Pergamon

Control Eng. Practice, Vol. 4, No. 2, pp. 239-249, 1996
Copyright © 1996 Elsevier Science Ltd
Printed in Great Britain. All rights reserved
0967-0661/96 $15.00 + 0.00

PII:S0967-0661(95)00231-6

THE STRUCTURED DESIGN OF AN INDUSTRIAL ROBOT CONTROLLER

G. Ferretti*, G. Magnani*, P. Putz and P. Rocco***

**Dipartimento di Elettronica e Informazione, Politecnico de Milano, Piazza Leonardo da Vinci 32, 20133 Milano , Italy (magnani@elet.polimi.it)*

***European Space Agency Research and Technology Center (WKR), P.O. Box 299, 2200 AG Noordwijk, The Netherlands*

(Received March 1995; in final form November 1995)

Abstract: This paper deals with the design of an industrial robot controller featuring sensor-based control capabilities. The design follows the guidelines of a methodology aimed at structuring the design process, to assure traceability between the requirements and capabilities of the controller and a high degree of operational flexibility. The design consists of three steps, namely activity analysis, functional design and implementation design. The methodology especially supports the first two steps with design principles and guidelines. The designed controller has the capabilities to execute machining operations, arc and laser welding, insertion of parts into others and other interesting operations from an industrial and commercial point of view. The new capabilities may be implemented optionally as an add-on to the base product.

Key Words: Robot control; industrial robots; force control; open control systems; design methodology

1. INTRODUCTION

The engineering design of a controller must satisfy a large number of requirements and constraints of different natures (e.g. performance, cost, safety and reliability), and it often involves a choice among several equally good solutions. Many factors, also subjective, may influence the choice. The definition of the right product features and the development of a design consistent with them are crucial to the realization of profitable products. The lack and need of a structured and systematic framework to assist this difficult task is well recognized (Åström and Wittenmark, 1990).

This study concerns this subject. The design of a new industrial controller is tackled according to a codified methodology. Referring to a state-of-the-art commercial product, the Comau C3G 9000, the first objective of the study is to design a new controller ("next generation", since it should replace the current product) with innovations based on the forthcoming commercial needs and advances in sensor, actuators and information technologies. The design development follows the Control Development Methodology (CDM), a design methodology defined by the European Space Agency (ESA) for structuring the design process of automation and robotics control systems for space applications. Actually, it is a second objective of the study to assess the effectiveness and possible drawbacks of the application of the CDM to industrial problems.

According to the CDM, the design process consists of three major steps. The first step, referred to as activity analysis, defines precisely and formally the tasks the robot has to accomplish. The second one (requirements or functional analysis) establishes the control functions required to accomplish the tasks and the interaction with the human operators. This entails, for instance, the choice of control principles and algorithms, as well as of control and measured variables. The third step (architectural design) concerns the detailed design of the controller hardware and software. The CDM supports the first two steps in particular, supplying principles, methods and tools in order to proceed in a structured and formal way.

The organization of the paper reflects the steps of the controller design process. In Section 2 the CDM is described. Section 3 deals with the activity analysis for next-generation industrial robots. Section 4 discusses, as an illustrative example, the functional analysis for hybrid position/force control, Section 5 outlines hardware and software implementation architectures.

2. THE CONTROL DEVELOPMENT METHODOLOGY

2.1 Objective and benefits

The CDM (Putz and Mau, 1992; Putz and Elfving, 1992) provides principles and guidelines for the design of open and extendable robot control systems within a simple overall structure, using unified and unambiguous terminology, and above all maintaining traceability between solution-independent requirements and final realizations.

2.2 The Steps of the CDM

The CDM defines a life-cycle model for the robot control system (which is seen as part of a larger "robot system"). Figure 1 illustrates the phases and steps within this life cycle. Guidelines and methods are offered in (Putz and Mau, 1992) for each of these phases, but special attention has been devoted to the initial steps 1 - 3.

The first step is called activity analysis. Already, the initial user requirements need to be refined in a systematic (hierarchical) structure. The proposed output from this step is a so-called "activity script", a semi-formal and fairly detailed prescription of the process which has to be automated by the robot system (key question: "FOR WHAT do I need automation and control ?").

The next step and phase is the refined analysis of the control (sub)system requirements. This has to take all the different kinds of requirements into account, most notably functional and performance requirements (stating WHAT control functions have to be performed HOW WELL) and (especially for space applications) operational requirements.

The last step particularly emphasized in the CDM is the control system architectural design which, for the software part, coincides with the well-known S/W architectural design phase. Also, control hardware and human control operations ("brainware") must be "designed" to the same level in this step. As opposed to the preceding requirements analysis phase, design starts to say HOW the required functions are going to be realized. This involves the specific processors, bus systems, real-time operating system services, algorithms, software communication schemes, operational procedures, etc.

The remaining steps after architectural design are the "conventional" detailed design and production/procurement (of control hardware and software), integration and testing of these parts to make the control system, and finally integration and testing of the control system into the overall robot system. This is by no means underestimated, but from its principles is so well known that it is not discussed any further here.

For each step, the CDM offers support in the form of generic architectural schemes (reference models), guidelines on their use, and even computer-based tool assistance. The two most important tools are an activity analysis method (ActAM) for Step 1 and a functional reference model (FRM) for Step 2. They will be introduced below.

2.3 Hierarchical activity analysis

The activity analysis method of the CDM is a systematic procedure to produce refined user requirements suitable for further system and subsystem analysis. From the analysis of the process to be automated, a hierarchical description of robot system activities to be performed is obtained.

