Chapter 15

# Compliant motion control

## 15.1. Introduction

Many industrial applications require the contact of the robot end-effector with an uncertain environment. A long list of such applications could be given, including contour following, pushing, polishing, twisting, deburring, grinding, assembling, etc. Implementation of all these tasks intrinsically necessitates that the robot follows the desired path while providing the force necessary either to overcome the resistance from the environment or to comply with it. In order to control force with purely position-based systems, a precise model of the mechanism and knowledge of the exact location and stiffness of the environment are required. High precision robots can be manufactured only at the expense of size, weight and cost. The ability to control the contact forces generated on the end-effector offers an alternative for extending effective precision. A classification of robot force control algorithms includes:

- methods involving the relation between position and applied force: passive stiffness control, active stiffness control;
- methods using the relation between velocity and applied force: impedance control or accommodation control;
- methods using position and force feedback: parallel hybrid position/force control and external hybrid control;
- methods using force feedback: explicit force control;
- methods based on passivity.

In this chapter we will develop the first three methods that constitute the most commonly used. For more details, the reader can refer to [Siciliano 00].

## 15.2. Description of a compliant motion

In pure position control, the user has to completely specify the end-effector position and orientation. This implies that the robot moves in free space. The absence of any contact prevents the exertion of forces. On the other hand, in pure force control the manipulator end-effector is constrained by the environment in all directions; hence, there is no motion at all.

Between the extremes of free space and totally constrained space is the workspace with constraint surfaces, termed *C-surfaces* [Mason 82]. In this case, motion is possible along the C-surface tangents, while force can be exerted along the C-surface normals. Thus, position control and force control exclude themselves mutually: we cannot control a force and a position along the same direction simultaneously. Consequently, compliant tasks require control of the end-effector forces along some directions and its motion along others.

Practically, a compliant task is defined in a frame, called a *compliance frame*, providing six degrees of freedom along and around the frame axes. For every degree of freedom, we specify either a position or a force. According to the task, this frame can be attached to the end-effector, to the environment or to the manipulated object (Figure 15.1).

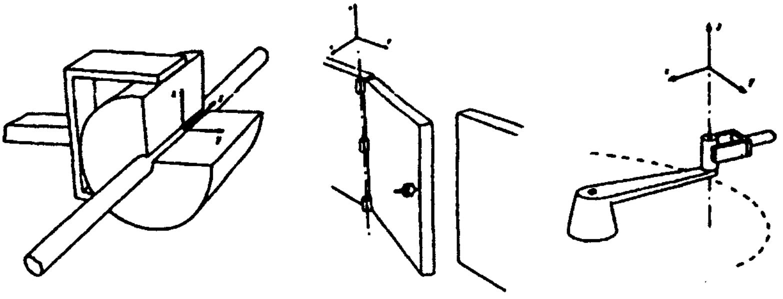


**Figure 15.1.** *Choice of compliance frame according to the task (from [Mason 82])*

## 15.3. Passive stiffness control

Passive stiffness control or passive compliance is a simple solution to reduce the contact forces between the robot and its environment. It consists of interposing between the manipulated part and the robot a mechanical device able to change its configuration under the effect of contact forces, thus adding to the structure an elastic behavior that compensates for positioning errors [Drake 77], [Whitney 79]. Figure 15.2 shows the principle of such a device, the so-called *RCC (Remote*

*Compliance Center)* [Nevins 77], that is typically used to handle peg-in-hole assembly problems. The basic compliance formulation follows from a generalization of the linear spring equation and is given by:

$$\mathbf{dX} = \mathcal{C}\mathbf{f} \qquad [15.1]$$

where $\mathcal{C}$ is the (6x6) compliance matrix, $\mathbf{f} = [\ \mathbf{f}^T\ \ \mathbf{m}^T\ ]^T$ represents the wrench that is composed of a force **f** and a moment **m** (§ 2.6). The differential displacement vector $\mathbf{dX} = [\ \mathbf{dP}^T\ \ \boldsymbol{\delta}^T\ ]^T$ is composed of the differential translation vector **dP** and the differential orientation vector **δ** (§ 2.5).

The compliance matrix $\mathcal{C}$ is diagonal with respect to the compliance frame whose origin $O_c$ is called the *compliance center*: the application of a force at $O_c$ along a given direction causes a pure translation in this direction; the application of a moment causes a pure rotation around an axis passing through $O_c$.

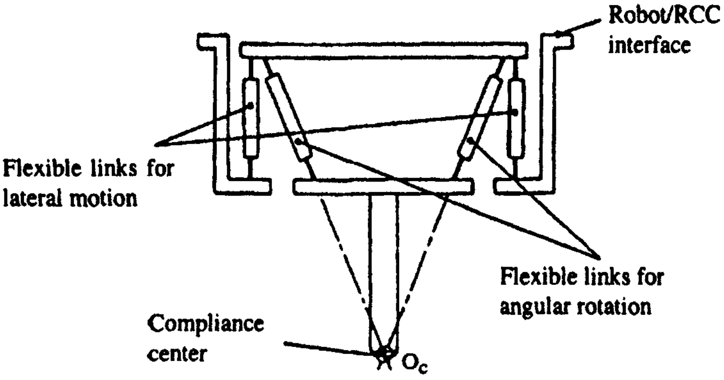

Robot/RCC interface
Flexible links for lateral motion
Flexible links for angular rotation
Compliance center
$O_c$

**Figure 15.2.** *Principle of the RCC device (from [Nevins 77])*

Passive compliance offers some advantages such as fast and accurate insertions of parts without requiring complex strategy (typically, less than 0.2 sec. for tolerance of the order of 1/100 mm). It has achieved success in specific assembly tasks, for example inserting a peg in a hole. The limitation is that each compliant device is devoted to a given task and to a given workpiece.

