Chapter 7
Robot control

The problem of robot control can be explained as a computation of the forces or torques which must be generated by the actuators in order to successfully accomplish the robot task. The appropriate working conditions must be ensured both during the transient period as well as in the stationary state. The robot task can be presented either as the execution of the motions in a free space, where position control is performed, or in contact with the environment, where control of the contact force is required. First, we shall study the position control of a robot mechanism which is not in contact with its environment. Then, in the further text we shall upgrade the position control with the force control.

The problem of robot control is not unique. There exist various methods which differ in their complexity and in the effectiveness of robot actions. The choice of the control method depends on the robot task. An important difference is, for example, between the task where the robot end-effector must accurately follow the prescribed trajectory (e.g. laser welding) and another task where it is only required that the robot end-effector reaches the desired final pose, while the details of the trajectory between the initial and the final point are not important (e.g. palletizing). The mechanical structure of the robot mechanism also influences the selection of the appropriate control method. The control of a cartesian robot manipulator in general differs from the control of an anthropomorphic robot.

Robot control usually takes place in the world coordinate frame, which is defined by the user and is called also the coordinate frame of the robot task. Instead of world coordinate frame we often use a shorter expression, namely external coordinates. We are predominantly interested in the pose of the robot end-effector expressed in the external coordinates and rarely in the joint positions, which are also called internal coordinates. Nevertheless, we must be aware that in all cases we directly control the internal coordinates i.e. joint angles or displacements. The end-effector pose is only controlled indirectly. It is determined by the kinematic model of the robot mechanism and the given values of the internal coordinates.

Figure 7.1 shows a general robot control system. The input to the control system is the desired pose of the robot end-effector, which is obtained by using trajectory interpolation methods, introduced in the previous chapter. The variable $\mathbf{x}_r$ represents

T. Bajd et al., *Robotics*, Intelligent Systems, Control and Automation: Science and Engineering 43, DOI 10.1007/978-90-481-3776-3_7,

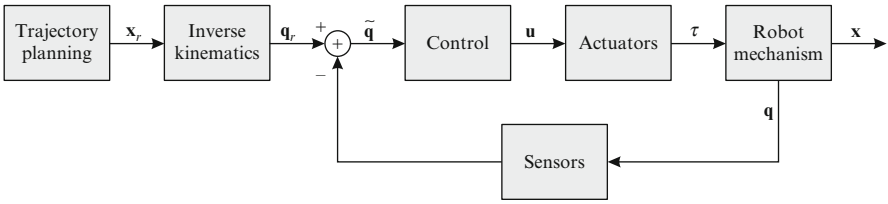

Fig. 7.1 A general robot control system

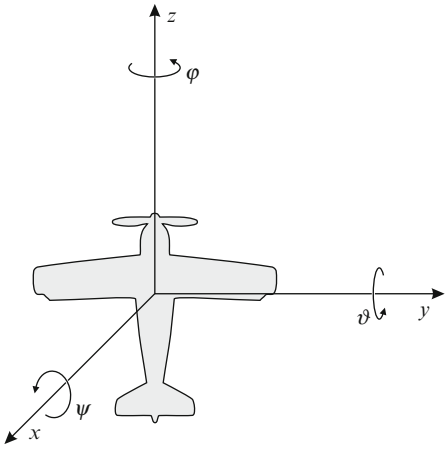

Fig. 7.2 The RPY description of the orientation

the desired, i.e. the reference pose of the robot end-effector. The $\mathbf{x}$ vector, describing the actual pose of the robot end-effector in general comprises six variables. Three of them define the position of the robot end-point, while the other three determine the orientation of the robot end-effector. Thus, we write $\mathbf{x} = \left[x\ y\ z\ \varphi\ \vartheta\ \psi\right]^T$. The position of the robot end-effector is determined by the vector from the origin of the world coordinate frame to the robot end-point. The orientation of the end-effector can be presented in various ways. One of the possible descriptions is the so called RPY notation, arising from aeronautics and shown in Figure 7.2. The orientation is determined by the angle φ around the z axis (Roll), the angle ϑ around the y axis (Pitch) and the angle ψ around the x axis (Yaw).