The CDM proposes a three-layer hierarchy of activities:

- The very top level, robotic missions, represents the highest level of activity for which a robot

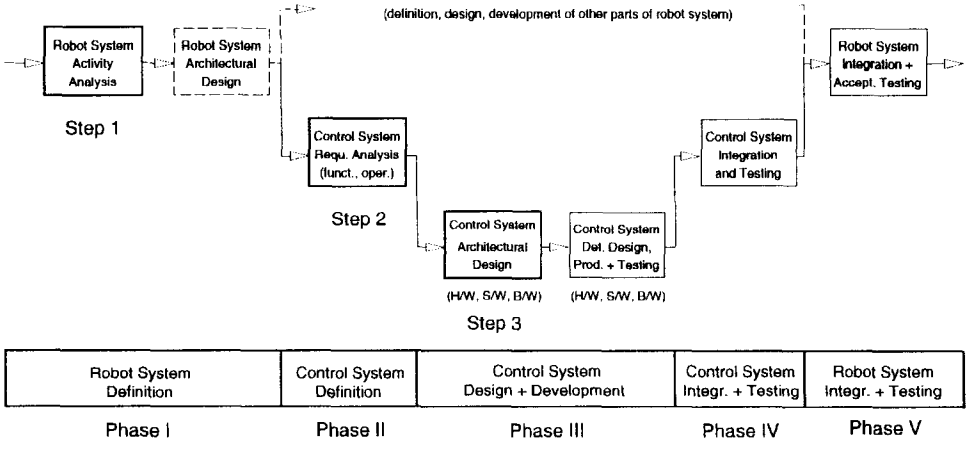

Fig. 1 Phases and Steps of the CDM

system is responsible (e.g. "SERVICE a life science experiment" , "REPAIR a satellite").

- Each mission can be decomposed into tasks, defined as the highest level of activity performed on a single subject (e.g., "OPEN a door", "INSTALL a sample in a processing furnace", "POLISH a surface", "WELD a seam").
- Finally, each task can be decomposed into actions (e.g., "GRIP a sample container", "DISPLACE tool to a position", "MOVE the container to the freezer", "INSERT the container in the port", "SLIDE a drawer", "RUB along a path on a surface", "FOLLOW a seam", "TRACK a part on a conveyor").

For each action, a realization with a particular control concept can be established (e.g., free continuous path control for MOVE, or impedance control for INSERT and SLIDE) such that there are well-defined criteria for identifying actions within a task.

The CDM offers a "catalog" of frequently occurring tasks and actions, together with templates on which action attributes should be defined. As an example, a template for the RUB action in a polishing application is reported in Section 3.

2.4 Functional requirements analysis

For control system requirements analysis, the key concept is to start with a purely functional analysis. Even the most diverse applications use the same fundamental control functions, but in possibly quite different realizations.

The control functions are cast in the same overall structure as the activities, namely a three-layer hierarchy responsible for achieving robot missions, tasks, and actions, and a general framework (logical model) for robot control functions, called the functional reference model (FRM) is defined. The top-level view of the FRM is shown in Fig. 2.

The global goal (the robot's mission objectives) is successively broken down into sub-goals: the tasks and actions, each handled by control functions on the appropriate layer, until the elementary "control outputs" can be issued to the robotic devices (e.g., currents to the joint motor drives).

The "vertical" branches of the FRM are based on the concept of feedback. The center branch is called forward control (FC). FC functions are responsible for activity decomposition, execution planning and control by taking the most appropriate a priori information known to each layer into account. The left branch consists of nominal feedback (NF) functions for the refinement and update of a priori knowledge ("world models"), based on the actual, but essentially expected, evolution of the process, and consequently the formulation of controlled adjustments of the FC. By FC and NF, "cascaded" control loops are closed on action, task, and mission

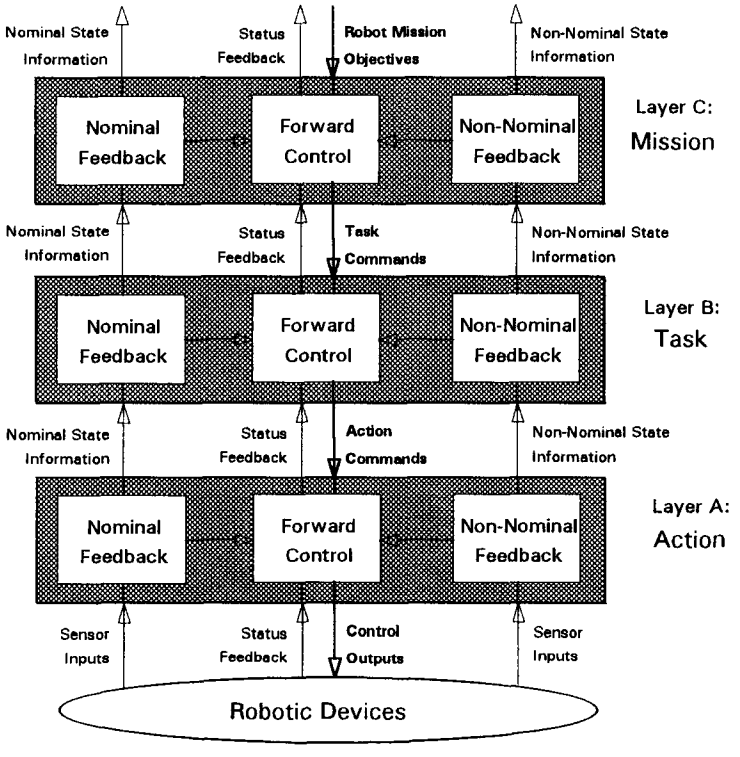

Fig. 2 Functional Reference Model (FRM) for Robot Control Systems

layers and are equipped with everything necessary for the "nominal" course of events. Besides FC and NF, the FRM foresees a third branch of "non-nominal feedback" (NNF) functions responsible for the monitoring of discrepancies between actual and allowable states in both the FC and NF functions, diagnosis of their origins, and generation of directives (including recovery strategies and constraints) for FC.