## 15.4. Active stiffness control

This method actively controls the apparent stiffness of the robot end-effector and allows simultaneous position and force control. The user can specify the three translational and three rotational stiffnesses of a desired compliance frame. Stiffness may be changed under program control to match varying task requirements [Salisbury 80]. High gain is assigned to the directions that have to be position controlled, while low gain is assigned to the force controlled directions. The basic stiffness formulation is given by:

$$^{c}\mathbb{f}_{c} = \mathbf{K}_{c}\, {}^{c}\mathbf{dX}_{c} \qquad [15.2]$$

where $\mathbf{K}_c$ is the desired (6x6) stiffness matrix, which is diagonal in frame $\mathbf{R}_c$. The wrench $^{c}\mathbb{f}_{c}$ and the differential displacement $^{c}\mathbf{dX}_{c}$ are expressed in frame $\mathbf{R}_c$, and will be simply denoted by $\mathbb{f}$ and $\mathbf{dX}$ respectively. Assuming that the friction and dynamic forces are compensated or are small enough to be neglected, equation [5.43] gives the joint torque $\mathbf{\Gamma}$ necessary to apply a wrench $\mathbb{f}$:

$$\mathbf{\Gamma} = \mathbf{J}^{T}\, \mathbb{f} \qquad [15.3]$$

Let us recall the differential model [5.2]:

$$\mathbf{dX} = \mathbf{J}\, \mathbf{dq} \qquad [15.4]$$

where $\mathbf{J}$ is the Jacobian matrix of the robot describing the differential translational and rotational vectors of the compliance frame as a function of the differential variations of joint positions $\mathbf{dq}$. It should be noted that the Jacobian may be computed for any point fixed in the end-effector frame. Combining equations [15.2], [15.3] and [15.4], we obtain:

$$\mathbf{\Gamma} = \mathbf{J}^{T}\mathbf{K}_{c}\,\mathbf{J}\,\mathbf{dq} = \mathbf{K}_{q}\,\mathbf{dq} \qquad [15.5]$$

The matrix $\mathbf{K}_q$ is called the *joint stiffness matrix* and is not diagonal but symmetric. It determines the proportional gains of the servo loops in the joint space. It presents the same singular positions as the Jacobian matrix of the kinematic model, which means that, for these configurations, we cannot get the desired stiffness along or about all the degrees of freedom of the compliance frame. The principle of this control scheme is shown in Figure 15.3. The joint torque vector is given by:

$$\mathbf{\Gamma} = \mathbf{K}_{q}\,(\mathbf{q}^{d} - \mathbf{q}) + \mathbf{K}_{d}\,(\dot{\mathbf{q}}^{d} - \dot{\mathbf{q}}) + \mathbf{Q} \qquad [15.6]$$

where $\mathbf{Q}$ represents gravity torque compensation, and $\mathbf{K}_d$ can be interpreted as a damping matrix. A feedforward force term can be added if pure force control is desired in some direction. It is computed using equation [15.3].

The advantage of such an active stiffness control scheme is that it is relatively simple to implement. The stiffness matrix can be changed on-line to adapt the robot behavior to various task constraints.

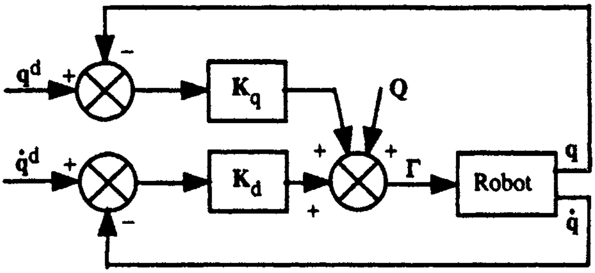


**Figure 15.3.** *Principle of the active stiffness control scheme*

## 15.5. Impedance control

According to Hogan [Hogan 85], [Hogan 87], the basic idea of impedance control is to assign a prescribed dynamic behaviour for the robot while its effector is interacting with the environment. The desired performance is specified by a generalized dynamic impedance representing a mass-spring-damper system.