By the use of the inverse kinematics algorithm, the internal coordinates $\mathbf{q}_r$, corresponding to the desired end-effector pose, are calculated. The variable q_r represents the joint position, i.e. the angle ϑ for the rotational joint and the distance d for the translational joint. The desired internal coordinates are compared to the actual internal coordinates in the robot control system. On the basis of the positional error $\tilde{\mathbf{q}}$, the control system output $\mathbf{u}$ is calculated. The output $\mathbf{u}$ is converted from a digital into an analogue signal, amplified and delivered to the robot actuators. The actuators ensure the forces or torques necessary for the required robot motion. The robot motion is assessed by the sensors which were described in the chapter devoted to robot sensors.

7.1 Control of the robot in internal coordinates

The simplest robot control approach is based on controllers where the control loop is closed separately for each particular degree of freedom. Such controllers are suitable for control of independent second order systems with constant inertial and damping parameters. This approach is less suitable for robotic systems characterized by nonlinear and time varying behavior.

7.1.1 PD control of position

First, a simple proportional-derivative (PD) controller will be analyzed. The basic control scheme is shown in Figure 7.3. The control is based on calculation of the positional error and determination of control parameters, which enable reduction or suppression of the error. The positional error is reduced for each joint separately, which means that as many controllers are to be developed as there are degrees of freedom. The reference positions $\mathbf{q}_r$ are compared to the actual positions of the robot joints $\mathbf{q}$

$$\tilde{\mathbf{q}} = \mathbf{q}_r - \mathbf{q}. \tag{7.1}$$

The positional error $\tilde{\mathbf{q}}$ is amplified by the proportional position gain $\mathbf{K}_p$. As a robot manipulator has several degrees of freedom, the error $\tilde{\mathbf{q}}$ is expressed as a vector, while $\mathbf{K}_p$ is a diagonal matrix of the gains of all joint controllers. The calculated control input provokes robot motion in the direction of reduction of the positional error. As the actuation of the robot motors is proportional to the error, it can occur that the robot will overshoot instead of stopping in the desired position. Such overshoots are not allowed in robotics, as they may result in collisions with objects in the robot vicinity. To ensure safe and stable robot actions, a velocity closed loop is introduced with a negative sign. The velocity closed loop brings damping into the system. It is represented by the actual joint velocities $\dot{\mathbf{q}}$ multiplied by a diagonal matrix of velocity gains $\mathbf{K}_d$. The control law can be written in the following form

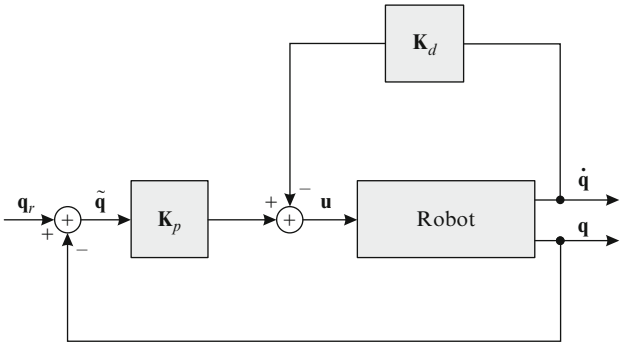

Fig. 7.3 PD position control with high damping

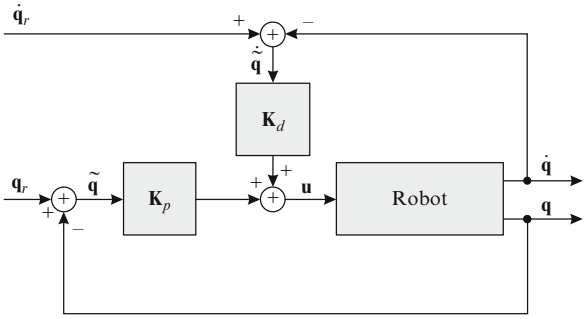

Fig. 7.4 PD position control

$$\mathbf{u} = \mathbf{K}_p(\mathbf{q}_r - \mathbf{q}) - \mathbf{K}_d\dot{\mathbf{q}}, \tag{7.2}$$

where $\mathbf{u}$ represents the control inputs, i.e. the joint forces or torques, which must be provided by the actuators. From equation (7.2) we can notice that at higher velocities of robot motions, the velocity control loop reduces the joint actuation and, by damping the system, ensures robot stability.