A much more refined definition of the FRM is given in (Putz and Mau, 1992). It should also be acknowledged that the FRM concept and structure have been heavily and beneficially influenced by the NASREM architecture developed at the US NIST (Albus, *et al.*, 1989).

2.5 Application reference model

The FRM is by its nature still very generic, and it applies to general automation control systems. To be more specific for the frequent application to "classical" robot control, the CDM has elaborated a more detailed reference model for this application class, called the application reference model (ARM).

The control concepts relevant to an industrial robot are completely located in the action layer of the FRM. In fact, the robot controller has to implement the control functions relevant to action planning, execution and control, while mission- and task-level functions are in the charge of the robot and workcell engineer, who has to program a mission execution as a suitable sequence of tasks, and tasks as sequences of actions. In the same way, it is a human operator that has to intervene to "rescue" unexpected, non-nominal situations. Therefore the ARM is a refinement of the FC and NF functionalities in the action layer.

In FC, one will thus find functions like path preparation and interpolation, path servo control, inverse kinematic transformation, and joint servo control. In NF, functions include proprioceptive and exteroceptive sensor data processing (SDP), and process-, or device-, and control-oriented data processing (PODP, CODP, respectively). Further details on this ARM can be found in (Putz and Mau, 1992), where it has also been shown that the most prominent control concepts can be well represented as certain instantiations of this general architecture.

For given user requirements (actions to be executed for a specific control application), the CDM recommends that the functional requirements should be derived according to the ARM framework. The resulting functional architecture (logical model) of the control system is called the application architecture (AA). An example is given in Section 4.

In the following sections, the design of an industrial controller is developed following the CDM principles and guidelines.

3. ACTIVITY ANALYSIS FOR INDUSTRIAL ROBOTS

Activity analysis performs a functional analysis of the application process(es) in order to define what has to be done by the robot (including the controller) under design. According to the CDM, the result of the activity analysis is the list of the activities (tasks, actions) the robot should be able to execute.

The activity analysis for industrial robots has been based on available technical and scientific documentation, and especially on the expertise of engineers from COMAU Robotics, the major Italian robot manufacturer and FMS integrator, who were extensively interviewed. Potential new applications have been carefully investigated with respect to their technical feasibility and commercial relevance for the immediate future. It has been found that it would be commercially interesting to improve the programming and control capabilities of the next generation of robots to allow easier and more effective application to tasks like polishing and deburring, arc-welding, peg-into-hole and conveyor-tracking. Such applications, in fact, require capabilities to generate a trajectory on-line or to control the interaction (force) with the environment that are not available in current industrial controllers. Sometimes they are tackled in industry using compliance devices such as remote center compliance (RCC), or very accurate (but expensive and inflexible) positioning mechanisms of the working parts, or special sensorized flange-mounted actuators that compensate for positioning errors (e.g., for arc welding) or for high contact forces (e.g., for deburring).

Robot system activities relevant to such applications have been analysed and decomposed into the actions of the CDM catalog (Putz and Mau, 1992). Works by Weule and Timmermann (1990) and Mizugaki, *et al.* (1990) for robotized polishing, by Her and Kazerooni (1991) and Bone, *et al.* (1991) for deburring, and by Whitney (1982) have been taken as references. It has been found that, besides basic actions like MOVE and APPROACH/RETRACT (to control gross and proximity motion) and ATTACH/DETACH (to grip/release the tool), new actions should be implemented, involving exteroceptive sensor-control capabilities. Among the most representative are RUB, required for polishing and deburring, FOLLOW, for arc welding and tracking, and INSERT for part mating. Templates have been drawn up for each action and for each application, thus obtaining a complete description of

Table 1 Performance Specification template for RUB action

Action	RUB
Application	Polishing an object (e.g. mechanical parts, molds, ...) with an appropriate tool (like a grinding tool).
Definition	RUB: to use the robot to move an object (payload) or another sub-system along a surface, maintaining continuous contact and exerting an assigned force.
Initial conditions	The tool is positioned at the beginning of the working path in contact with the surface to be worked.
Boundary cond.	The robot has to follow the working path maintaining the contact with the surface and exerting a constant pressure on it. For this purpose, the reference value of the normal force along the path is given by the user, while interaction torques should remain null.
Terminal cond.	The tool has reached the end point (known) of the working path.
Environment attributes	It is completely stiff. There could be some obstacles on the surface that the robot has to avoid. Their positions and shapes are known. Surface shape is known a priori via CAD or in some cases by teaching. The positions of edges of the workpiece are known.
Subject attributes	The shape of the tool is known.
Action performance requirements	Nominal pressure values : 0.1 - 1 N/mm^2 Nominal velocity : 1 - 30 mm/s Pressure accuracy: ± 10-20% of nominal value. Torque disturbance: ±0.1-0.2Nm Velocity accuracy: ± 5-10% of nominal value.
Motion control requirements	The nominal path is given

a robot system's activities. As an example, the template for the RUB action for a polishing application is shown in Table 1.

4. ANALYSIS OF CONTROL FUNCTIONS FOR HYBRID POSITION / FORCE CONTROL

The second step of the CDM consists in the specification of the controller's functionality in the form of an AA, to be obtained by summing up the functions required for each action listed in the activity script. For each action it is necessary to select a planning and a control algorithm, and then to split them according to the general ARM for robotics.

As an illustrative example, this section shows the derivation of the AA for the RUB action. First, general controller requirements and constraints influencing the selection of planning and control algorithms are discussed, and then the selected algorithms are described. Finally, the structure of the AA is shown.

