, it could prove useful, for example, in vision assisted assembly tasks with position-based controls, as already experimented on the 3-CPU translating parallel machine by Palmieri et al. [9].

3.5. Implementation in Real-Time Controller. During the implementation of algorithms (12)–(14) on the real-time controller, it was took into consideration the sensitiveness to noise of differentiation. Considering the Laplace transform of (10), the mentioned problem has been numerically mitigated by substituting the classic derivative term $K_P T_D s$ with the following derivative operator:

$$D^*(s) = \frac{K_P T_D s}{1 + s T_D / N}, \tag{15}$$

where N has been chosen equal to 10. Another problem in the implementation of PID controllers over a real-time system is the windup effect. This phenomenon is due to the integral action which saturates the actuators output.

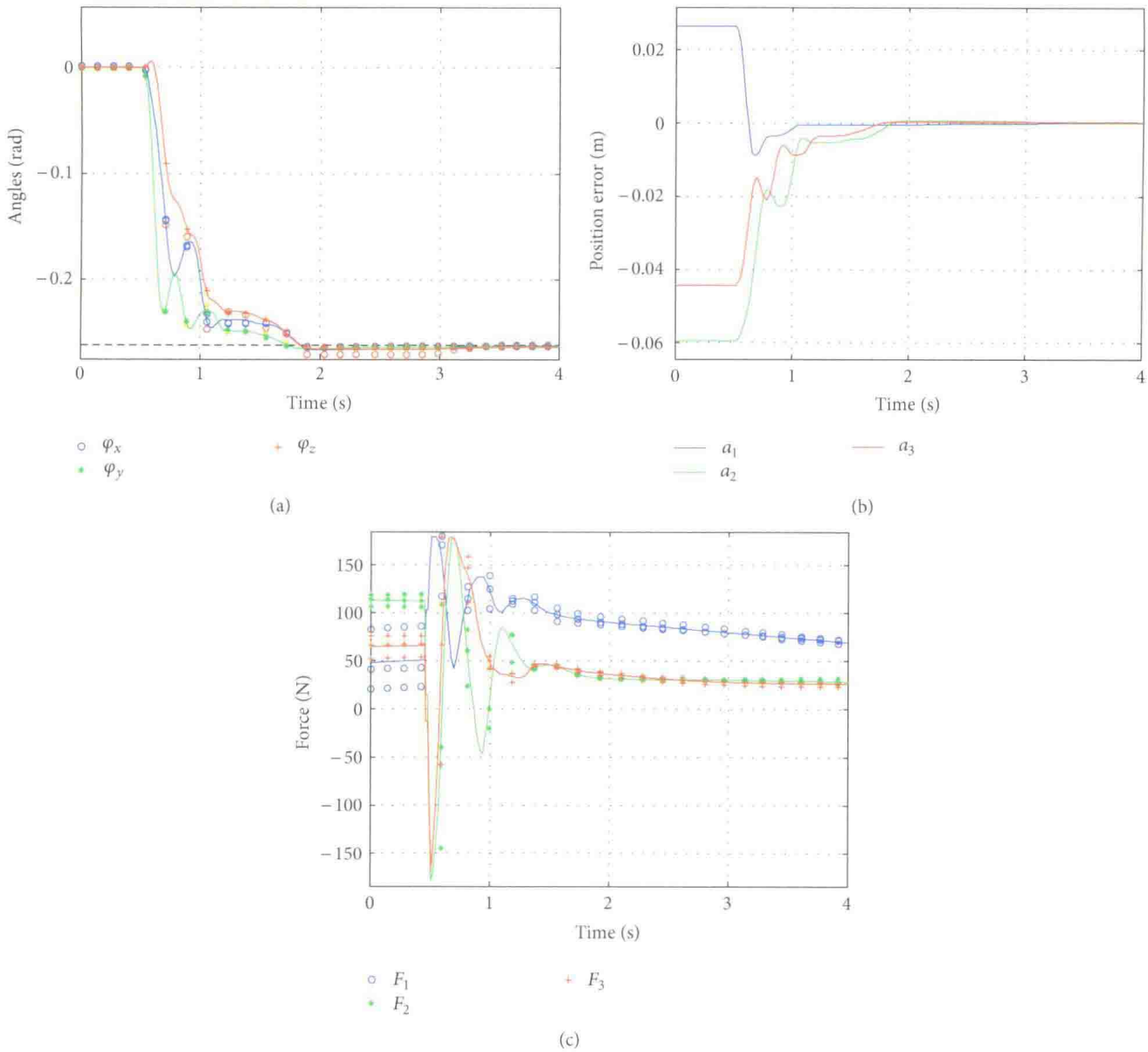

FIGURE 11: Joint-space PID controller: platform's trajectory in task-space (a), joint space errors (b), and motors thrusts (c).

Figure 6 shows a modified scheme of the PID control which implements a typical anti-windup strategy; the model of the actuator, mentioned in the scheme, was easily obtained after identification of the motor mechanical and electrical parameters summarized in Table 2.

4. Simulation Results

4.1. Simulation Environment. Figure 7 shows the virtual prototyping environment used at the Robotics Laboratory of the Polytechnic University of Marche for the design of automated and robotized systems, in particular for the design and virtual testing of parallel kinematic manipulators. The mechanical design is developed through conventional CAD tools, which allow to easily define even the most complex geometries and also to perform, for example, by means of FEM modules, the needed structural analyses; the interface with a multibody code allows to perform closed-loop dynamic analyses, with different levels of difficulty according to the associativity of the used programs. In this case, the LMS Virtual. Lab Motion package has been used, which is able to handle conveniently also complex situations like, for instance, the occurrence of an impact. The multibody package receives in input from the controller the actuation torques and integrates the equation of direct dynamics, providing in output the state variables assumed to be measured. The control system, which is implemented in the Matlab/Simulink environment, computes the control actions by taking into account the commanded task and sometimes, just like the present case, by also exploiting the complete or partial knowledge of robot's dynamics (inverse dynamics model). If the task is constrained by the contact with the environment, like is usually the case for assembly, the contact forces can be evaluated too, to set up more efficient force control schemes. It is noted that, by