The control method shown in Figure 7.3 provides high damping of the system in the fastest part of the trajectory, which is usually not necessary. Such behavior of the controller can be avoided by upgrading the PD controller with the reference velocity signal. This signal is obtained as the numerical derivative of the desired position. The velocity error is used as control input

$$\dot{\tilde{\mathbf{q}}} = \dot{\mathbf{q}}_r - \dot{\mathbf{q}}. \tag{7.3}$$

The control algorithm demonstrated in Figure 7.4 can be written as

$$\mathbf{u} = \mathbf{K}_p(\mathbf{q}_r - \mathbf{q}) + \mathbf{K}_d(\dot{\mathbf{q}}_r - \dot{\mathbf{q}}). \tag{7.4}$$

As the difference between the reference velocity $\dot{\mathbf{q}}_r$ and $\dot{\mathbf{q}}$ is used instead of the total velocity $\dot{\mathbf{q}}$, the damping effect is reduced. For a positive difference the control loop can even accelerate the robot motion.

The synthesis of the PD position controller consists of determining the matrices $\mathbf{K}_p$ and $\mathbf{K}_d$. For fast response, the $\mathbf{K}_p$ gains must be high. By proper choice of the $\mathbf{K}_d$ gains, critical damping of the robot systems is obtained. The critical damping ensures fast response without overshoot. Such controllers must be built for each joint separately. The behavior of each controller is entirely independent of the controllers belonging to the other joints of the robot mechanism.

7.1.2 PD control of position with gravity compensation

In the chapter on robot dynamics we found that the robot mechanism is under the influence of inertial, Coriolis, centripetal and gravitational forces (4.46). In general

also friction forces, occurring in robot joints, must be included in the robot dynamic model. In a somewhat simplified model, only viscous friction, being proportional to the joint velocity, will be taken into account ($\mathbf{F}_v$ is a diagonal matrix of the joint friction coefficients). The enumerated forces must be overcome by the robot actuators which is evident from the following equation, similar to equation (4.46)

$$\mathbf{B}(\mathbf{q})\ddot{\mathbf{q}} + \mathbf{C}(\mathbf{q}, \dot{\mathbf{q}})\dot{\mathbf{q}} + \mathbf{F}_v\dot{\mathbf{q}} + \mathbf{g}(\mathbf{q}) = \tau. \tag{7.5}$$

When developing the PD controller, we did not pay attention to the specific forces influencing the robot mechanism. The robot controller calculated the required actuation forces solely on the basis of the difference between the desired and the actual joint positions. Such a controller cannot predict the force necessary to produce the desired robot motion. As the force is calculated from the positional error, this means that in general the error is never equal to zero. When knowing the dynamic robot model, we can predict the forces which are necessary for the performance of a particular robot motion. These forces are then generated by the robot motors regardless of the positional error signal.

In quasi-static conditions, when the robot is moving slowly, we can assume zero accelerations $\ddot{\mathbf{q}} \approx \mathbf{0}$ and velocities $\dot{\mathbf{q}} \approx \mathbf{0}$. The robot dynamic model is simplified as follows

$$\tau \approx \mathbf{g}(\mathbf{q}). \tag{7.6}$$

According to equation (7.6), the robot motors must above all compensate the gravity effect. The model of gravitational effects $\hat{\mathbf{g}}(\mathbf{q})$ (the circumflex denotes the robot model), which is a good approximation of the actual gravitational forces $\mathbf{g}(\mathbf{q})$, can be implemented in the control algorithm as shown in Figure 7.5. The PD controller, shown in Figure 7.3, was upgraded with an additional control loop, which calculates the gravitational forces from the actual robot position and directly adds them to the controller output. The control algorithm shown in Figure 7.5 can be written as follows

$$\mathbf{u} = \mathbf{K}_p(\mathbf{q}_r - \mathbf{q}) - \mathbf{K}_d\dot{\mathbf{q}} + \hat{\mathbf{g}}(\mathbf{q}). \tag{7.7}$$

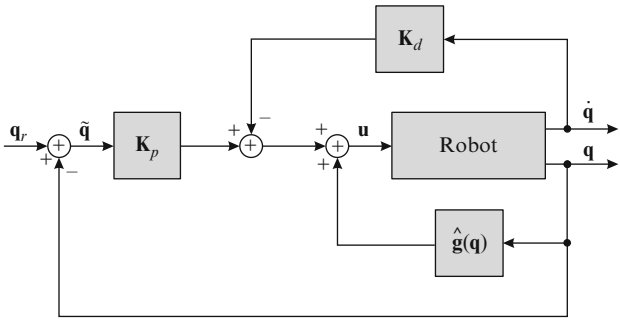

Fig. 7.5 PD control with gravity compensation

By introducing gravity compensation, the burden of reducing the errors caused by gravity, is taken away from the PD controller. In this way the errors in trajectory tracking are significantly reduced.