The end-effector velocity $\dot{\mathbf{X}}$ and the applied force are related by a mechanical impedance **Z**. In the frequency domain, this is represented by:

$$\mathbf{F}(s) = \mathbf{Z}(s)\,\dot{\mathbf{X}}(s) \tag{15.7a}$$

In terms of position **X**(s), we can write:

$$\mathbf{F}(s) = s\,\mathbf{Z}(s)\,\mathbf{X}(s) \tag{15.7b}$$

The robot should behave like a mechanical system whose impedance **Z** is variable according to the different phases of the task. In general, we suppose that the robot is equivalent to a mass-spring-damper second order system, whose transfer function is:

$$s\,\mathbf{Z}(s) = \mathbf{\Lambda}\,s^2 + \mathbf{B}\,s + \mathbf{K} \tag{15.8}$$

where **Λ**, **B** and **K** represent the desired inertia, damping and stiffness matrices respectively. The values of these matrices are chosen to obtain the desired performance:

- high values are given to $\Lambda$ in the directions where a contact is expected in order to limit the dynamics of the robot;
- high values are given to **B** in the directions where it is necessary to dissipate the kinetic energy and therefore to damp the response;
- the stiffness **K** affects the accuracy of the position control: along the force controlled directions, the stiffness should be small enough to limit the contact forces; conversely, along the position controlled directions, the user should set a high stiffness to obtain an accurate positioning of the end-effector.

Two families of control schemes can be implemented depending on whether or not a force sensor is available (Figures 15.4 and 15.5).

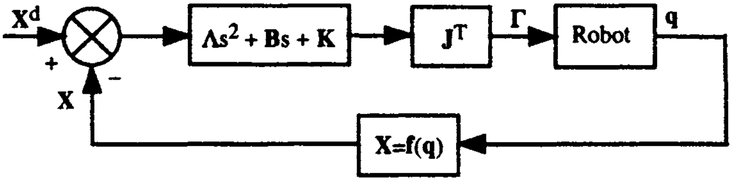
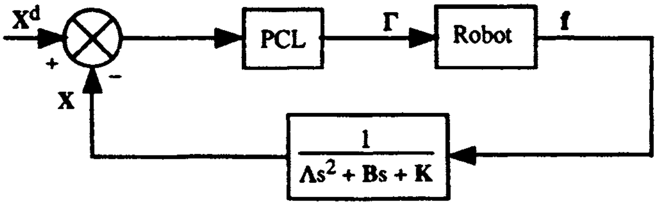


**Figure 15.4.** *Impedance control scheme without force feedback*



*PCL: Position Control Law*

**Figure 15.5.** *Impedance control scheme with force feedback*

Figure 15.6 shows an implementation of the impedance control scheme of the first family. The dynamics of the robot is neglected. The control law is given by:

$$\Gamma = J^T [B (\dot{X}^d - \dot{X}) + K (X^d - X)] + Q \qquad [15.9]$$

The **K** and **B** matrices contain the proportional and derivative gains in the task space, which can be interpreted as the stiffness matrix and the damping matrix of the robot respectively. As previously, the vector **Q** represents gravity torque compensation. This control scheme places the compliance center at the desired point $X^d$. It is equivalent to the PD control in the task space (§ 14.3.3).

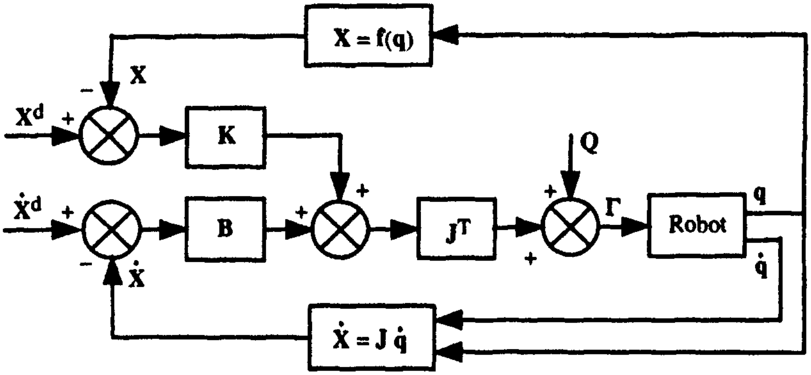


**Figure 15.6.** *Impedance control scheme without force sensor feedback and with a PD control law in the task space*

In the following, we present two forms of impedance control representative of the second family, using the dynamic model of the robot in the joint space, then in the task space.

Let us note that the dynamic model of a robot exerting a wrench $\mathfrak{f}$ on its environment is written as (Chapter 9):

$$\Gamma = A(q)\,\ddot{q} + C(q, \dot{q})\,\dot{q} + Q(q) + J^T\,\mathfrak{f} \qquad [15.10]$$

The desired behavior is deduced from equation [15.7b] as:

$$\mathfrak{f} = \Lambda\,(\ddot{X}^d - \ddot{X}) + B\,(\dot{X}^d - \dot{X}) + K\,(X^d - X) \qquad [15.11]$$

which leads to:

$$\ddot{X}(t) = \ddot{X}^d + \Lambda^{-1}\,[B\,(\dot{X}^d - \dot{X}) + K\,(X^d - X) - \mathfrak{f}] \qquad [15.12]$$

where $X^d(t)$ is the desired trajectory.