4.1 General requirements and constraints

Industrial robot controllers are currently designed as positioning devices: independent positioning servomechanisms are implemented at each joint. This solution is well established, and is based on motion control technology and on the relevant commercial products and experience. For this reason it is assumed here that the independent joint control scheme will also be adopted in the next generation of controllers

In applications like polishing and deburring the manipulator may have to work on very stiff materials. The force control algorithm should be robust with respect to high values of environment stiffness, and possibly should not require knowledge of the stiffness parameters. On the other hand, it should take account of the high torsional flexibility that commonly affects the joints of industrial manipulators.

4.2 Selection of control algorithms for the RUB action

The execution of the RUB action involves the modelling of contact, the planning of trajectories and contact forces, and their actuation through a proper hybrid position/force controller.

Contact modelling and action planning. It is assumed that the contact between the working tool

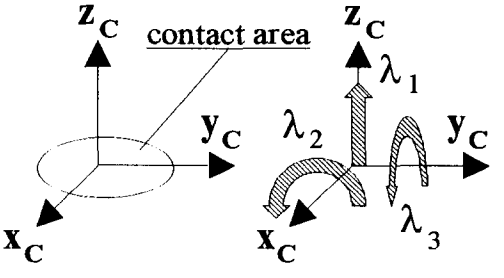

Fig. 3 Contact model

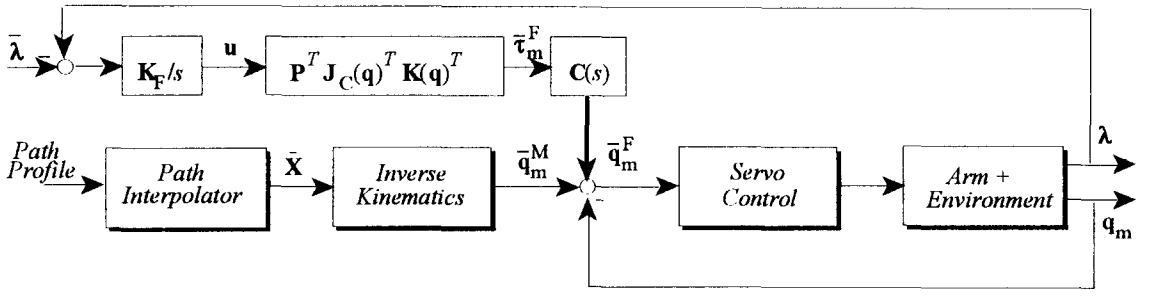

Fig. 4 Position/force control scheme

and the working surface extends to a given plane contact area, and a compliant frame is chosen with its origin in the center point or at a convenient point of the contact area, with the z_C axis directed along the normal to the area. This is sketched in Fig. 3.

If the working surface and the tool are rigid, virtual displacements of the origin of the compliant frame along the normal z_C, and virtual rotations of the compliant frame about axes x_C and y_C are inhibited, while reaction force along z_C and torques about x_C and y_C can be generated. To ensure the proper contact between the working tool and the workpiece it may be necessary to control both the force and the torques.

The user plans the action by defining the contact frame on the end effector, assigning the path of the origin O_C on the working surface, the orientation of axis z_C (parallel to the surface normal) and of axes x_C and y_C (conveniently), and the set points of the contact force and torques. An off-line programming environment might help the introduction of such values in cases of working surfaces of complex shapes.

Hybrid position/force control algorithm. In the scientific literature, several force-control algorithms have been proposed that comply with the presence of position control loops (see for instance (De Schutter, *et al.*, 1988; Duelen, *et al.*, 1992)). The rigid contact task and the joint elasticity, however, are never explicity considered. To fully comply with the requirements stated above, the hybrid position/force control algorithm described in (Ferretti, *et al.*, 1995b), and sketched in Fig.4, is adopted.

Vectors $\bar{\lambda} \in \Re^m$ and $\lambda \in \Re^m$ are contact force set points and measures expressed in the contact frame, respectively, m being the number of degrees of freedom constrained by the contact (m = 3 for the RUB action). Force control is implemented as an outer loop cascaded to the independent motor position loops. It adds a bias $\bar{q}_m^F \in \Re^n$ (n being the number of robot axes) to the nominal motor position set points $\bar{q}_m^M$ and thus it integrates easily in a common functional architecture of an industrial controller, consisting of trajectory planning, kinematics inversion and joint position control. The bias is computed from the force error by means of three blocks: the force regulator, the computation of motor torques, and the torque conversion into motor displacement.

The motor torques $\bar{\tau}_m^F \in \Re^n$ required to actuate a force vector $\mathbf{u} \in \Re^n$ are computed as:

$$\bar{\tau}_m^F = \mathbf{P}^T \mathbf{J}_C(\mathbf{q})^T \mathbf{K}(\mathbf{q})^T \mathbf{u} \; . \qquad (1)$$

Equation (1) is derived from the constrained dynamic model of the manipulator (McClamroch and Wang, 1988) by imposing that $\bar{\tau}_m^F$ controls the interaction forces without disturbing the programmed motion in the unconstrained directions (Ferretti, *et al.*, 1995b). Matrix $\mathbf{P}_{[n\times n]}$ in (1) accounts for the kinematic relations among motor coordinates $\mathbf{q}_m$ and joint coordinates $\mathbf{q}$ (in general, $\mathbf{P}$ is not diagonal, and $P_{ii} = N_i$, N_i being the i-th gear ratio). Matrix $\mathbf{J}_C(\mathbf{q})_{[6\times n]}$ is the manipulator Jacobian, relating joint velocities to the linear and angular velocities of the compliant frame, expressed in the compliant frame itself. In practice, $\mathbf{J}_C(\mathbf{q})$ is computed from the Jacobian $\mathbf{J}(\mathbf{q})$, relating joint and compliant frame velocities expressed in the base frame, and from the robot direct kinematics (rotation matrix). Matrix $\mathbf{K}(\mathbf{q})_{[m\times 6]}$ is a mapping between vector λ and the vector of contact forces and moments $\mathbf{F}_C \in \Re^6$ exerted in the compliant frame, and can be directly derived by analysing the directions of constrained motion (Ferretti, *et al.*, 1995b). For the RUB action, matrix $\mathbf{K}(\mathbf{q})$ is constant, and is given by:

$$\mathbf{K}(\mathbf{q}) = \mathbf{K} = \left[\mathbf{0}_{[3\times 2]} \quad \mathbf{I}_{[3\times 3]} \quad \mathbf{0}_{[3\times 1]}\right].$$