7.1.3 Control of the robot based on inverse dynamics

When studying the PD controller with gravity compensation, we investigated the robot dynamic model in order to improve the efficiency of the control method. With the control method based on inverse dynamics, this concept will be further upgraded. From the equations describing the dynamic behavior of a two-segment robot manipulator (4.46), we can clearly observe that the robot model is nonlinear. A linear controller, such as the PD controller, is therefore not the best choice.

We shall derive the new control scheme from the robot dynamic model described by equation (7.5). Let us assume that the torques τ, generated by the motors, are equal to the control outputs $\mathbf{u}$. Equation (7.5) can be rewritten

$$\mathbf{B}(\mathbf{q})\ddot{\mathbf{q}} + \mathbf{C}(\mathbf{q},\dot{\mathbf{q}})\dot{\mathbf{q}} + \mathbf{F}_v\dot{\mathbf{q}} + \mathbf{g}(\mathbf{q}) = \mathbf{u}. \tag{7.8}$$

In the next step we will determine the direct robot dynamic model, which describes robot motions under the influence of the given joint torques. First we express the acceleration $\ddot{\mathbf{q}}$ from equation (7.8)

$$\ddot{\mathbf{q}} = \mathbf{B}^{-1}(\mathbf{q})\left(\mathbf{u} - \left(\mathbf{C}(\mathbf{q},\dot{\mathbf{q}})\dot{\mathbf{q}} + \mathbf{F}_v\dot{\mathbf{q}} + \mathbf{g}(\mathbf{q})\right)\right). \tag{7.9}$$

By integrating the acceleration, while taking into account the initial velocity value, the velocity of robot motion is obtained. By integrating the velocity, while taking into account the initial position, we calculate the actual positions in the robot joints. The direct dynamic model of a robot mechanism is shown in Figure 7.6.

In order to simplify the dynamic equations, we shall define a new variable $\mathbf{n}(\mathbf{q},\dot{\mathbf{q}})$ comprising all dynamic components except the inertial component

$$\mathbf{n}(\mathbf{q},\dot{\mathbf{q}}) = \mathbf{C}(\mathbf{q},\dot{\mathbf{q}})\dot{\mathbf{q}} + \mathbf{F}_v\dot{\mathbf{q}} + \mathbf{g}(\mathbf{q}). \tag{7.10}$$

The robot dynamic model can be described with the following shorter equation

$$\mathbf{B}(\mathbf{q})\ddot{\mathbf{q}} + \mathbf{n}(\mathbf{q},\dot{\mathbf{q}}) = \tau. \tag{7.11}$$

In the same way also equation (7.9) can be written in a shorter form

$$\ddot{\mathbf{q}} = \mathbf{B}^{-1}(\mathbf{q})\left(\mathbf{u} - \mathbf{n}(\mathbf{q},\dot{\mathbf{q}})\right). \tag{7.12}$$

Let us assume that the robot dynamic model is known. The inertial matrix $\hat{\mathbf{B}}(\mathbf{q})$ is an approximation of the real values $\mathbf{B}(\mathbf{q})$, while $\hat{\mathbf{n}}(\mathbf{q},\dot{\mathbf{q}})$ represents an approximation of $\mathbf{n}(\mathbf{q},\dot{\mathbf{q}})$ as follows