To achieve this impedance control scheme, let us consider the decoupling nonlinear control law in the task space of equation [14.33] (resolved acceleration control law) in which w(t) is replaced by equation [15.12], and the external wrench exerted by the robot on the environment $J^T\,\mathfrak{f}$ is taken into account:

$$\Gamma = \hat{A} J^{-1}\{\ddot{X}^d + \Lambda^{-1}[B(\dot{X}^d - \dot{X}) + K(X^d - X) - \mathfrak{f}] - \dot{J}\,\dot{q}\} + \hat{H}(q, \dot{q}) + J^T\mathfrak{f} \qquad [15.13]$$

Another formulation of this control law can be derived from the dynamic model in the task space [Zodiac 96]. Combining the first and second order kinematic models (§ 5.10) with the dynamic equation [15.10], yields:

$$J^{-T}\,\Gamma = A_x(q)\,\ddot{X} + C_x(q, \dot{q})\,\dot{X} + Q_x(q) + \mathfrak{f} \qquad [15.14]$$

where:

- $J^{-T}$ is the inverse of $J^T$;
- $A_x(q)$ is the inertia matrix in the task space[1] equal to $J^{-T}\,A(q)\,J^{-1}$;
- $C_x(q, \dot{q})$ is the vector of Coriolis and centrifugal torques in the task space. It is equal to $[J^{-T}\,C(q, \dot{q})\,J^{-1} - A_x(q)\,\dot{J}\,J^{-1}]$;
- $Q_x(q) = J^{-T}\,Q(q)$ is the vector of gravity torques in the task space.

We obtain the decoupled control law as indicated in § 14.4:

$$\Gamma = J^T[\hat{A}_x(q)\,w(t) + \hat{C}_x(q, \dot{q})\,\dot{X} + \hat{Q}_x(q) + \mathfrak{f}] \qquad [15.15a]$$

Replacing w(t) by $\ddot{X}(t)$, as given by equation [15.12], leads to:

$$\Gamma = J^T\,\hat{A}_x(q)\,\{\ddot{X}^d + \Lambda^{-1}[B(\dot{X}^d - \dot{X}) + K(X^d - X)]\} + $$
$$J^T\,[\hat{C}_x(q, \dot{q})\,\dot{X} + \hat{Q}_x(q) + (I - \hat{A}_x(q)\,\Lambda^{-1})\,\mathfrak{f}] \qquad [15.15b]$$

This control scheme is represented in Figure 15.7. It is equivalent to the control scheme [15.13], which is easier to implement when the complete control law must be computed. The algorithm is similar to that presented in Appendix 10. The control scheme [15.15] is preferred in quasi-static cases, where the inertia matrix and the Jacobian matrix are roughly constant [Kazerooni 86]. Thus, $\dot{J}$ and $\hat{C}_x(q, \dot{q})\,\dot{X}$ are considered to be equal to zero, and equation [15.15b] becomes:

$$\Gamma = J^T\,\hat{A}_x\,\Lambda^{-1}[\Lambda\,\ddot{X}^d + B(\dot{X}^d - \dot{X}) + K(X^d - X) - \mathfrak{f}] + Q(q) + J^T\,\mathfrak{f} \qquad [15.16]$$

[1] The reader can find in [Lilly 90] an efficient algorithm for computing $A_x$ without computing the inertia matrix.

Note that the wrench $\mathbf{f}$ appears twice in this equation: once with the term $\mathbf{J}^T \mathbf{f}$ that compensates for the external wrench exerted by the robot and once with the term $-\mathbf{J}^T \hat{\mathbf{A}}_x \Lambda^{-1} \mathbf{f}$, which represents a force feedback whose gain is $-\mathbf{J}^T \hat{\mathbf{A}}_x \Lambda^{-1}$. Besides, note that if $\Lambda = \mathbf{A}_x(\mathbf{q})$, the terms containing $\mathbf{f}$ vanish, which yields to the decoupled control law shown in Figure 15.7:

$$\Gamma = \hat{\mathbf{A}}(\mathbf{q})\, \mathbf{J}^{-1}[\ddot{\mathbf{X}}^d + \Lambda^{-1}[\mathbf{B}(\dot{\mathbf{X}}^d - \dot{\mathbf{X}}) + \mathbf{K}(\mathbf{X}^d - \mathbf{X})] - \dot{\mathbf{J}}\, \dot{\mathbf{q}}] + \hat{\mathbf{H}}(\mathbf{q}, \dot{\mathbf{q}}) \qquad [15.17]$$

NOTES.–

- a long time before the formulation of the impedance control by Hogan, particular cases of this control had been proposed in the literature such as those based on a stiffness matrix or on a damping matrix [Whitney 85]. In the former, $\Lambda = \mathbf{0}$ and $\mathbf{B} = \mathbf{0}$; in the latter, $\Lambda = \mathbf{0}$ and $\mathbf{K} = \mathbf{0}$;
- the active stiffness control proposed by Salisbury (§ 15.2.2) is also a particular case of impedance control where $\Lambda = \mathbf{0}$ and $\mathbf{B} = \mathbf{0}$;
- impedance control is similar to resolved acceleration control with the only difference being the inclusion of the desired inertia in the force gain.

