

Embedding Force Control into Industrial Robots

F. Caccavale¹, C. Natale², B. Siciliano³ and L. Villani³

¹ Dipartimento di Ingegneria e Fisica dell'Ambiente
Università degli Studi della Basilicata
Contrada Macchia Romana, 85100 Potenza, Italy

² Dipartimento di Ingegneria dell'Informazione
Seconda Università degli Studi di Napoli
Via Roma 29, 81031 Aversa, Italy

³ Dipartimento di Informatica e Sistemistica
Università degli Studi di Napoli Federico II
Via Claudio 21, 80125 Napoli, Italy
E-mail: {siciliano,lvillani}@unina.it

Accepted for publication on:
IEEE Robotics & Automation Magazine
Special issue on: *Industrial Robotics Applications &*
Industry-Academia Cooperation in Europe
New Trends and Perspectives
June 2004

Abstract

The leading industrial robotics community seems to be ready for the integration of force control capabilities with traditional motion controllers. This paper presents the guidelines for the effective realization of such integration, where the key knots from motion control to interaction control are highlighted. Two different interaction control strategies are introduced, namely impedance control and parallel force/position control. The resulting interaction control schemes are obtained from motion control schemes suitably modified by the closure of an outer force feedback loop. Finally, an industrial set up with open control architecture is considered as a paradigm for the implementation of interaction control.

1 Introduction

Research on robot force control has flourished in the past two decades. Such a wide interest is motivated by the general desire of providing robotic systems with enhanced sensory capabilities. Robots using force, touch, distance, visual feedback are expected to autonomously operate in unstructured environments other than the typical industrial shop floor.

When the robot executes a task involving interaction with the environment, the use of a purely motion control strategy turns out to be inadequate. The adoption of force feedback may avoid the rise of the contact forces and/or an unstable behavior during the interaction. On the other hand, since the early work on telemanipulation, the use of force feedback was conceived to assist the human operator in the remote handling of objects with a slave manipulator. More recently, cooperative robot systems have been developed where two or more manipulators (viz. the fingers of a dexterous robot hand) are to be controlled so as to limit the exchanged forces and avoid squeezing of a commonly held object.

Control of interaction between a robot manipulator and the environment is crucial for successful execution of a number of practical tasks where the robot end effector has to manipulate an object or perform some operation on a surface. Typical examples include polishing, deburring, machining or assembly. A complete classification of possible robot tasks is practically infeasible in view of the large variety of cases that may occur, nor would such a classification be really useful to find a general strategy to control interaction with environment.

During interaction, the environment sets constraints on the geometric paths that can be followed by the end effector. In such a case, the use of a purely motion control strategy for controlling interaction is a candidate to fail, as explained below.

Successful execution of an interaction task with the environment by using motion control could be obtained only if the task were accurately planned. This would in turn require an accurate model of both the robot manipulator (kinematics and dynamics) and the environment (geometry and mechanical features). Manipulator modelling can be known with enough precision, but a detailed description of the environment is often difficult to obtain.

To understand the importance of task planning accuracy, it is sufficient to observe that to perform a mechanical part mating with a positional approach, the relative positioning of the parts should be guaranteed with an accuracy of an order of magnitude greater than part mechanical tolerance. Once the absolute position of one part is exactly known, the manipulator should guide the motion of the other with the same accuracy.

In practice, due to the planning errors, the manipulator may attempt to violate the constraints imposed by the environment. This gives rise to a contact force and a deviation of the end effector from the desired trajectory. On the other hand, the control system reacts to reduce such deviation. This ultimately leads to a build-up of the contact force until saturation of the joint actuators is reached or breakage of the parts in contact occurs. The higher the environment stiffness and position control accuracy are, the easier a situation like the one just described can occur. This drawback can be overcome if a compliant behavior is ensured during the interaction. This can be achieved either in a passive fashion by interposing a suitable compliant mechanical device between the manipulator end effector and the environment, or in an active fashion by devising a suitable interaction control strategy.

The contact force is the quantity describing the state of interaction in the most complete fashion. Therefore, it is expected that enhanced performance can be achieved provided that force measurements are available. To this purpose, a force/torque sensor can be mounted on a robot manipulator, typically between the wrist and the end effector, and its readings shall be passed to the robot control unit via a suitable interface.

Robot force control has attracted a wide number of researchers in the past two decades. A state-of-the-art of the first decade is provided in [22], whereas the progress of the last decade is surveyed in [7] and in the monograph [20, 14].

Interaction control strategies can be grouped in two categories; those performing indirect force control and those performing direct force control. The main difference between the two categories is that the former achieve force control via motion control, without imposing a force set-point; the latter, instead, offer the possibility of controlling the contact force to a desired value.

To the first category belong compliance (or stiffness) control [15, 17] and impedance control [10], where the position error is related to the contact force through a mechanical stiffness or impedance of adjustable parameters. A robot manipulator under impedance control is described by an equivalent mass-spring-damper system with the contact force as input. The resulting impedance in the various task space directions is typically nonlinear and coupled. If a force/torque sensor is available, then force measurements can be used in the control law so as to achieve a linear and decoupled impedance.

If a detailed model of the environment is available, a widely adopted strategy belonging to the second category is the hybrid position/force control which aims at controlling position along the unconstrained task directions and force along the constrained task directions. A selection matrix acting on both desired and feedback quantities serves this purpose for typically planar contact surfaces [16], whereas the explicit constraint equations have to be taken into account for general curved contact surfaces [23, 11, 12].

In most practical situations, a detailed model of the environment is not available. In such a case, an effective strategy still in the second category is the motion/force control obtained by closing a force control loop around a motion control loop [8]. In order to embed the possibility of controlling motion along the unconstrained task directions, the desired end-effector motion can be input to the motion control loop. The resulting parallel control is composed of a force control action and a motion control action, where the former is designed so as to dominate the latter in order to ensure force control along the constrained task directions [4].

The present paper is aimed at presenting some interaction control schemes suitable for the implementation on industrial robot control units. In particular, the focus is on the impedance control and parallel control, which are conceived to manage the interaction with a more or less compliant environment without requiring an accurate model thereof. All the considered schemes are based on an inner motion control loop which can be the usual independent joint control of PID type, or an advanced model-based motion control law with superior tracking performance. The inner motion control loop is in charge of tracking the reference trajectory computed by the outer interaction control loop.

Special emphasis is given to controlling six-degree-of-freedom interaction tasks involving the end-effector position and orientation when a contact force and moment is applied.

Finally, an industrial setup with open control architecture is considered as a paradigm for the implementation of interaction control. Namely, the COMAU Smart-3S industrial robot with an open version of the C3G 9000 control unit is considered. The manipulator is equipped with a six-axis force/torque sensor ATI FT30-100 mounted at the arm's wrist.

2 Interaction control schemes

The realization of an interaction control scheme can be entrusted to the closure of an outer force control loop generating the reference input to the motion control loop which is available on every industrial robot. In detail, the inner motion control loop operates to ensure tracking of a reference trajectory, which is computed by a centralized outer loop acting on the force error.

A schematic of this control strategy is represented in Fig. 1. The outer interaction control is in charge of computing a suitable reference end-effector trajectory which ensures a compliant behavior of the manipulator when the end effector interacts with an external environment; tracking of this reference trajectory is guaranteed by the inner motion control.

Depending on the adopted interaction control strategy, the reference trajectory is computed on the basis of the desired motion trajectory and of a desired dynamic behavior (as for impedance control) or on the basis of a desired force and moment and a desired motion trajectory (as for parallel control).