The transfer function diagonal matrix $\mathbf{C}(s)_{[n\times n]}$ is introduced to transform motor torques $\bar{\tau}_m^F$ into corresponding motor rotations $\bar{q}_m^F$. It accounts for position regulator transfer functions and for joint torsional flexibility. It is shown in (Ferretti, *et al.*, 1993), that under reasonable assumptions, $\mathbf{C}(s)$ is a diagonal matrix whose elements $C_{ii}(s)$ are second

order transfer functions whose singularities depend on K_{Pi}, K_{Ii}, K_{Di}, the proportional, integral and derivative gain, respectively, of the i-th PID regulator, and on K_i, the stiffness constant of the i-th joint. K_i can be experimentally identified (Ferretti, *et al.*, 1994ab). In particular, it results that $C_{ii}(0) = 1/K_i$.

The force regulator implements decoupled integral control laws for each force component: matrix $\mathbf{K}_F$ is thus constant and diagonal. This choice is motivated by the fact that the integral regulator, if properly designed, guarantees a crossover frequency of the force-control loop which is compatible with the phase shifts due to delays in the loop and to unmodelled dynamics. This point is confirmed by recent results presented by Ferretti, *et al.* (1995a), which have shown the increasing stability of integral control with increasing contact stiffness.

Finally, the vector of contact force measurements, $\boldsymbol{\lambda}$, can be obtained from the outputs of a wrist-mounted force sensor through a fixed transformation, where the compliant frame is rigidly connected to the end-effector.

4.3 Application architecture for hybrid control

The AA for the algorithm previously discussed is reported as a data flow diagram (DFD) (Lawrence, 1988) in Fig. 5. Actually the DFD has a hierarchical representation, and each function (circle) of Fig. 5 may be represented by another DFD, and so on until a decomposition in sufficiently simple functions is obtained. Functions which cannot be further decomposed are called "leaves" of the overall DFD, and are described by the process specifications (PS). Functions marked with "*" in Fig. 5 should be further decomposed, while those marked with "p" should be described by the PS. Non-filled circles

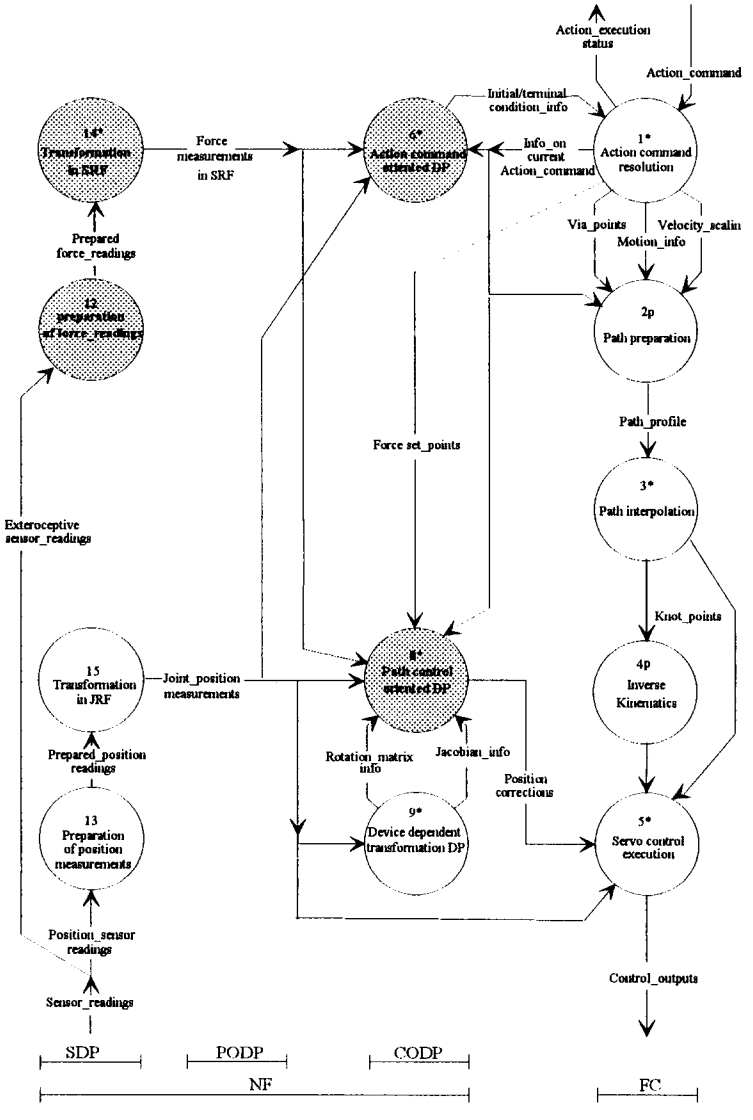

Fig. 5 AA for hybrid control

represent functions already implemented in the standard C3G.

Control functions joined to the right-hand column of circles (1, 2, 3, 4, 5) are those which are relevant to the forward path of the current C3G (see Fig. 4). They include the action command interpretation (circle 1), trajectory computation in joint or Cartesian space as time function polynomials (2), trajectory knot-point computation at each sampling time (3), inverse kinematics (4), microinterpolation and motor position servos (5). It is assumed here that the path is introduced directly from the work program; optionally, it might be stored in external files.