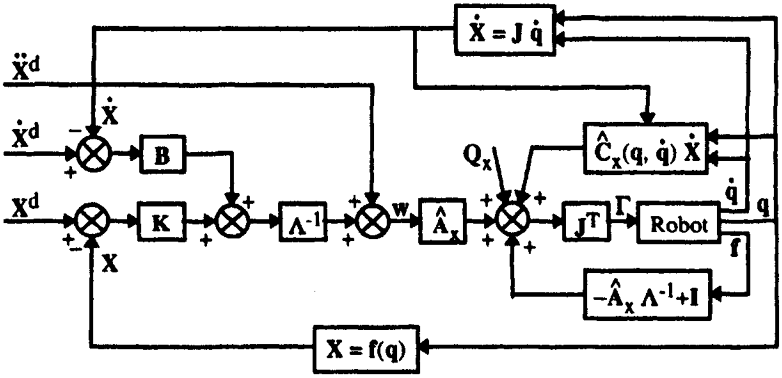


**Figure 15.7.** *Nonlinear decoupling impedance control without force feedback*

## 15.6. Hybrid position/force control

Using the previous methods, we can specify a desired dynamic behavior of the robot but we cannot prescribe a desired wrench. In the following, we address some methods where both position and force can be specified. Much work has been carried out on this topic such as that of [Nevins 73], [Reboulet 85], [Merlet 86], [Robert 86], [Perdereau 91], [Dégoulange 93], [Morel 94], etc. Two families of control schemes with force control loops are introduced: parallel hybrid position/force control and external hybrid control.

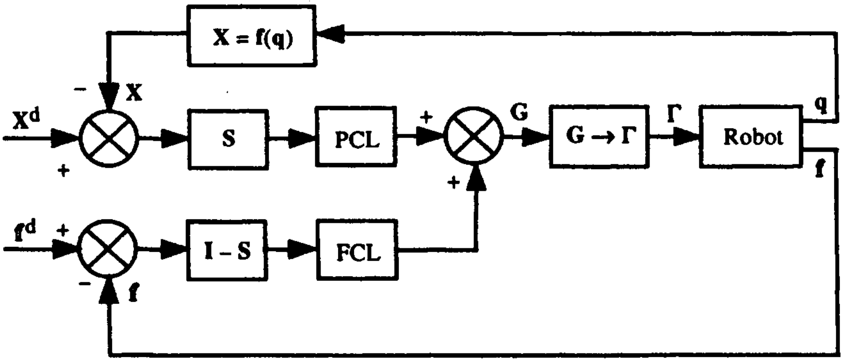
### 15.6.1. *Parallel hybrid position/force control*

The parallel hybrid position/force control finds its roots in the work of Raibert and Craig [Raibert 81]. It satisfies simultaneously the desired position and force constraints of the task. Positions and forces are specified according to the Mason formulation: directions that are constrained in position are force controlled, while those that are constrained in force (zero force) are position or velocity controlled. Duffy [Duffy 90] has shown that it is not correct to consider the velocity subspace and the force subspace as orthogonal as suggested in [Raibert 81]. Rather, it is the position or velocity controlled directions and the force controlled directions that have to be orthogonal in the compliance frame.

In the parallel hybrid control method, the robot is controlled by two complementary feedback loops, one for the position, the other for the force. Each has its own sensory system and control law. The control laws of both loops are added before being sent to the actuator as a global control signal G (Figure 15.8). Each degree of freedom of the compliant frame is controlled by the position or force loop through the use of a *compliance selection matrix* S, which is diagonal such that:

$$\mathbf{S} = \mathbf{diag}(s_1, s_2, \ldots, s_6) \qquad [15.18]$$

where $s_j = 1$ if the $j^{th}$ degree of freedom of the compliance frame is position controlled or $s_j = 0$ if it is force controlled.



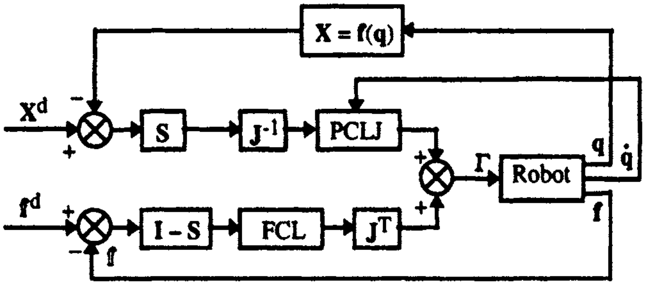
*PCL: Position Control Law; FCL: Force Control Law*

**Figure 15.8.** *Principle of the hybrid position/force control*

Since both loops act cooperatively, each joint contributes to the realization of both the position control and the force control.

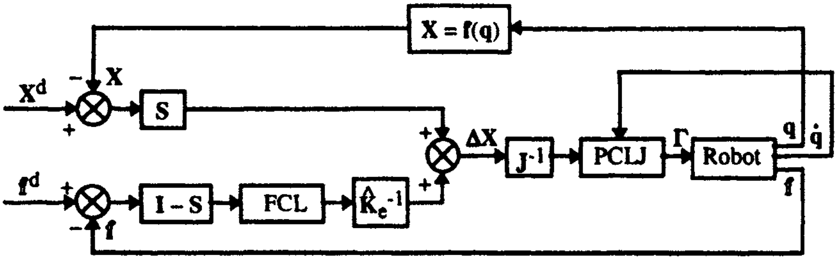
Three forms of hybrid control schemes can be distinguished according to the type of the global control signal G:

- G is equivalent to joint torques (Figure 15.9);
- G is equivalent to displacements or velocities in the task space and has to be multiplied by the robot inverse Jacobian to obtain joint positions (Figure 15.10);
- G is equivalent to forces in the task space and has to be multiplied by the transpose of the Jacobian matrix (Figure 15.11).