In order to further investigate the different interaction control schemes, the kinematic model of the manipulator has to be introduced. For an open-chain robot manipulator, the relationship between the $(n \times 1)$ joint vector \mathbf{q} and the (3×1) position vector \mathbf{p}_e and the (3×3) rotation matrix \mathbf{R}_e has the form

$$\mathbf{p}_e = \mathbf{p}_e(\mathbf{q}) \quad (1)$$

$$\mathbf{R}_e = \mathbf{R}_e(\mathbf{q}) \quad (2)$$

where the quantities \mathbf{p}_e and \mathbf{R}_e characterize the position and orientation of the end-effector frame Σ_e with respect to a fixed base frame Σ_b . At industrial level, alternative representations of end-effector orientation are adopted, namely, minimal representations such as Euler angles, or non-minimal representations such as the unit quaternion. Hereafter, the unit quaternion is adopted, since it avoids representation singularities and possess a clear geometrical meaning. The unit quaternion is defined as

$$\mathcal{Q}_e = \{\eta_e, \boldsymbol{\epsilon}_e\},$$

with

$$\eta_e = \cos \frac{\theta}{2} \quad (3)$$

$$\boldsymbol{\epsilon}_e = \sin \frac{\theta}{2} \mathbf{r}, \quad (4)$$

where θ is the rotation angle about the axis \mathbf{r} required to align the base frame to the end-effector frame. Further details on unit quaternions and their use in force control applications can be found, e.g., in [14] and references therein.

Therefore, hereafter the desired motion trajectory for the end effector is specified in terms of the position \mathbf{p}_d and orientation \mathcal{Q}_d of a desired frame Σ_d , while the reference trajectory for the inner motion controller is indicated with the position \mathbf{p}_r and with the orientation \mathcal{Q}_r of a reference frame Σ_r .

2.1 Impedance control

Impedance control is designed to achieve a desired dynamic behavior of the interaction, i.e., a mechanical impedance usually described by a mass-spring-damper system.

In order to implement this strategy, the position and orientation of the desired frame together with the measured contact force and moment are input to the impedance controller which generates the position \mathbf{p}_c and orientation \mathcal{Q}_c of a compliant frame Σ_c , describing the end-effector behavior corresponding to the desired impedance. In detail, a suitable impedance equation is designed between the contact force and moment and the displacement of the desired frame Σ_d with respect to the compliant frame Σ_c .

The translational part of the impedance equation is chosen so as to enforce an equivalent mass-damper-spring behavior for the position displacement $\Delta \mathbf{p}_{dc} = \mathbf{p}_d - \mathbf{p}_c$ when the end effector exerts a force \mathbf{f} on the environment, i.e.

$$\mathbf{M}_p \Delta \ddot{\mathbf{p}}_{dc} + \mathbf{D}_p \Delta \dot{\mathbf{p}}_{dc} + \mathbf{K}_p \Delta \mathbf{p}_{dc} = \mathbf{f}, \quad (5)$$

where \mathbf{M}_p , \mathbf{D}_p and \mathbf{K}_p are positive definite matrices.

The rotational part of the impedance equation can be chosen as

$$\mathbf{M}_o \Delta {}^c \dot{\boldsymbol{\omega}}_{dc} + \mathbf{D}_o \Delta {}^c \boldsymbol{\omega}_{dc} + \mathbf{K}'_o {}^c \boldsymbol{\epsilon}_{dc} = {}^c \boldsymbol{\mu}, \quad (6)$$

where ${}^c \boldsymbol{\mu}$, is the moment exerted by the end effector on the environment referred to the frame Σ_c , and $\Delta {}^c \boldsymbol{\omega}_{dc} = {}^c \boldsymbol{\omega}_d - {}^c \boldsymbol{\omega}_c$, being ${}^c \boldsymbol{\omega}_d$ (${}^c \boldsymbol{\omega}_c$) the angular velocity of the frame Σ_d (Σ_c) referred to the frame Σ_c . The quantity ${}^c \boldsymbol{\epsilon}_{dc}$ is the vector part of the quaternion $\mathcal{Q}_{dc} = \mathcal{Q}_c^{-1} * \mathcal{Q}_d$ (see, e.g., [14] for the definition of the inverse operator and the composition operator $*$). The rotational stiffness matrix in (6) is

$$\mathbf{K}'_o = 2\mathbf{E}^T(\eta_{dc}, {}^c \boldsymbol{\epsilon}_{dc}) \mathbf{K}_o \quad (7)$$

with

$$\mathbf{E}(\eta_{dc}, {}^c \boldsymbol{\epsilon}_{dc}) = \eta_{dc} \mathbf{I} - \mathbf{S}({}^c \boldsymbol{\epsilon}_{dc}), \quad (8)$$

being η_{dc} the scalar part of \mathcal{Q}_{dc} and $\mathbf{S}(\cdot)$ the skew symmetric operator [18].

It is worth remarking that for equation (7) the property of task geometric consistency can be shown [3], i.e. the rotational stiffness matrix \mathbf{K}_o can be expressed in terms of three parameters representing the stiffness about three principal axes. As a consequence, an elastic moment about such an axis produces an orientation displacement about the same axis.

A block diagram of the impedance controllers for the translational and rotational part is reported in Figs. 2 and 3, respectively. Notice that the position and orientation trajectory for the compliant frame Σ_c coincides with the position and orientation trajectory of the reference frame Σ_r to be tracked by the inner motion controller. Moreover, in Fig. 3, the quaternion propagation rule [14] is used to relate the angular velocity to the time derivative of the unit quaternion.

2.2 Parallel force/position control

The objective of the parallel force/position control is to regulate the contact force and moment to a desired value assigned along the constrained directions, and track a desired position and orientation trajectory assigned along the unconstrained directions.

To realize this strategy, the desired force and moment together with the measured contact force and moment are input to the force controller, which generates the position and orientation trajectory for the compliant frame Σ_c ; these are composed with the desired position and orientation trajectory to compute the position and orientation trajectory for the reference frame Σ_r to be tracked by the inner motion controller.

Notice that both force and motion are controlled in all task directions, and thus the control actions can be designed on the basis of a simplified model of the environment, so providing robustness to the uncertain environment knowledge.

For the translational part, the idea of parallel control is to compose the position \mathbf{p}_c computed by the force controller with the desired position as

$$\mathbf{p}_r = \mathbf{p}_c + \mathbf{p}_d. \quad (9)$$

The corresponding reference velocity and acceleration are computed as

$$\dot{\mathbf{p}}_r = \dot{\mathbf{p}}_c + \dot{\mathbf{p}}_d \quad (10)$$

$$\ddot{\mathbf{p}}_r = \ddot{\mathbf{p}}_c + \ddot{\mathbf{p}}_d. \quad (11)$$

The displacement \mathbf{p}_c in (9) is the solution to the differential equation

$$k_{Af}\ddot{\mathbf{p}}_c + k_{Vf}\dot{\mathbf{p}}_c = \Delta\mathbf{f} \quad (12)$$

with $k_{Af} > 0$ and $k_{Vf} > 0$. It is worth pointing out that the position \mathbf{p}_c resulting from integration of (12) provides an integral control action on the force error. This guarantees regulation of the contact force to the desired value along the constrained task directions, and tracking of the desired end-effector position along the unconstrained task directions [6].