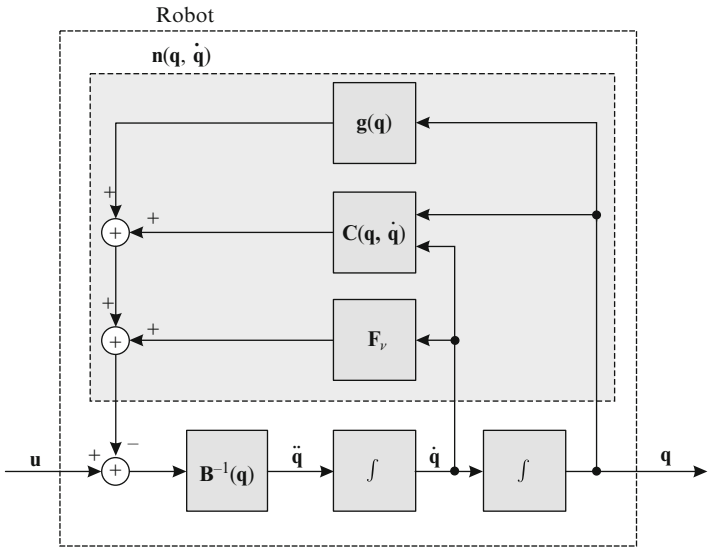

Fig. 7.6 The direct dynamic model of a robot mechanism

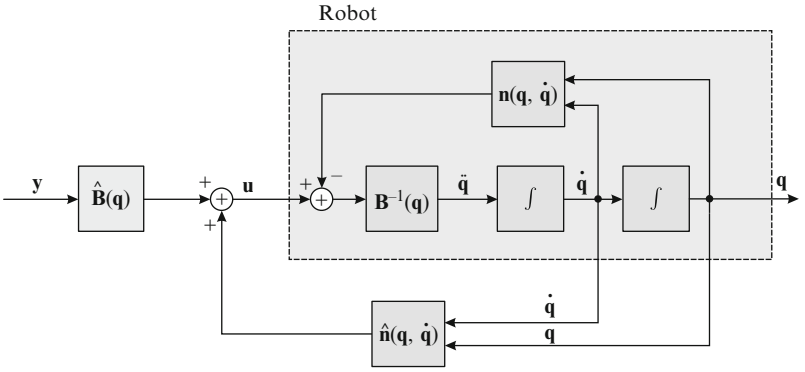

Fig. 7.7 Linearization of the control system by implementing the inverse dynamic model

$$\hat{\mathbf{n}}(\mathbf{q},\dot{\mathbf{q}}) = \hat{\mathbf{C}}(\mathbf{q},\dot{\mathbf{q}})\dot{\mathbf{q}} + \hat{\mathbf{F}}_v\dot{\mathbf{q}} + \hat{\mathbf{g}}(\mathbf{q}). \tag{7.13}$$

The controller output $\mathbf{u}$ is determined by the following equation

$$\mathbf{u} = \hat{\mathbf{B}}(\mathbf{q})\mathbf{y} + \hat{\mathbf{n}}(\mathbf{q},\dot{\mathbf{q}}), \tag{7.14}$$

where the approximate inverse dynamic model of the robot was used. The system, combining equations (7.12) and (7.14), is shown in Figure 7.7.

Let us assume the equivalence $\hat{\mathbf{B}}(\mathbf{q}) = \mathbf{B}(\mathbf{q})$ and $\hat{\mathbf{n}}(\mathbf{q},\dot{\mathbf{q}}) = \mathbf{n}(\mathbf{q},\dot{\mathbf{q}})$. In Figure 7.7 we observe that the signals $\hat{\mathbf{n}}(\mathbf{q},\dot{\mathbf{q}})$ and $\mathbf{n}(\mathbf{q},\dot{\mathbf{q}})$ subtract, as one is presented

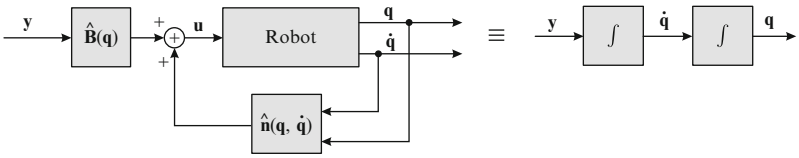

Fig. 7.8 The linearized system

with a positive and the other with a negative sign. In a similar way, the product of matrices $\hat{\mathbf{B}}(\mathbf{q})$ and $\mathbf{B}^{-1}(\mathbf{q})$ results in a unit matrix, which can be omitted. The simplified system is shown in Figure 7.8. By implementing the inverse dynamics (7.14), the control system is linearized, as there are only two integrators between the input $\mathbf{y}$ and the output $\mathbf{q}$. The system is not only linear, but is also decoupled, as e.g. the first element of the vector $\mathbf{y}$ only influences the first element of the position vector $\mathbf{q}$. From Figure 7.8 it is also not difficult to realize that the variable $\mathbf{y}$ has the characteristics of acceleration, thus

$$\mathbf{y} = \ddot{\mathbf{q}}. \tag{7.15}$$

In an ideal case, it would suffice to determine the desired joint accelerations as the second derivatives of the desired joint positions and the control system will track the prescribed joint trajectories. As we never have a fully accurate dynamic model of the robot, always a difference will occur between the desired and the actual joint positions and will increase with time. The positional error is defined by