Functions of the circles in the left leg (12, 13, 14, 15) group together the processing of sensory signals. The output of the motor position sensors is filtered (circle 13) and then transformed into joint positions (15); likewise circle 12 is in charge of the low-level processing of force sensor outputs, and circle 14 of yielding force measures in the sensor reference frame (SRF).

Circle 6 groups the computations required to check initial and terminal conditions (e.g. contact established and destination reached, respectively, for RUB). Checks, however, are performed in circle 1. Circle 9 computes the rotation matrix and the Jacobian of the compliant frame with respect to the base frame ($\mathbf{J(q)}$). Circle 8 executes the force-control algorithm to obtain $\mathbf{u}$, $\bar{\boldsymbol{\tau}}_{\mathrm{m}}^{F}$ and $\bar{\mathbf{q}}_{\mathrm{m}}^{F}$. Info_on_current action_command is the command identifier (e.g. *RUB*). A decomposition of the functions of circle 8 is shown in Fig. 6, whose clear rationale is given by the force algorithm scheme of Fig. 4.

Thanks to this modular organization, changing (for instance) the force-control algorithm is reflected in changes to just one or few circle functions, i.e. a small part with sharp boundaries. The same is true for changing the force sensor, and so on.

Impedance control and on-line path-planning algorithms, needed to implement actions like INSERT and FOLLOW, may be implemented using the same circles, introducing the proper functions in each circle. The overall controller AA therefore has the same structure, even if the functions of each circle are actually the logic sum of the functions relevant to each action.

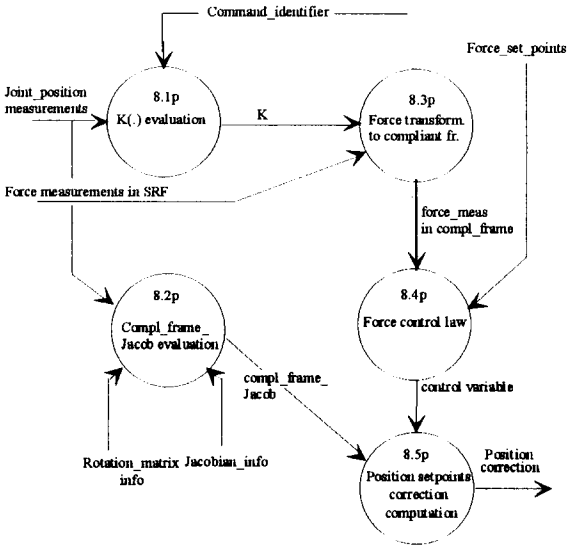

Fig. 6 Decomposition of circle 8 functions (hybrid control algorithm)

5. DESIGN OF THE IMPLEMENTATION ARCHITECTURE

As the final part of the case study, this section shows how the functional architecture proposed above may be implemented by the extension of a commercial controller. The specific case of the Comau C3G 9000 controller is considered. The new controller is named here C4G. The hardware modifications will be considered first, and then a software architecture will be proposed. The illustrative example will then focus on the AA relevant to the RUB action. The proposed solution is, however, also suitable for implementing the other actions defined in Section 3, provided that the relevant additional software modules are developed.

In the choice of a solution, economic constraints are of major concern when mass-produced systems like industrial controllers are considered. Moreover, taking into account that applications requiring sensor-based control will likely remain in a minority for many years, a modular solution featuring exteroceptive sensor-based controls as an add-on item to the standard controller seems the most interesting, and perhaps the only viable, one.

5.1 Hardware architecture

Figure 7 shows that the architecture of the control unit of the C4G is derived from the standard C3G control unit by integrating additional computing and interface boards. The C3G control unit is based on the VME-bus and comprises two processing boards, the robot CPU (RBC) and the servo CPU (SCC).

The RBC board is equipped with Motorola 68020/68882 CPUs, and with a shared memory area that can be accessed from the RBC itself, as well as from other boards on the VME-bus. Tasks running on the RBC accomplish user interface functions, and the translation and interpretation of the user's programs. They also drive the Operator Control panel and the programming terminal. The programming language is PDL2, a Pascal-like language endowed with powerful motion-control instructions. PDL2 supports multi-tasking, communication via LAN and serial links, and analog and digital I/O, so that the flow of program execution can be easily controlled and modified in real time, according to external events.

The SCC is a multiprocessor board equipped with Motorola 68020/68882 CPUs and with Texas 320C25 DSP. The Motorola CPUs perform trajectory generation, in either joint or Cartesian space, and kinematic inversion, both every 10 msec; the DSP executes position set-point micro-interpolation and joint position control, at a sampling interval of 1 msec.

The additional boards are the sensor-based control CPU (SBC) and I/O boards to interface exteroceptive sensors. The SBC is in charge of supplying the additional computing power required to execute the new control functions. It gets sensor data from dedicated I/O boards plugged in the VME bus, and exchanges data with the RBC and SCC through the shared memory on the RBC, and it has to compute the corrections to position set points for hybrid control. The SBC hardware design has not been carried out; a second RBC or SCC would, rather, be used to eliminate the need for the board design and reduce spare-part costs.

5.2 Software architecture

The software modules that implement the control functions relevant to the RUB action on the SBC board are shown in the structural chart in Fig. 8.

Finding the mapping between the functions of the AA for hybrid control and the modules of the software architecture (numbers in italics point to circles in Figs 5 and 6) is straightforward; thus the traceability between the essential functions of the controller and the design solutions is evident.

The main program "Compute_position_corrections" successively activates the modules "Transform_force-readings", "Compute_compl-frame_Jacob", "Compute-contact_force" and "Compute set_point_corrections".