As regards the rotational part, the parallel composition is defined as

$$\mathbf{Q}_r = \mathbf{Q}_c * \mathbf{Q}_{dc} \quad (13)$$

$${}^c\boldsymbol{\omega}_r = {}^c\boldsymbol{\omega}_c + {}^c\boldsymbol{\omega}_{dc} \quad (14)$$

$${}^c\dot{\boldsymbol{\omega}}_r = {}^c\dot{\boldsymbol{\omega}}_c + {}^c\dot{\boldsymbol{\omega}}_{dc}, \quad (15)$$

where \mathbf{Q}_{dc} , $\boldsymbol{\omega}_{dc}$ and $\dot{\boldsymbol{\omega}}_{dc}$ represent the desired rotational motion. The double subscript indicates that such a motion shall be assigned as a relative motion with respect the compliant frame Σ_c . The rotational motion of Σ_c is computed by solving the differential equation

$$k_{Am}{}^c\dot{\boldsymbol{\omega}}_c + k_{Vm}{}^c\boldsymbol{\omega}_c = \Delta^c\boldsymbol{\mu} \quad (16)$$

with $k_{Am} > 0$ and $k_{Vm} > 0$. Similarly to the translational part of the controller, \mathcal{Q}_c is computed by integrating the equation (16) together with the quaternion propagation rule, thus providing an integral control action on the moment error.

A block diagram of the force and moment controllers are sketched in Figs. 4,5, where the parallel composition is evidenced.

3 Motion control schemes

The interaction control schemes presented above require the presence of an inner motion control loop, which can be realized according to different architectures.

3.1 Independent joint control

The basic solution for an industrial control architecture is represented in Fig. 6, where the inner loop is the native position control for each joint. In detail, the inner motion loop, for each joint, operates to ensure tracking of a reference trajectory, which is computed by the centralized interaction controller.

The resulting reference trajectory in the Cartesian space, expressed in terms of position \mathbf{p}_r and orientation \mathcal{Q}_r , has to be translated into the corresponding motion trajectory in the joint space \mathbf{q}_r . This is accomplished by resorting to a suitable inverse kinematics algorithm, which can be implemented, e.g., in a closed-loop fashion [19]. If the inner motion loop requires velocity and acceleration feed-forward actions, a second-order closed-loop inverse kinematic algorithm can be adopted [1],[5].

This solution can be certainly very easily implemented on a standard industrial robot control unit, which already possess most of the functionalities required by the control scheme, namely a joint position control loop and the inverse kinematics algorithm. In fact, some “open control architectures” are commercially available, which allows the user to correct a planned trajectory, both at Cartesian level and at joint level, at a fixed sampling rate. It is well-known that a bandwidth of the inner position loop much higher than the bandwidth of the outer force control loop is a requirement for closed-loop stability. Despite of the wide bandwidth guaranteed by the inner joint motion loop of many industrial controllers, most of these implementations require a few sampling time intervals before the correction to the planned trajectory can be actuated. This is imputed to the cascaded control structure of the joint servo loop, which contains several inner loops, i.e. a current loop, a velocity loop and a position loop, which confer to the servo system very good tracking performance.

In fact, the cascaded structure causes a finite dead-time on the input of the motion-controlled robot; this negatively affects the stability of the closed-loop system and unavoidably limits the performance of the force control loop [13]. Hence, it is worth pursuing further improvements of the performance of the inner motion loop.

3.2 Resolved joint acceleration control

By keeping the same architecture depicted in Fig. 6, a possible solution to overcome the limits of the independent joint control consists in the adoption of a model-based inner position control loop. The aim is to linearize and decouple the manipulator dynamics, thus allowing the design of interaction control strategies with enhanced performances.

Modelling of a robot manipulator is therefore a necessary premise to designing model-based control strategies. The dynamic model of a n -joints robot manipulator can be written in the Lagrangian form as

$$\mathbf{B}(\mathbf{q})\ddot{\mathbf{q}} + \mathbf{C}(\mathbf{q}, \dot{\mathbf{q}})\dot{\mathbf{q}} + \mathbf{F}\dot{\mathbf{q}} + \mathbf{g}(\mathbf{q}) = \boldsymbol{\tau} - \mathbf{J}^T(\mathbf{q})\mathbf{h}. \quad (17)$$

where \mathbf{B} is the $(n \times n)$ symmetric and positive definite inertia matrix, $\mathbf{C}\dot{\mathbf{q}}$ is the $(n \times 1)$ vector of Coriolis and centrifugal torques, $\mathbf{F}\dot{\mathbf{q}}$ is the $(n \times 1)$ vector of viscous friction torques, and \mathbf{g} is the $(n \times 1)$ vector of gravity torques. Also, in (17), $\mathbf{h} = [\mathbf{f}^T \quad \boldsymbol{\mu}^T]^T$, where \mathbf{f} denotes the (3×1) vector of external end-effector force and

$\boldsymbol{\mu}$ the (3×1) vector of external end-effector moment, respectively. The matrix \mathbf{J} is the $(6 \times n)$ end-effector geometric Jacobian, providing the relationship between the joint velocity vector $\dot{\mathbf{q}}$ and the end-effector linear ($\dot{\mathbf{p}}_e$) and angular ($\boldsymbol{\omega}_e$) velocities in the form

$$\mathbf{v}_e = \begin{bmatrix} \dot{\mathbf{p}}_e \\ \boldsymbol{\omega}_e \end{bmatrix} = \mathbf{J}(\mathbf{q})\dot{\mathbf{q}}. \quad (18)$$

A model-based solution to the motion control problem is represented by the inverse dynamics control, which is aimed at linearizing and decoupling the manipulator dynamics via feedback. Nonlinearities such as Coriolis and centrifugal torques, friction torques and gravity torques can be merely cancelled by adding these terms to the control input, while decoupling can be achieved by weighting the control input by the inertia matrix. According to this dynamic model-based compensation, the joint torques can be chosen as

$$\boldsymbol{\tau} = \mathbf{B}(\mathbf{q})\boldsymbol{\alpha} + \mathbf{C}(\mathbf{q}, \dot{\mathbf{q}})\dot{\mathbf{q}} + \mathbf{F}\dot{\mathbf{q}} + \mathbf{g}(\mathbf{q}) + \mathbf{J}^T(\mathbf{q})\mathbf{h} \quad (19)$$

where $\boldsymbol{\alpha}$ constitutes a new control input to be properly designed.

Folding the control law (19) into the system model (17), and taking into account that $\mathbf{B}(\mathbf{q})$ is always nonsingular, yields

$$\ddot{\mathbf{q}} = \boldsymbol{\alpha} \quad (20)$$

which constitutes a linear and decoupled system corresponding to a double integrator between the input $\boldsymbol{\alpha}$ and the output \mathbf{q} . The quantity $\boldsymbol{\alpha}$ is the new control input to be designed to achieve tracking of a reference end-effector position and orientation trajectory. The usual way of solving this problem consists of two stages; first the manipulator inverse kinematics is computed to provide the vector of joint variables \mathbf{q}_r corresponding to the given end-effector position and orientation. Then, tracking of \mathbf{q}_r is achieved by choosing $\boldsymbol{\alpha}$ as follows

$$\boldsymbol{\alpha} = \ddot{\mathbf{q}}_r + \mathbf{K}_{Dq}\Delta\dot{\mathbf{q}}_{re} + \mathbf{K}_{Pq}\Delta\mathbf{q}_{re} \quad (21)$$

Notice that this algorithm requires both velocity and acceleration feed-forward actions, therefore the most suitable inverse kinematics algorithm is the second-order strategy [3].

3.3 Resolved task-space acceleration control

The alternative architecture sketched in Fig. 7 is based on the so-called task-space control, since the inner motion control loop utilizes direct task-space feedback. Therefore, explicit computation of the manipulator inverse kinematics is no longer required. The task-space variables, i.e., the end-effector position and orientation, are computed via the direct kinematics relationships from the joint measurements.