$$\tilde{\mathbf{q}} = \mathbf{q}_r - \mathbf{q}, \tag{7.16}$$

where $\mathbf{q}_r$ represents the desired robot position. In a similar way also the velocity error can be defined as the difference between the desired and the actual velocity

$$\dot{\tilde{\mathbf{q}}} = \dot{\mathbf{q}}_r - \dot{\mathbf{q}}. \tag{7.17}$$

The vector $\mathbf{y}$, having the acceleration characteristics, can be now written as

$$\mathbf{y} = \ddot{\mathbf{q}}_r + \mathbf{K}_p(\mathbf{q}_r - \mathbf{q}) + \mathbf{K}_d(\dot{\mathbf{q}}_r - \dot{\mathbf{q}}). \tag{7.18}$$

It consists of the reference acceleration $\ddot{\mathbf{q}}_r$ and two contributing signals which depend on the errors of position and velocity. These two signals suppress the error arising because of the imperfectly modeled dynamics. The complete control scheme is shown in Figure 7.9.

By considering equation (7.18) and the equality $\mathbf{y} = \ddot{\mathbf{q}}$, the differential equation describing the robot dynamics can be written as

$$\ddot{\tilde{\mathbf{q}}} + \mathbf{K}_d\dot{\tilde{\mathbf{q}}} + \mathbf{K}_p\tilde{\mathbf{q}} = \mathbf{0}, \tag{7.19}$$

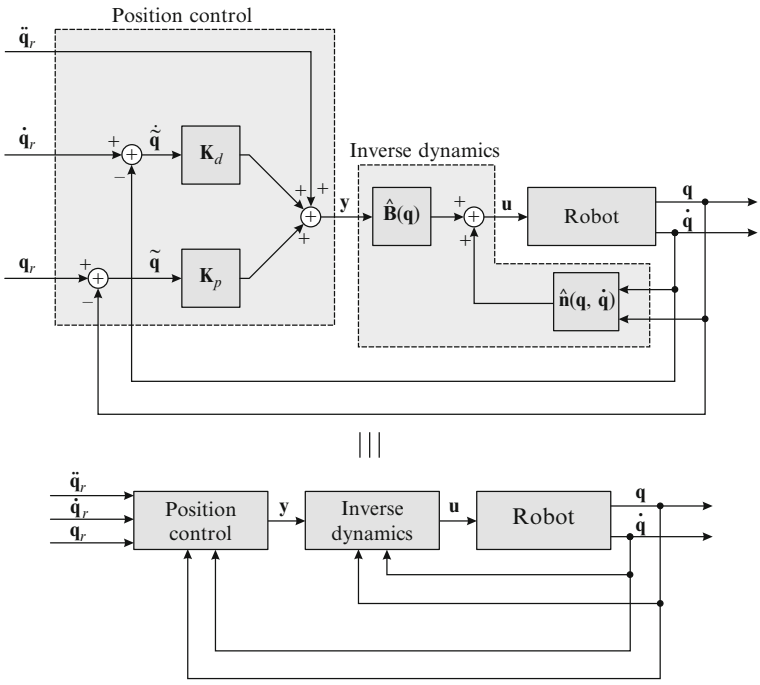

Fig. 7.9 Control of the robot based on inverse dynamics

where the acceleration error $\ddot{\tilde{\mathbf{q}}} = \ddot{\mathbf{q}}_r - \ddot{\mathbf{q}}$ was introduced. The differential equation (7.19) describes the time dependence of the control error as it approaches zero. The dynamics of the response is determined by the gains $\mathbf{K}_p$ and $\mathbf{K}_d$.

7.2 Control of the robot in external coordinates

All the control schemes studied up to now were based on control of the internal coordinates, i.e. joint positions. The desired positions, velocities and accelerations were determined by the robot joint variables. Usually we are more interested in the motion of the robot end-effector than in the displacements of particular robot joints. At the tip of the robot, different tools are attached to accomplish various robot tasks. In the further text we shall focus on the robot control in the external coordinates.

7.2.1 Control based on the transposed Jacobian matrix

The control method is based on the already known equation (4.17) connecting the forces acting at the robot end-effector with the joint torques. The relation is defined by the use of the transposed Jacobian matrix

$$\tau = \mathbf{J}^T(\mathbf{q})\mathbf{f}, \tag{7.20}$$

where the vector τ represents the joint torques and $\mathbf{f}$ is the force at the robot end-point.