*PCLJ: Position Control Law in the Joint space; FCL: Force Control Law*

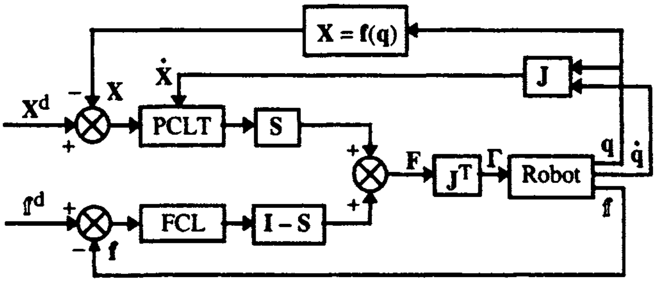
**Figure 15.9.** *Hybrid force-position control scheme with addition of joint torques (from [Raibert 81])*



*PCLJ: Position Control Law in the Joint space; FCL: Force Control Law*

$\hat{K}_e^{-1}$*: estimate of the stiffness matrix of the environment*

**Figure 15.10.** *Hybrid force-position control scheme with addition of velocities*



*PCLT: Position Control Law in the Task space; FCL: Force Control Law*

**Figure 15.11.** *Hybrid force-position control scheme with addition of task forces*

In these figures, the frame transformation computations for velocities, forces and for the Jacobian matrix are not indicated. Practically, the matrices S and (I − S) are applied to signals expressed in the compliance frame. For position control in the joint space (Figures 15.9 and 15.10), we can use one of the laws presented in Chapter 14, for example the PID controller of equation [14.5], which is:

$$\Gamma = K_p (q^d - q) + K_d (\dot{q}^d - \dot{q}) + K_I \int_{t0}^{t} (q^d - q)\, d\tau \qquad [15.19]$$

whereas for a PID control in the task space (Figure 15.11), we have (equation [14.17]):

$$\Gamma = J^T [K_p (X^d - X) + K_d (\dot{X}^d - \dot{X}) + K_I \int_{t0}^{t} (X^d - X)\, d\tau] \qquad [15.20]$$

Normally, the force control law is chosen as:

$$\Gamma = J^T [f^d + K_f (f^d - f) - K_{fd} \dot{X} + K_{fI} \int_{t0}^{t} (f^d - f)\, d\tau] \qquad [15.21]$$

Note that, due to the noise of force sensors, the velocity in the task space is used with the derivative gain rather than the derivative of the force.

In these schemes, we can also include feedforward compensation for the nonlinear dynamics of the robot. For example, the position loop of the hybrid control of Figure 15.11 may be realized by the nonlinear decoupling control law in the task space described in § 14.4.3. The corresponding block diagram is given in

Figure 15.12 [Khatib 87]. The computation of the control vector $\Gamma$ can be achieved with the Newton-Euler algorithm in a similar way to that described in Appendix 10, *with the following arguments:*

- the joint position is equal to the current joint position $\mathbf{q}$;
- the joint velocity is equal to the current joint velocity $\dot{\mathbf{q}}$;
- the joint acceleration is equal to:

$$\ddot{\mathbf{q}} = \mathbf{J}^{-1}\,\mathbf{S}\,[\ddot{\mathbf{X}}^d + \mathbf{K}_d(\dot{\mathbf{X}}^d - \dot{\mathbf{X}}) + \mathbf{K}_p(\mathbf{X}^d - \mathbf{X}) - \dot{\mathbf{J}}\,\dot{\mathbf{q}}] \qquad [15.22]$$

- the force exerted by the terminal link on the environment can be taken to be equal to:

$$\mathbf{f}_{en} = (\mathbf{I} - \mathbf{S})\,[\mathbf{f}^d + \mathbf{K}_f(\mathbf{f}^d - \mathbf{f}) - \mathbf{K}_{fd}\,\dot{\mathbf{X}} + \mathbf{K}_{fI}\int_{t0}^{t}(\mathbf{f}^d - \mathbf{f})\,d\tau] \qquad [15.23]$$

where all terms are computed in the compliance frame $R_c$.