In order to design a control action acting directly on the end-effector position and orientation errors, it is worth considering the time derivative of (18)

$$\dot{\mathbf{v}}_e = \mathbf{J}(\mathbf{q})\ddot{\mathbf{q}} + \dot{\mathbf{J}}(\mathbf{q}, \dot{\mathbf{q}})\dot{\mathbf{q}}, \quad (22)$$

which provides the relationship between the joint accelerations and the end-effector linear and angular accelerations. Then, the new control input $\boldsymbol{\alpha}$ in (20) can be chosen as

$$\boldsymbol{\alpha} = \mathbf{J}^{-1}(\mathbf{q}) \left(\mathbf{a} - \dot{\mathbf{J}}(\mathbf{q}, \dot{\mathbf{q}})\dot{\mathbf{q}} \right), \quad (23)$$

which in view of (22) leads to

$$\dot{\mathbf{v}}_e = \mathbf{a} \quad (24)$$

where \mathbf{a} attains the meaning of a resolved acceleration in terms of end-effector variables.

In deriving (23), a nonredundant manipulator ($n = 6$) moving in a singularity-free region of the workspace has been considered to compute the inverse of the Jacobian.

In view of the partition of \mathbf{v}_e , it is appropriate to partition the vector \mathbf{a} into its linear and angular components, i.e. $\mathbf{a} = [\mathbf{a}_p^T \quad \mathbf{a}_o^T]^T$ where \mathbf{a}_p and \mathbf{a}_o are (3×1) vectors. Therefore, Equation (24) can be rewritten as

$$\ddot{\mathbf{p}}_e = \mathbf{a}_p \quad (25)$$

$$\dot{\boldsymbol{\omega}}_e = \mathbf{a}_o, \quad (26)$$

where \mathbf{a}_p and \mathbf{a}_o shall be designed so as to ensure tracking of the reference end-effector position and orientation trajectory, respectively.

The resolved linear acceleration can be chosen as

$$\mathbf{a}_p = \ddot{\mathbf{p}}_r + \mathbf{K}_{Dp}\Delta\dot{\mathbf{p}}_{re} + \mathbf{K}_{Pp}\Delta\mathbf{p}_{re}, \quad (27)$$

where \mathbf{K}_{Dp} and \mathbf{K}_{Pp} are suitable feedback matrix gains and $\Delta\mathbf{p}_{re} = \mathbf{p}_r - \mathbf{p}_e$.

As regards the resolved angular acceleration, \mathbf{a}_o can be chosen as

$$\mathbf{a}_o = \dot{\boldsymbol{\omega}}_r + \mathbf{K}_{Do}\Delta\boldsymbol{\omega}_{re} + \mathbf{K}_{Po}\boldsymbol{\epsilon}_{re}, \quad (28)$$

where \mathbf{K}_{Do} and \mathbf{K}_{Po} are suitable feedback matrix gains, $\Delta\boldsymbol{\omega}_{re} = \boldsymbol{\omega}_r - \boldsymbol{\omega}_e$, and the orientation error is the vector part of the unit quaternion $\mathbf{Q}_{re} = \mathbf{Q}_e^{-1} * \mathbf{Q}_r$, referred to the base frame. Further details on the implementation of resolved task-space acceleration control can be found in [2].

4 Implementation of the control architecture

In this section the real-time implementation of the control schemes described above is discussed for the open version of the COMAU C3G 9000 control unit. A picture of the COMAU robots Smart-3S available at PRISMA Lab is reported in Fig. 8.

The C3G 9000 control unit has a VME-based architecture with 2 processing boards (Robot CPU and Servo CPU) both based on a Motorola 68020/68882, where the latter has an additional DSP and is in charge of trajectory generation, inverse kinematics and joint position servo control. Independent joint control is adopted where the individual servos are implemented as standard PID controllers. The native robot programming language is PDL 2, a high-level Pascal-like language with typical motion planning instructions.

An open version of the control unit is available [9] which allows testing of advanced control algorithms on a conventional industrial robot. Connection of the VME bus of the C3G 9000 unit to the ISA bus of a standard PC is made possible by a BIT 3 Computer bus adapter board, and the PC and C3G controller communicate via the shared memory available in the Robot CPU; at present, a PC Pentium MMX/233 is used. Time synchronization is implemented by means of a synchronization flag set by the C3G and read by the PC, exchanging data at a given sampling rate. A set of C routines are available to drive the bus adapter boards. These routines are collected in a library (PCC3Link) and are devoted to performing communication tasks, e.g., reading shaft motor positions and/or writing motor reference currents from/to the shared memory. Also, a set of routines are devoted to performing safety checks and monitoring system health, e.g., a watchdog function and/or maximum current checks.

Various operating modes are available in the control unit, allowing the PC to interact with the original controller both at trajectory generation level and at joint control level. To implement advanced control schemes, the operating mode number 4 is used in which the PC is in charge of computing the control algorithm and passing the references to the current servos through the communication link at 1 ms sampling time. Joint velocities are reconstructed through numerical differentiation of joint position readings.

A six-axis force/torque sensor ATI FT30-100 with force range of ± 130 N and torque range of ± 10 N·m can be mounted at arm's wrist. The sensor is connected to the PC by a parallel interface board which provides readings of six components of generalized force at 1 ms.

A schematic of the open control architecture is sketched in Fig. 9.

The C3G unit and the PC can exchange information by means of a shared memory where every millisecond (which is, therefore, the sampling time of the digital control loop) the C3G executes the following operations

- set the synchronization flag `IntActive`
- write the values of the motor shaft angular positions (in DAC units)
- read the desired values for the motor current set-points (in DAC units)

and the PC has to

- reset the synchronization flag `IntActive`
- read the values of the motor shaft angular positions (in DAC units)
- compute the current set-points for the next time interval
- write the desired values for the motor current set-points (in DAC units)
- set the watchdog flag

To fulfill the real-time constraint, the PC must execute the above steps within 0.7 ms. Otherwise, due to the watchdog mechanism, the system reaches an alarm state and the robot drives are switched off and brakes are activated. The watchdog timer, as in every real-time system, is aimed at preventing harmful consequences of unpredictable events, which can occur during the normal operation, e.g. when the PC crashes and thus is no longer able to write a correct current set-point to be actuated. The sequence outlined above is depicted in Fig. 10.

The ATI force sensor is interfaced via a parallel link to the ISA bus of the PC. To obtain the force measurement, the PC has to send a data acquisition request, which starts the data A/D conversion on the sensor conditioning electronics; after a time lapse of about 0.25 ms, the six components of the force are available in a memory buffer. Since the conversion is executed on remote hardware, during the conversion time interval, the PC can continue its elaboration. Hence, to avoid simply waiting the end of conversion, it is convenient to ask for the start of conversion at the beginning of the control cycle, perform all the computations not requiring force measurements, e.g. kinematics, dynamic model compensation, then read the force data and, finally, complete the control algorithm. The timing of these operations is outlined in Fig. 10.

Even though the use of a watchdog timer is a good solution for a safe operation of a real-time system, it cannot be considered enough to prevent every dangerous situation. In the C3G open control architecture several safety checks can be carried out to monitor the system functioning and to prevent damage, for example:

- set and check joint limits
- set and check maximum joint velocities
- set and check maximum instantaneous currents
- set and check maximum sustained currents
- set and check maximum forces and torques

The first type of safety check is particularly useful when an interaction task is tested, in fact limiting the robot workspace allows to prevent damage to the environment, to the robot and especially to the force sensor. Of course, monitoring also contact forces and moments allows a safer contact phase. Typically, interaction tasks evolve with limited velocities, hence monitoring joint velocities is also useful to detect undesirable situations during the task execution.

For the purpose of preventing damage of robot motors and motor drives, it is opportune to monitor the motor currents, whose instantaneous values should not exceed maximum fixed levels. Sometimes, it is also useful to limit the maximum sustained value of the currents, to prevent overheating of robot servo motors and their driving electronics.