Execution of tasks on SBC and SCC has to be synchronised at each sampling instant, and start and

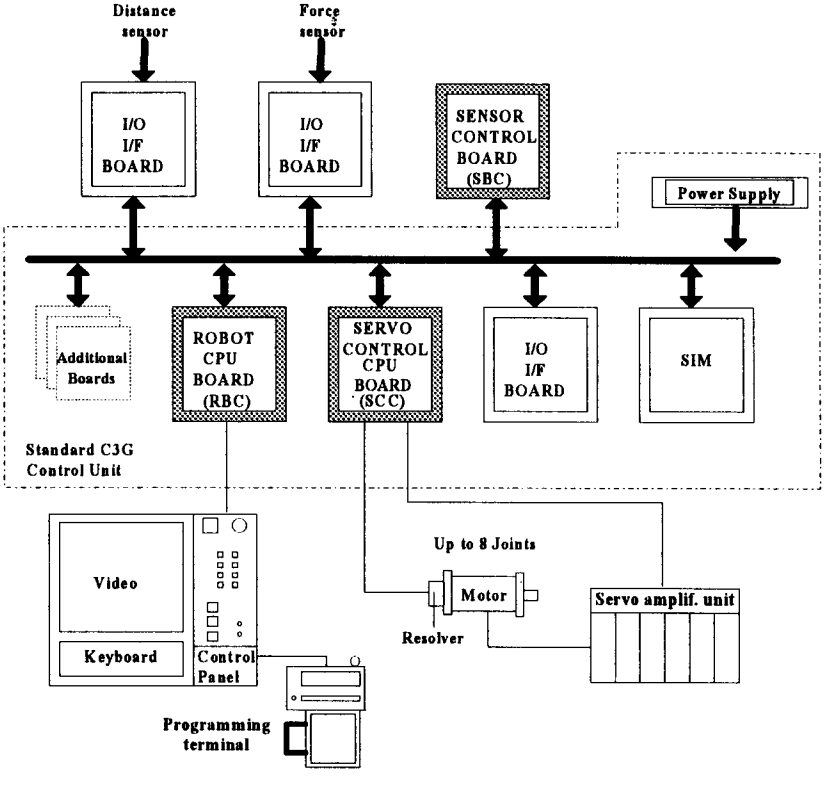

Fig. 7 C4G control unit

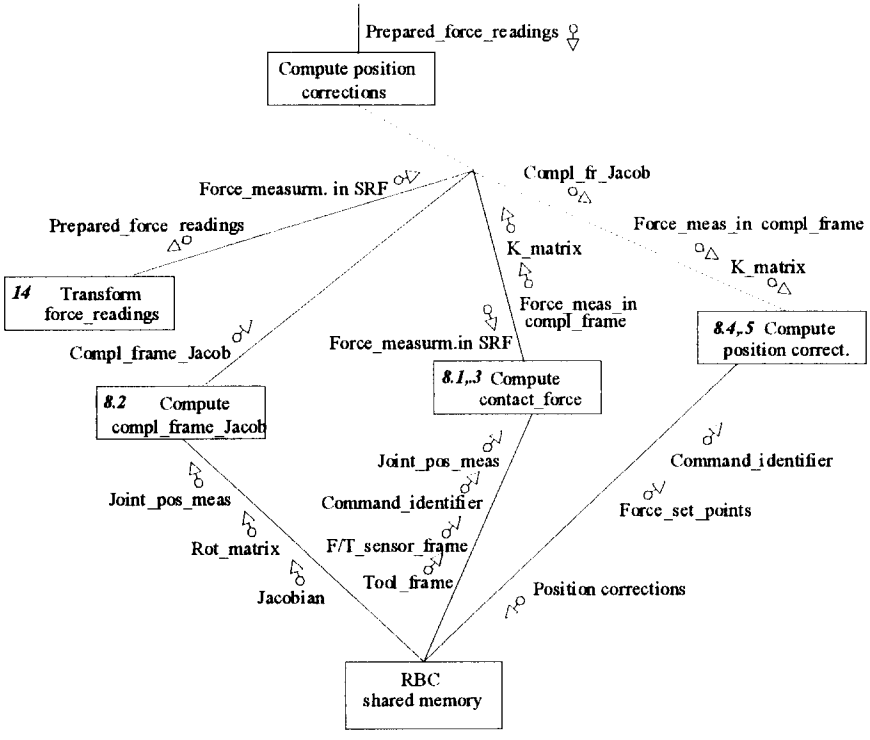

Fig. 8 Structure chart of SBC software for hybrid control

end conditions have to be checked in the SBC. Provided that execution is started synchronously at each sampling instant, tasks on the SCC and SBC may work in parallel to compute the position nominal set point $\bar{\mathbf{q}}_m^M$ and the correction $\bar{\mathbf{q}}_m^F$, respectively. Once $\bar{\mathbf{q}}_m^M$ has been computed, an SCC interface routine gets $\bar{\mathbf{q}}_m^F$ from the shared memory to compute $\bar{\mathbf{q}}_m$ and send it to the DSP for actuation.

5.3 Extension of the PDL 2 programming language for force control

The execution of actions like RUB, INSERT and FOLLOW may be coded as PDL2 routines with minor extensions to the basic instruction and predefined variable sets. Action RUB may be programmed, for instance, by the instruction:

RUB <*trajectory*> *dest_clause force_set* <*opt_clauses*> <*and_clauses*>,

(<, > delimit an optional field) derived directly from the standard PDL2 motion-control command MOVE.

Trajectory may be LINEAR (default) or CIRCULAR, *dest_clause* specifies the final point of the path of the compliant frame (either the value or the pointer to a file containing it), *force_set* contains force set points, *opt_clauses* further specifies the kind of motion and possible termination conditions, *and_clauses* concern the synchronization of two or more arms. The final destination, Cartesian position and orientation, may be specified in many ways, e.g. absolute or relative to the current position, to the approach axis, and so on. Intermediate via (i.e. non-stop) points may also be specified.