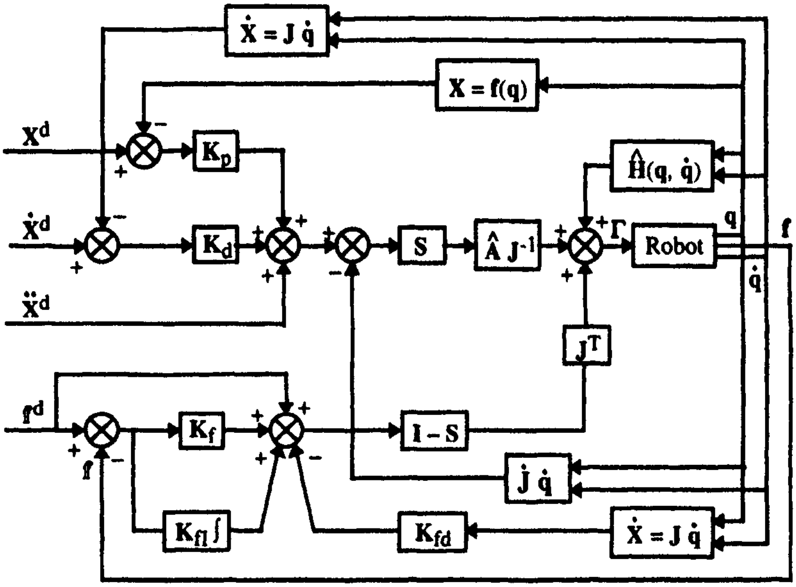


**Figure 15.12.** *Implementation of the dynamic hybrid position-force control scheme*

An and Hollerbach [An 87] showed that the control schemes of Figures 15.9 and 15.10, which require the inverse of the Jacobian matrix, have an unstable behavior when implemented on a robot with revolute joints, even in non-singular configurations. They assigned this instability to an interaction between the inertia matrix and the inverse kinematic model $\mathbf{J}^{-1}$, whereas the scheme using $\mathbf{J}^{T}$ (Figure 15.11) always produces stable results. Fisher and Mutjaba [Fisher 92] showed that this instability comes from the formulation of the inverse kinematic model in the position loop of the hybrid scheme. Using the selection matrix **S** to separate position and force requirements in the task space is conceptually straightforward. Geometrically, the selection matrix is a projection that reduces the task space to a desired subspace of interest. Problems may arise when this selected task subspace is mapped onto the joint space using the robot Jacobian matrix. From the classical scheme of Figure 15.9 and equation [5.2], we can write that:

$$\mathbf{S}\,\mathbf{dX} = (\mathbf{SJ})\,\mathbf{dq} \qquad [15.24]$$

From this equation, it can be seen that the selection matrix **S** reduces the task space of the robot, which becomes redundant with respect to the displacement task. Thus, instabilities of the hybrid control scheme of Craig and Raibert are the consequence of an erroneous formulation of the projection of the task error vector into the joint space. In fact, knowing that $(\mathbf{SJ})^{+}\,\mathbf{S} = (\mathbf{SJ})^{+}$, the general solution of [15.24] is:

$$\mathbf{dq} = (\mathbf{SJ})^{+}\,\mathbf{dX} + [\mathbf{I} - (\mathbf{SJ})^{+}\,(\mathbf{SJ})]\,\mathbf{Z} \qquad [15.25]$$

Fisher and Mutjaba [Fisher 92] showed that choosing $\mathbf{Z} = \mathbf{J}^{-1}\,\mathbf{S}\,\mathbf{dX}$ as the optimization term in equation [15.25] is equivalent to the inverse kinematic relation $\mathbf{dq} = \mathbf{J}^{-1}\,\mathbf{S}\,\mathbf{dX}$. This choice of **Z** does not ensure stability and explains the instabilities that can appear with the hybrid control scheme. Indeed, they showed that the first term of equation [15.25], which is the minimal norm solution, is always stable. Consequently, as indicated in Figure 15.13, the position loop reference input should be:

$$\mathbf{dq} = (\mathbf{SJ})^{+}\,\mathbf{dX} \qquad [15.26a]$$

In a similar manner, they showed that the force loop reference input could remain as the original one:

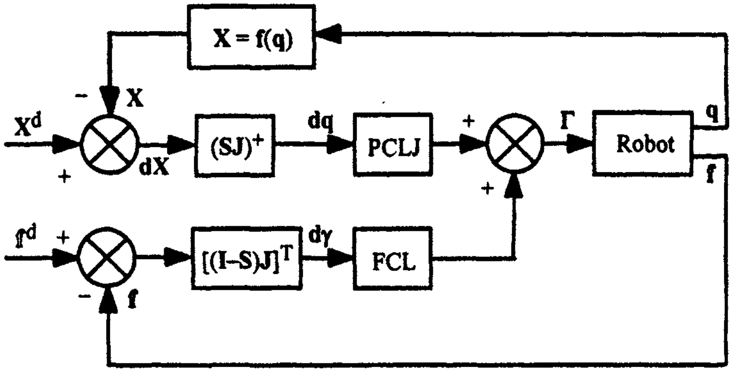
$$\mathbf{d}\boldsymbol{\gamma} = [(\mathbf{I} - \mathbf{S})\,\mathbf{J}]^{T}\,(\mathbf{f}^{d} - \mathbf{f}) \qquad [15.26b]$$

More general solutions with optimization terms are:

$$\mathbf{dq} = (\mathbf{SJ})^{+}\,\mathbf{dX} + [\mathbf{I} - \mathbf{J}^{+}\,\mathbf{J}]\,\mathbf{Z}_{q} \qquad [15.27a]$$

$$\mathbf{d}\boldsymbol{\gamma} = [(\mathbf{I} - \mathbf{S})\,\mathbf{J}]^{T}\,(\mathbf{f}^{d} - \mathbf{f}) + [\mathbf{I} - \mathbf{J}^{+}\,\mathbf{J}]\,\mathbf{Z}_{f} \qquad [15.27b]$$

where $Z_q$ and $Z_f$ are arbitrary position and force vectors in the joint space respectively.