Two different decisions can be taken in case one of the above checks fails, namely immediately stop the robot by invoking a special emergency routine, or set an error flag to exit from the main control loop and switch off the drives. The former approach is faster and thus safer, but requires a complete reboot of the system. The latter approach implies a delay of one sampling period, but leads the system to a less critical state, which does not require a complete reboot. The best policy is probably to decide on the particular safety check which failed, e.g., if one of the motor currents exceeded the maximum sustained value, an immediate stop is not necessary, on the contrary if the force is over the maximum allowed value, an immediate stop is advisable.

The software resources of the control unit can be organized in the modular multilayer structure of Fig. 11, where four layers are illustrated, both for the standard and the open operating modes, representing the various functions implemented in the controller.

The task planning function is implemented only in the standard operating mode via a PDL 2 software module containing the motion instructions needed to execute a specific task. It is worth pointing out that such a function could be realized also in the open operating mode, if desired, by developing a library of C functions dedicated to task planning.

The trajectory generation layer is in charge of computing the motion trajectories corresponding to the planned task. In the standard operating mode, the references for the joint servos are directly computed by the Servo CPU via an inverse kinematics procedure with joint interpolation from the specifications given in the PDL 2 program. In the open operating mode, the position vector \mathbf{p}_r and unit quaternion \mathcal{Q}_r representing the origin and orientation of a reference frame are computed from the force and moment measurements on the basis of the interaction control strategy specified by the user.

The servo layer for the standard operating mode implements standard decentralized PID joint position control. In the open operating mode, the joint-space or the task-space model-based (inverse dynamics) motion control is implemented.

The bottom layer includes joint motors, joint position resolvers and, for the open operating mode, a six-axis force/torque sensor.

In order to program the system for the execution of a given task, the user has to develop different software modules, according to the particular interaction control strategy. If the strategy of the scheme in Fig. 6 is selected, where the native joint-space motion control is used, the user has to develop only the PDL2 module of the task planning level and the C module implementing the proper interaction control strategy in the trajectory generation level. Still in the case of the scheme of Fig. 6, if the model-based joint-space motion control law of Section 3.2 is adopted, the user has to develop the C module of the inverse kinematics algorithm at the trajectory generation level as well as the motion control algorithm in the servo level. Finally, if the scheme of Fig. 7 is considered, the C module implements a motion control algorithm that involves both trajectory generation and servo levels.

5 Conclusion

The interaction control schemes presented throughout the paper have been tested in a number of experiments for representative interaction tasks, as extensively described in [20]. A commercially available industrial robot has been purposefully utilized to demonstrate the credibility of force control in the perspective of the next generation of robot control units with enhanced sensory feedback capabilities. The problem of interfacing the force/torque sensor has been solved thanks to the open architecture of the control unit. This feature is crucial also for the implementation of control schemes requiring model-based compensation actions.

The punch line is that the field is mature for a transfer of know-how from the academic to the industrial community. In this respect, encouraging signals are coming from leading industrial robot manufacturers and some product embedding a force sensor is already on the market.

Acknowledgements

This paper is largely based on [20]. The work was supported by *Ministero dell'Università e della Ricerca Scientifica e Tecnologica*.

References

- [1] Caccavale, F., Chiaverini, S., and Siciliano, B. (1997). Second-order kinematic control of robot manipulators with Jacobian damped least-squares inverse: Theory and experiments. *IEEE/ASME Transactions on Mechatronics*, 2:188-194.
- [2] Caccavale, F., Natale, C., Siciliano, B., and Villani, L. (1998). Resolved-acceleration control of robot manipulators: A critical review with experiments. *Robotica*, 16:565-573.
- [3] Caccavale, F., Natale, C., Siciliano, B., and Villani, L. (1999). Six-DOF impedance control based on angle/axis representations. *IEEE Trans. on Robotics and Automation*, 15:289-300.
- [4] Chiaverini, S. and Sciavicco, L. (1993). The parallel approach to force/position control of robotic manipulators. *IEEE Trans. on Robotics and Automation*, 9:361-373.
- [5] Chiaverini, S., and Siciliano, B. (1999). The unit quaternion: A useful tool for inverse kinematics of robot manipulators. *Systems Analysis, Modelling and Simulation*, 35:4560.
- [6] Chiaverini, S., Siciliano, B., and Villani, L. (1999). A survey of robot interaction control schemes with experimental comparison. *IEEE/ASME Trans. on Mechatronics*, 4:273-285.
- [7] De Schutter, J., Bruyninckx, H., Zhu, W.-H., and Spong, M.W. (1998). Force control: A bird's eye view. In *Control Problems in Robotics and Automation*, Siciliano, B. and Valavanis, K.P. (Eds.), Springer-Verlag, London, pp. 1-17.
- [8] De Schutter, J. and Van Brussel, H. (1988). Compliant robot motion II. A control approach based on external control loops. *Int. J. of Robotics Research*, 7(4):18-33.
- [9] Dogliani, F., Magnani, G., and Sciavicco, L. (1993) An open architecture industrial controller. *News of IEEE Robotics and Automation Soc*, 7(3):19-21.
- [10] Hogan, N. (1985). Impedance control: An approach to manipulation: Parts I—III. *ASME J. of Dynamic Systems, Measurement, and Control*, 107:1-24.
- [11] McClamroch, N.H. and Wang, D. (1988). Feedback stabilization and tracking of constrained robots. *IEEE Trans. on Automatic Control*, 33:419-426.
- [12] Mills, J.K. and Goldenberg, A.A. (1989). Force and position control of manipulators during constrained motion tasks. *IEEE Trans. on Robotics and Automation*, 5:30-46.
- [13] Natale, C., Koeppel, R., and Hizinger, G. (2000). A Systematic Design Procedure of Force Controllers for Industrial Robots. *IEEE/ASME Trans. on Mechatronics*, 5:122-123.
- [14] Natale, C. (2003). *Interaction Control of Robot Manipulators—Six-degrees-of-freedom Tasks*. Springer-Verlag, Berlin.
- [15] Paul, R. and Shimano, B. (1976). Compliance and control. In *Proc. 1976 Joint Automatic Control Conf.*, San Francisco, CA, pp. 694-699.

- [16] Raibert, M.H. and Craig, J.J. (1981). Hybrid position/force control of manipulators. *ASME J. of Dynamic Systems, Measurement, and Control*, 103:126–133.
- [17] Salisbury, J.K. (1980). Active stiffness control of a manipulator in Cartesian coordinates. In *Proc. 19th IEEE Conf. on Decision and Control*, Albuquerque, NM, pp. 95–100.
- [18] Sciavicco, L. and Siciliano, B. (2000). *Modelling and Control of Robot Manipulators*. 2nd Ed., Springer-Verlag, London.
- [19] Siciliano, B. (1990). A closed-loop inverse kinematic scheme for on-line joint-based robot control, *Robotica*, 8:231243.
- [20] Siciliano, B. and Villani, L. (1999). *Robot Force Control*. Kluwer Academic Publishers, Boston, MA.
- [21] Takegaki, M. and Arimoto, S. (1981). A new feedback method for dynamic control of manipulators. *ASME J. of Dynamic Systems, Measurement, and Control*, 102:119–125.
- [22] Whitney, D.E. (1987). Historical perspective and state of the art in robot force control. *Int. J. of Robotics Research*, 6(1):3–14.
- [23] Yoshikawa, T. (1987). Dynamic hybrid position/force control of robot manipulators—Description of hand constraints and calculation of joint driving force. *IEEE J. of Robotics and Automation*, 3:386–392.