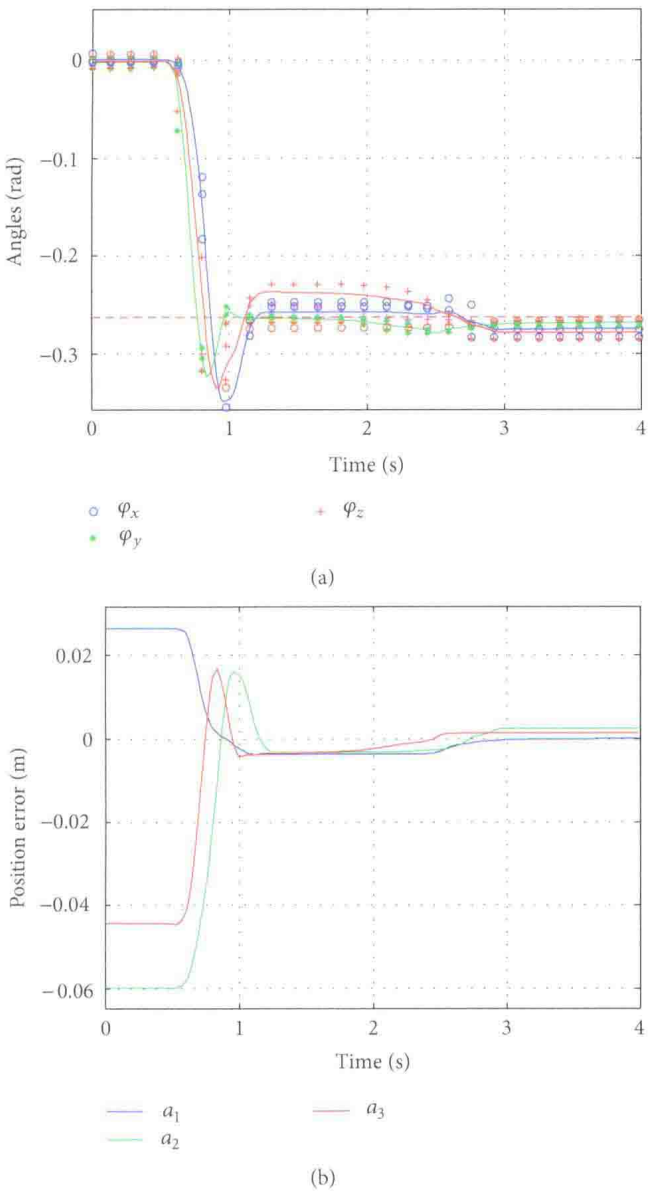

FIGURE 12: Joint-space PID with gravity compensation: platform's trajectory in task-space (a) and joint space errors (b).

TABLE 1: PIDs gains.

	Joint space	Joint space with gravity compensation	Task space with gravity compensation
K_P [N/m]	25000	15000	1000
T_I [s]	200	100	100
T_D [s]	1	0.2	0.25

using the Real-Time Workshop package of the Matlab suite, the same code used during the simulations in the virtual prototyping environment has been directly ported to the real-time control hardware afterwards.

In this way, by means of the mentioned prototyping software, a model of the spherical robot has been made available for the design of the control system and for the

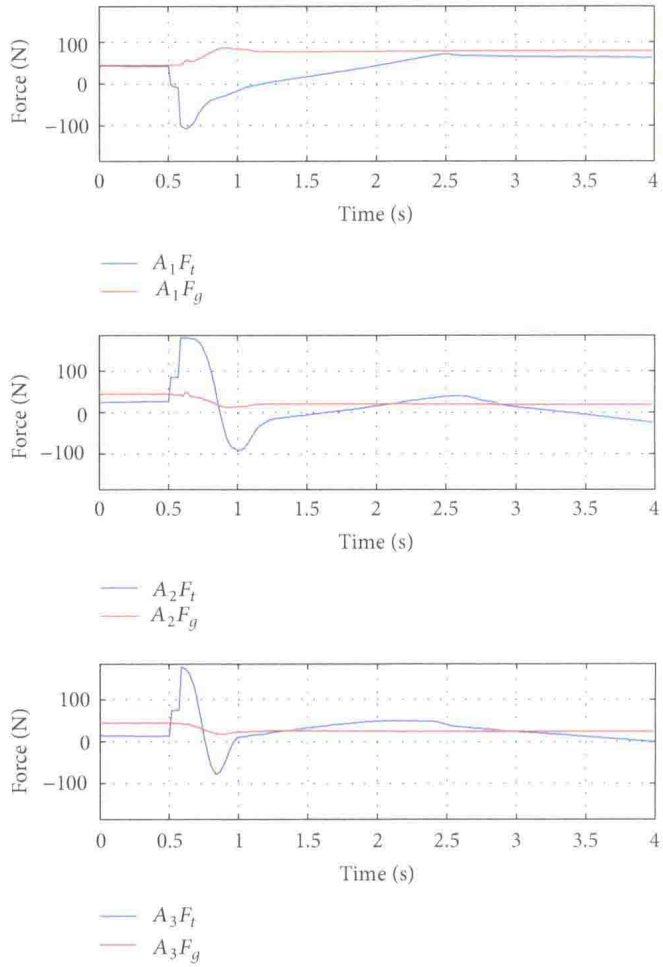

FIGURE 13: Joint-space PID with gravity compensation: F_t total force provided by the motors, F_g gravity component.

TABLE 2: Parameters of the linear drives.

Motors properties			
M_s	2.95	kg	Stator mass
K_t	58	N/A	Torque constant
I_n	3	A	Nominal supply current
T_n	184	N	Nominal thrust
v_n	6	m/s	Nominal speed

tuning of the PID's. Table 1 collects some control gains at the end of the tuning procedure, based on both simulation runs and experimental tests.

4.2. Simulation Analysis. A few test cases have been set up in simulation to evaluate the performances of the 3 PID controllers described in Section 3. The figures show the response of the system when the robot started at rest from