*PCLJ: Position Control Law in the Joint space; FCL: Force Control Law*

**Figure 15.13.** *Hybrid force-position control scheme (from [Fisher 92])*

### 15.6.2. *External hybrid control scheme*

The external hybrid control scheme is composed of two embedded control loops [De Schutter 88], [Perdereau 91]: the outer loop controls force while the inner one controls position (Figure 15.14). The output of the outer loop is transformed into a desired position input for the inner loop. The resulting displacement of the robot permits exertion of the desired contact force on the environment. The external hybrid control scheme is relatively easy to implement and requires a rather small amount of computation. It can be implemented in industrial robots while keeping their conventional controllers [Thérond 96].

The position control loop can be achieved either in the task space or in the joint space by implementing one of the methods presented in Chapter 14.

The additional displacement reference signal is given by:

$$dX_f = \hat{K}_e^{-1} \left[f^d + K_f(f^d - f) + K_{fI} \int_{t0}^{t} (f^d - f)\, d\tau\right] \qquad [15.28]$$

where $\hat{K}_e$ is an estimate of the stiffness of the environment.

Thanks to the integral force action, the wrench error $(f^d - f)$ is allowed to prevail over the position error $(X^d - X)$ at steady state [Pujas 95].

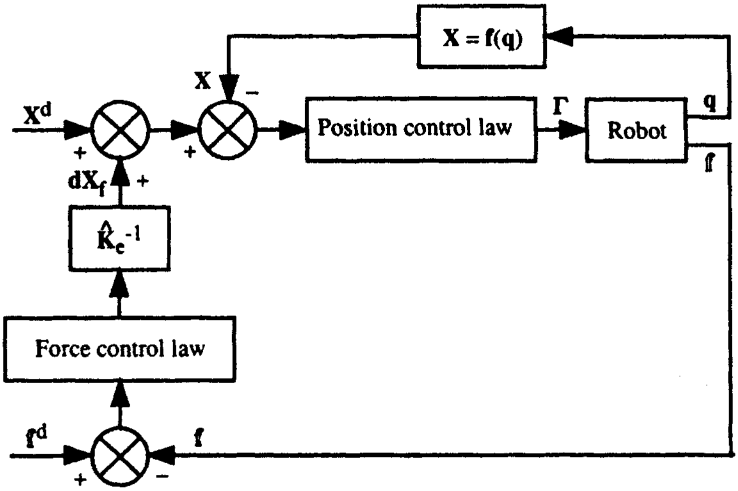


**Figure 15.14.** *Principle of the external hybrid control scheme*

This control can be applied with a nonlinear decoupling impedance control in the task space [Chiaverini 93] by setting w(t) as the sum of two terms, $w_X(t)$ and $w_F(t)$, which are the contributions of the position loop and the force loop respectively:

$$w(t) = w_X(t) + w_F(t) \qquad [15.29]$$

$$w_X(t) = \ddot{X}^d + \Lambda^{-1}[K_d(\dot{X}^d - \dot{X}) + K_p(X^d - X)] \qquad [15.30]$$

$$w_F(t) = \Lambda^{-1}[K_f(\mathbb{f}^d - \mathbb{f}) + K_{fI}\int_{t0}^{t}(\mathbb{f}^d - \mathbb{f})\,d\tau] \qquad [15.31]$$

Note that $w_F(t)$ is obtained by multiplying the force signal by $\Lambda^{-1}$ because it is equivalent to an acceleration. The decoupled control law is obtained from equation [15.15] as:

$$\Gamma = J^T[\hat{A}_x(q)(w_X + w_F) + \hat{C}_x(q, \dot{q})\dot{X} + \hat{Q}_x(q) + \mathbb{f}] \qquad [15.32]$$

If the robot dynamic model is perfectly known, combining equations [15.30], [15.31] and [15.32] yields:

$$\Lambda(\ddot{X}^d - \ddot{X}) + K_d(\dot{X}^d - \dot{X}) + K_p(X^d - X) + K_f(\mathbb{f}^d - \mathbb{f}) + K_{fI}\int_{t0}^{t}(\mathbb{f}^d - \mathbb{f})\,d\tau = 0 \qquad [15.33]$$

When $\mathbf{K}_f = \mathbf{0}$ and $\mathbf{K}_{fI} = \mathbf{0}$, the control law [15.32] becomes equivalent to the impedance control (Figure 15.7). Besides, if $\Lambda = \mathbf{I}$, it reduces to the decoupling nonlinear control in the task space such as that shown in Figure 14.5.

## 15.7. Conclusion

In this chapter, we have presented the most popular position/force control approaches. For other methods, the reader should refer to [Brogliato 91], [Siciliano 96a], for the passive force control, to [Siciliano 93], [Colbaugh 93], [Arimoto 93], [Siciliano 96b] for the adaptive force control or to [Volpe 93], [Volpe 95] for explicit force control.

We did not address the stability problem of force control and the interested reader should refer to [Wen 91], [Yabuta 92], [Wang 93], [Zodiac 96], [Siciliano 00].

The problem of exerting a force on a moving target, thus of controlling simultaneously force and velocity along the same direction, is addressed in the work of [de Luca 91b].

Among the yet open problems, we have to mention the control of impact when the robot and the environment enter in contact. Another class of problem concerns the programming of compliant tasks: the choice of the axes of the compliance frame and their roles requires from the user a lot of experience and is much more difficult, in terms of abstraction capabilities, than programming displacements. Besides, some physical parameters, like the stiffness of the robot and the environment, are not easy to quantify, which results in instability problems.