The compliant frame may be specified by the predefined variable $TOOL with respect to the built-in fixed flange frame. The new predefined variables, $F/T_SENSOR_FRAME, to specify position and orientation of the sensor frame with respect to the flange frame, and $FORCE_DATA, containing force measurements, should also be introduced, the latter for monitoring (e.g. checking for possible overshoots) and general usage.

6. CONCLUSIONS

The design of an industrial robot controller has been tackled following the guidelines of an established design methodology. A controller suitable for a commercial implementation in the near future has been obtained. The design solutions have been driven by a typical industrial approach, when mass products are considered, that enhances the cost and standardization requirements instead of following the leading edge of technology. From a commercial point of view, it is interesting that the new controller functions may be supplied optionally as an add-on to the base product.

The principles and tools of the structured design adopted here have proved very beneficial. The structures and templates and the catalog of

"common" activities proposed by the CDM provide a systematic and effective way of capturing user needs. The decomposition of activities into tasks and actions is also beneficial for robot programming. In the requirements analysis, the reference functional architecture models enforce a rigorous and thorough analysis of the data flow (interfaces), the support design modularity, the minimality and the re-use of common functions, and permit a clear traceability between user needs and design solutions.

Needless to say, the mere application of structured design concepts does not ensure the quality of the design; that remains dependent on the designers' work and skills, and calls for the adoption of proper quantitative analysis and simulation tools and, if necessary, of suitable experimental work. The full application of the guidelines of this methodology appears profitable, especially with projects of increasing size and complexity, while it may cause too much overhead to smaller projects. The basic concepts and tools of activity analysis and functional design seem effective in either case.

ACKNOWLEDGMENTS

This research has been partly supported by MURST 40%, "Sistemi di controllo per robot operanti in spazi strutturati e non strutturati". The authors acknowledge the contribution and support of S. Deplano (Comau Robotics), A. Elfving (ESTEC), C. Maffezzoni (Politecnico di Milano) and L. Leo (former student at Politecnico di Milano).

REFERENCES

Albus J. S., H.G. McCain, and R. Lumia (1989). NASA/NIST Standard Reference Model for Telerobot Control System Architecture (NASREM), *NIST Technical Note 1235.*

Åström K. J. and B. Wittenmark (1990). *Computer Controlled Systems: Theory and Design*, Prentice Hall

Bone G. M., M.A. Elbestawi, R. Lingkar and L. Liu (1991). Force Control for Robotic Deburring, *ASME J. Dyn. Sys., Meas., Control*, Vol. 113, pp. 395-400.

De Schutter, J. and H. Van Brussel (1988). Compliant Robot Motion. II. A Control Approach Based on External Control Loops, *The International Journal of Robotics Research*, Vol. 7, No. 4, pp. 18-33.

Duelen, G., H. Münch, D. Surdilovic and J. Timm (1992). Automated Schemes for Robotics Deburring: Development and Experimental Evaluation, *IECON '92*, San Diego (USA)

Ferretti G., C. Maffezzoni, G. Magnani and P. Rocco (1993). Decoupling Force and Motion Control in Industrial Robots, *Control Engineering Practice*, Vol. 1, N° 6, pp. 1019-1027.

Ferretti G., C. Maffezzoni, G. Magnani and P. Rocco (1994a). Joint Stiffness Estimation Based on Force Sensor Measurements in Industrial Manipulators, *ASME J. Dyn. Sys., Meas., Control*, Vol. 116, No. 1, pp. 163-167.

Ferretti G., G. Magnani and P.Rocco (1994b). Estimation of Resonant Transfer Functions in the Joints of an Industrial Robot, *2nd IFAC Symp. on Intelligent Control and Control Appl., SICICA '94*, Budapest, pp. 371-376.

Ferretti G., G. Magnani and P. Rocco (1995a). On the Stability of Integral Force Control in Case of Contact with Stiff Surfaces, *ASME J. Dyn. Sys., Meas., Control*, Vol. 117.

Ferretti, G., G. Magnani and P. Rocco (1995b). Towards the Implementation of Hybrid Position / Force Control in Industrial Robots, Internal Report 95.033, Politecnico di Milano, Dipartimento di Elettronica e Informazione

Her, M. and H. Kazerooni (1991). Automated Robotic Deburring of Parts Using Compliance Control, *ASME J. of Dyn. Sys., Meas., and Control*, Vol. 113, pp. 60-66

Lawrence, P. (1988). *Advanced Structured Analysis and Design*, Prentice-Hall

McClamroch, N. H. and D. Wang, (1988). Feedback Stabilization and Tracking of Constrained Robots, *IEEE Transaction on Automatic Control*, Vol. 33, pp. 419-426

Mizugaki, Y., M. Sakamoto and K. Kamijo (1990). Development of a Metal-Mold Polishing Robot System with Contact Pressure Control Using CAD/CAM Data, *Annals of the CIRP*, 39/1,523-526.

Putz P. and A. Elfving (1992). ESA's Control Development Methodology for Space A&R Systems, in *Robotics and Manuf.: Recent Trends in Research, Education, and Appl.* (M. Jamshidi et al., eds.), Vol. 4, pp. 487-492, ASME Press

Putz P. and K.-D. Mau (1992). A&R Control Development Methodology Definition Report, *Doc. Nr. CT2/CDR/DO, Issue 2.0* (ESA Contract 9292/90/NL/JG)

Weule, H. and S. Timmermann (1990). Automation of the Surface Finishing in the Manufacturing of Dies and Molds, *Annals of the CIRP*, 39/1.

Whitney D. E. (1982). Quasi-Static Assembly of Compliantly Supported Rigid Parts, *ASME J. of Dyn. Sys., Meas., and Control*, Vol. 104, pp. 65-77.