List of Figures

1	Interaction control architecture.	14
2	Impedance control – translational part.	14
3	Impedance control – rotational part.	15
4	Parallel force/position control – translational part.	15
5	Parallel force/position control – rotational part.	16
6	Interaction control with inner joint-space motion control loop.	16
7	Interaction control with inner task-space motion control loop.	17
8	Robots COMAU Smart-3 S at PRISMA Lab.	17
9	C3G 9000 open control architecture.	18
10	Timing diagram of the real-time control loop	19
11	Modular multilayer control structure.	19

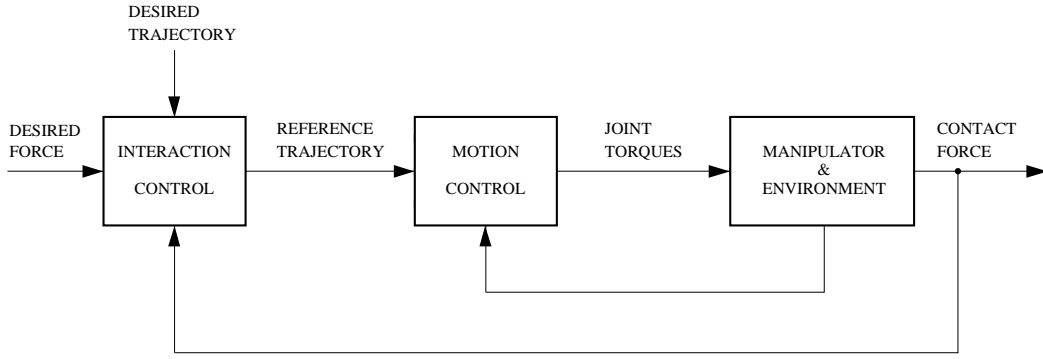


Figure 1: Interaction control architecture.

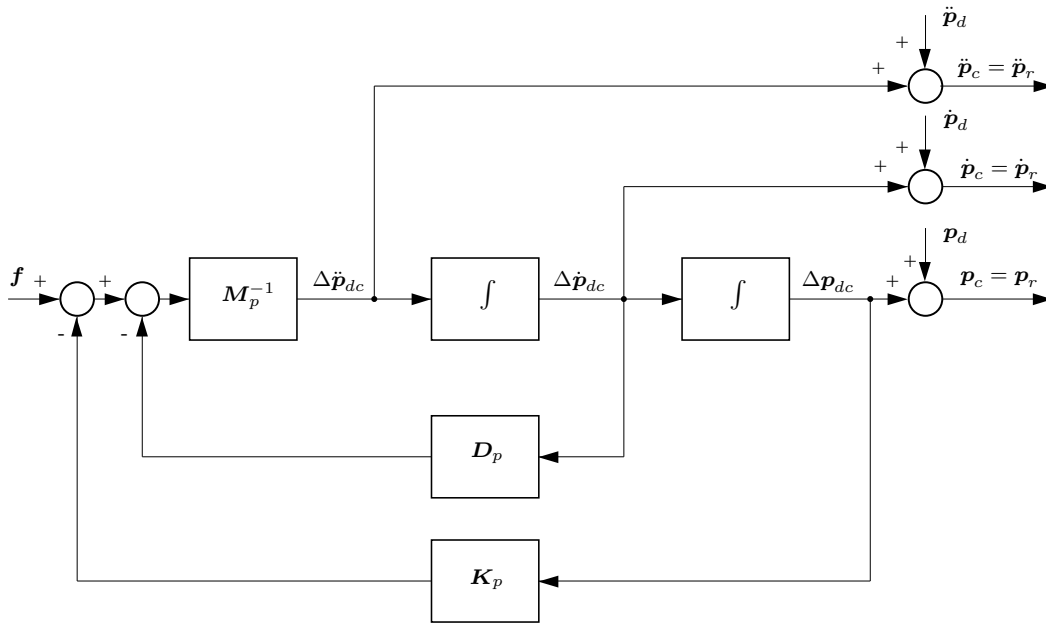


Figure 2: Impedance control – translational part.

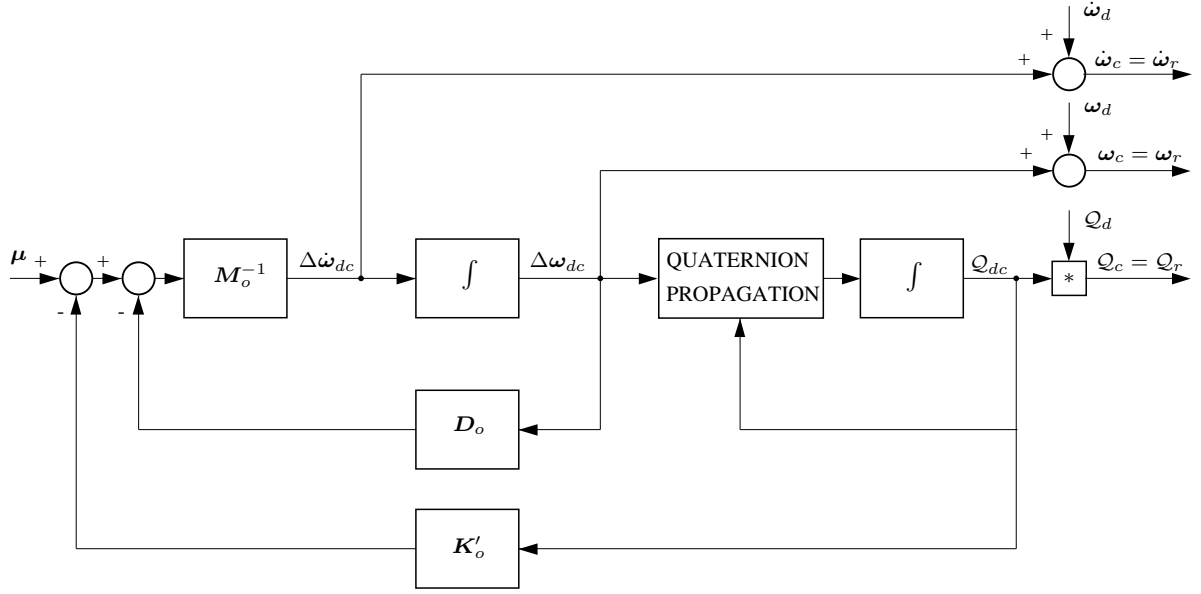


Figure 3: Impedance control – rotational part.

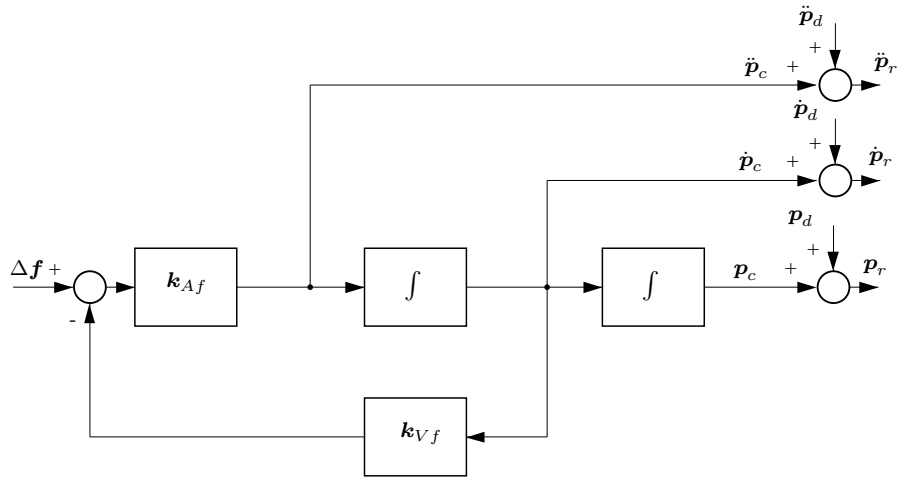


Figure 4: Parallel force/position control – translational part.

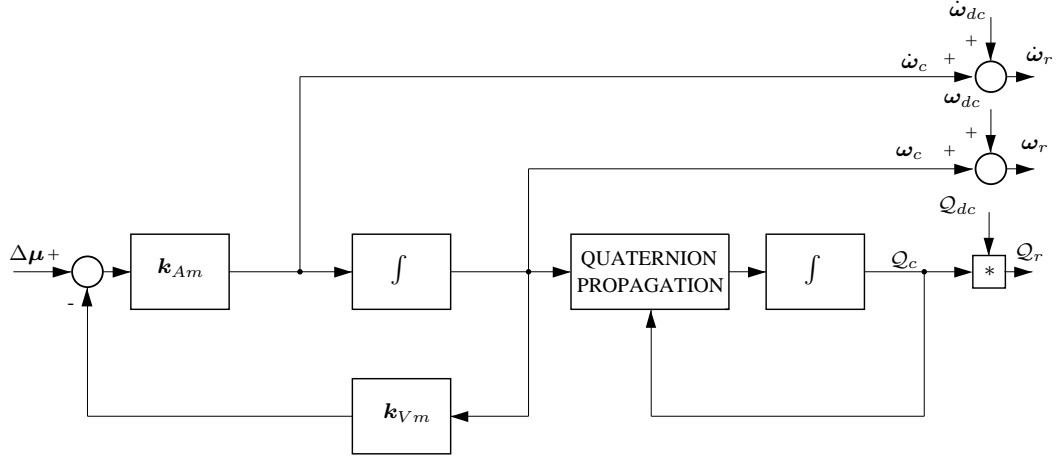


Figure 5: Parallel force/position control – rotational part.

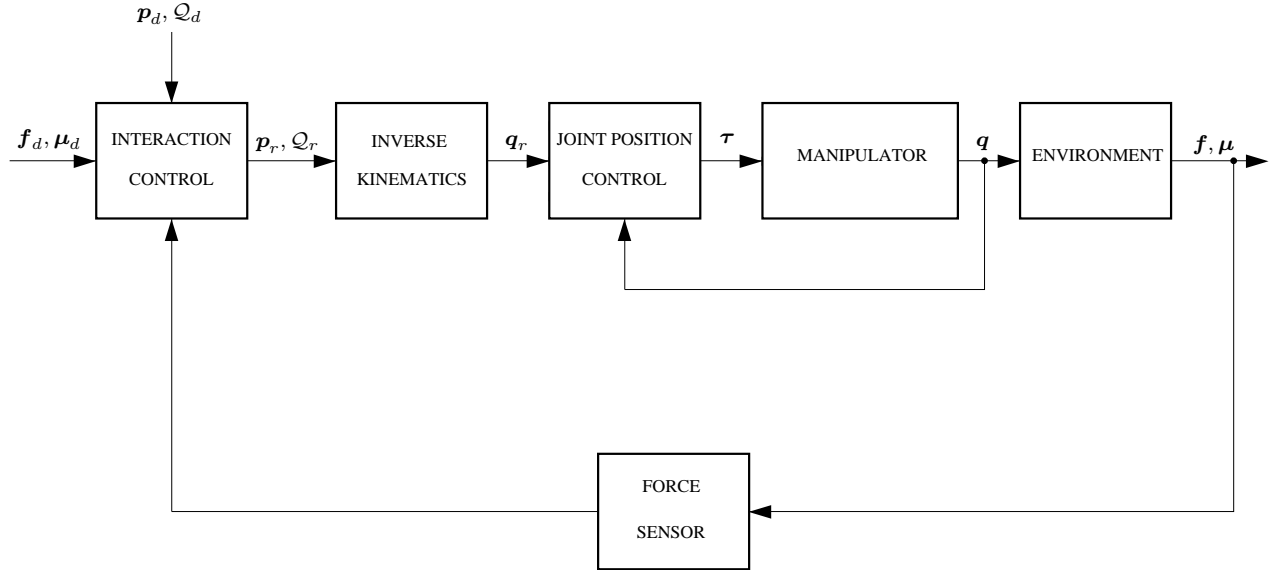


Figure 6: Interaction control with inner joint-space motion control loop.

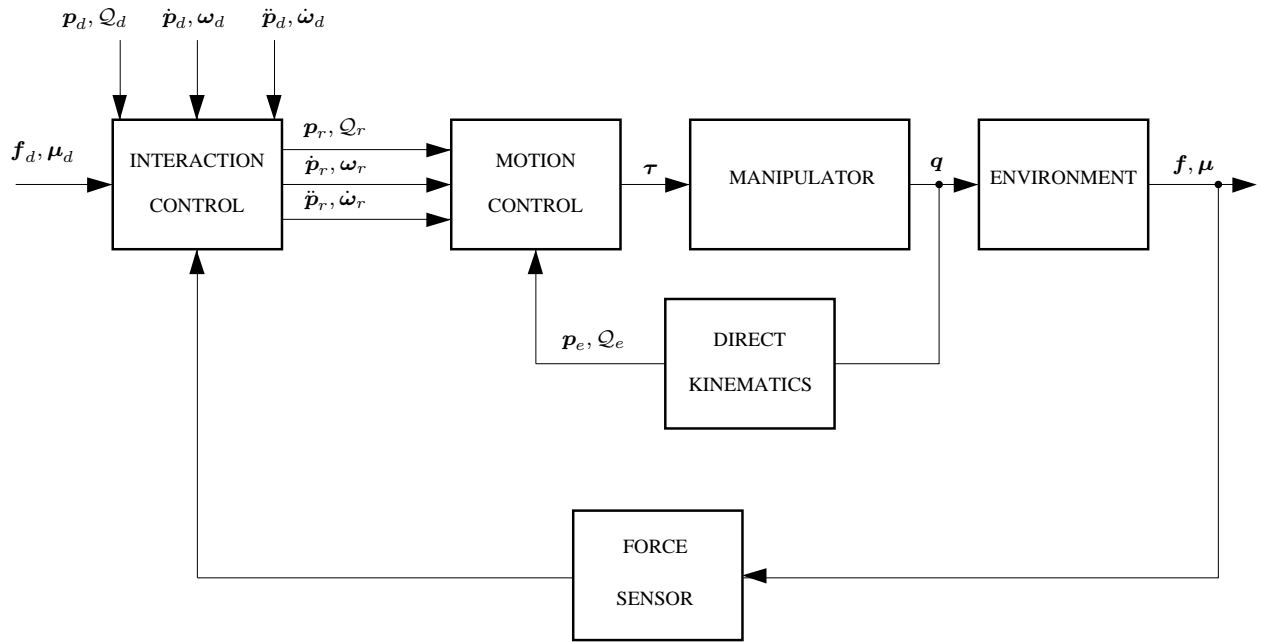


Figure 7: Interaction control with inner task-space motion control loop.

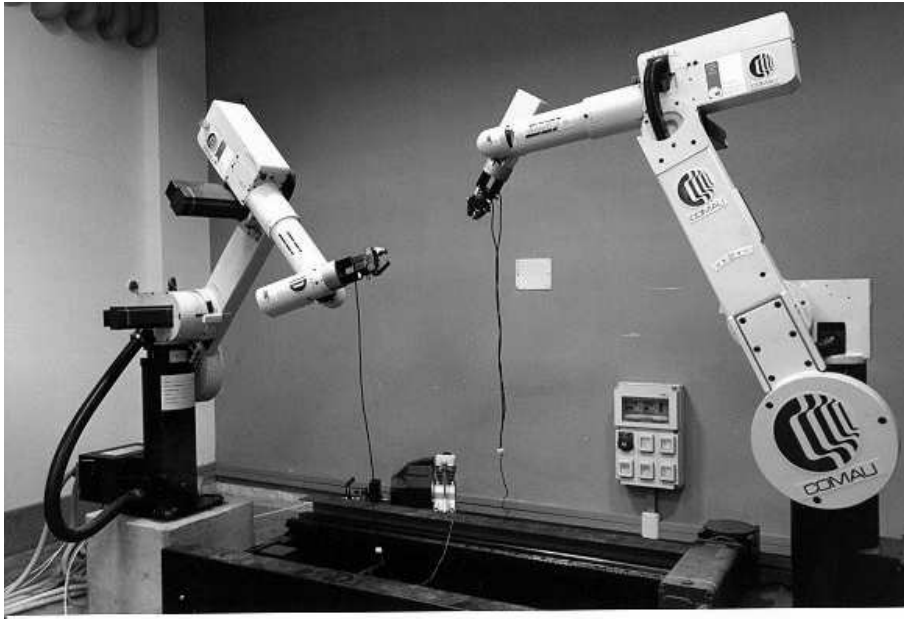


Figure 8: Robots COMAU Smart-3 S at PRISMA Lab.

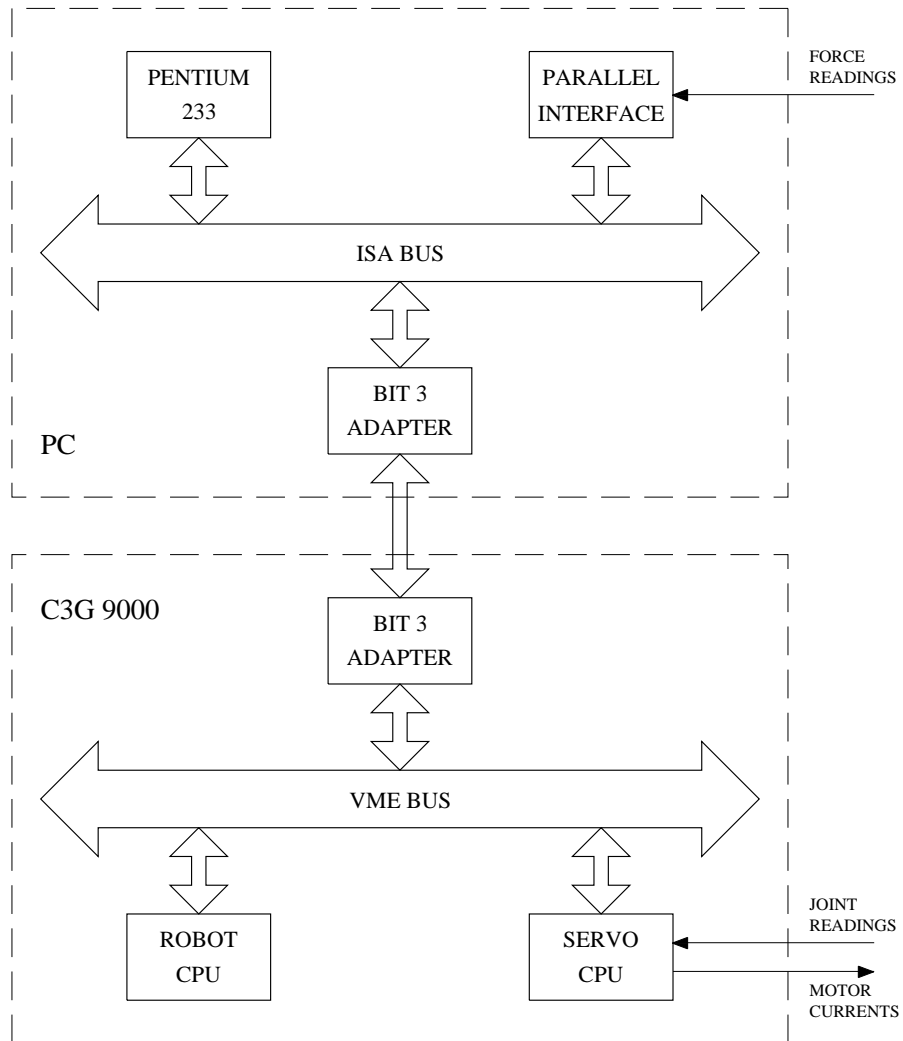


Figure 9: C3G 9000 open control architecture.

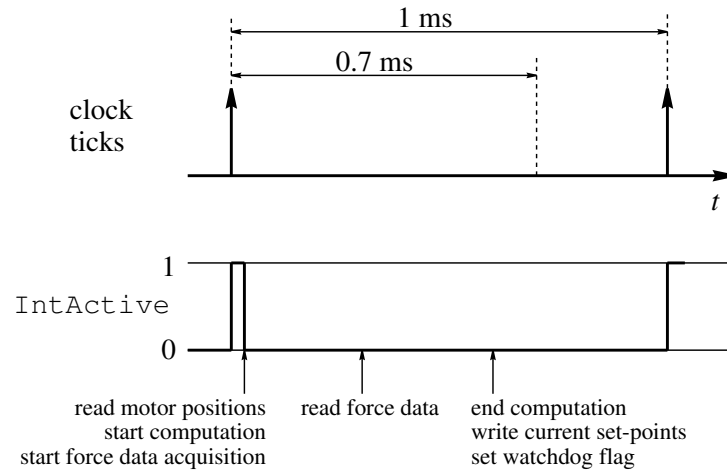


Figure 10: Timing diagram of the real-time control loop

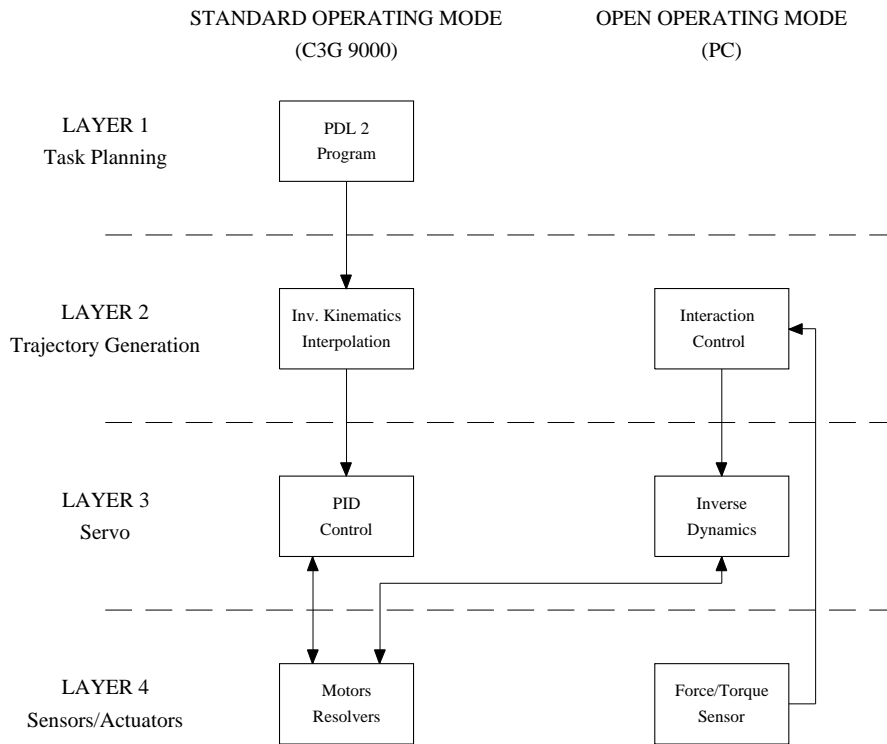


Figure 11: Modular multilayer control structure.