It is our aim to control the pose of the robot end-effector, where its desired pose is defined by the vector $\mathbf{x}_r$ and the actual pose is given by the vector $\mathbf{x}$. The vectors $\mathbf{x}_r$ and $\mathbf{x}$ in general comprise six variables, three determining the position of the robot end-point and three for the orientation of the end-effector, thus $\mathbf{x} = \left[x\ y\ z\ \varphi\ \vartheta\ \psi\right]^T$. Robots are usually not equipped with sensors assessing the pose of the end-effector; robot sensors measure the joint variables. The pose of the robot end-effector must be therefore determined by using the equations of the direct kinematic model $\mathbf{x} = \mathbf{k}(\mathbf{q})$ introduced in the chapter on robot kinematics (4.4). The positional error of the robot end-effector is calculated as

$$\tilde{\mathbf{x}} = \mathbf{x}_r - \mathbf{x} = \mathbf{x}_r - \mathbf{k}(\mathbf{q}). \tag{7.21}$$

The positional error must be reduced to zero. A simple proportional control system with the gain matrix $\mathbf{K}_p$ is introduced

$$\mathbf{f} = \mathbf{K}_p\tilde{\mathbf{x}}. \tag{7.22}$$

When analyzing equation (7.22) more closely, we find that it reminds us of the equation describing the behavior of a spring, where the force is proportional to the spring elongation. This consideration helps us to explain the introduced control principle. Let as imagine that there are six springs virtually attached to the robot end-effector, one spring for each degree of freedom (three for position and three for orientation). When the robot moves away from the desired pose, the springs are elongated and pull the robot end-effector into the desired pose with the force proportional to the positional error. The force $\mathbf{f}$ therefore pushes the robot end-effector towards the desired pose. As the robot displacement can only be produced by the motors in the joints, the variables controlling the motors must be calculated from the force $\mathbf{f}$. This calculation is performed by the help of the transposed Jacobian matrix as shown in equation (7.20)

$$\mathbf{u} = \mathbf{J}^T(\mathbf{q})\mathbf{f}. \tag{7.23}$$

The vector $\mathbf{u}$ represents the desired joint torques. The control method based on the transposed Jacobian matrix is shown in Figure 7.10.

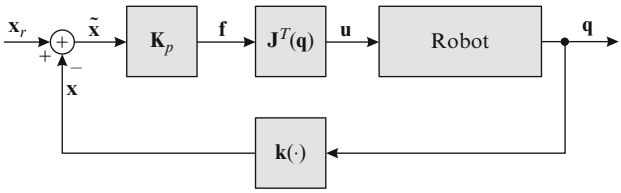

Fig. 7.10 Control based on the transposed Jacobian matrix

7.2.2 Control based on the inverse Jacobian matrix

The control method is based on the relation between the joint velocities and the velocities of the robot end-point (4.10), which is given by the Jacobian matrix. In equation (4.10) we emphasize the time derivatives of external coordinates $\mathbf{x}$ and internal coordinates $\mathbf{q}$

$$\dot{\mathbf{x}} = \mathbf{J}(\mathbf{q})\dot{\mathbf{q}} \quad \Leftrightarrow \quad \frac{d\mathbf{x}}{dt} = \mathbf{J}(\mathbf{q})\frac{d\mathbf{q}}{dt}. \tag{7.24}$$

As dt appears in the denominator on both sides of equation (7.24), it can be omitted. In this way we obtain the relation between changes of the internal coordinates and changes of the pose of the robot end-point

$$d\mathbf{x} = \mathbf{J}(\mathbf{q})d\mathbf{q}. \tag{7.25}$$

Equation (7.25) is valid only for small displacements.

As with the previously studied control method, based on the transposed Jacobian matrix, also in this case we first calculate the error of the pose of the robot end-point by using equation (7.21). When the error in the pose is small, we can calculate the positional error in the internal coordinates by the inverse relation (7.25)

$$\tilde{\mathbf{q}} = \mathbf{J}^{-1}(\mathbf{q})\tilde{\mathbf{x}}. \tag{7.26}$$

In this way the control method is translated to the known method of robot control in the internal coordinates. In the simplest example, based on the proportional controller, we can write

$$\mathbf{u} = \mathbf{K}_p\tilde{\mathbf{q}}. \tag{7.27}$$

The control method, based on the inverse Jacobian matrix, is shown in Figure 7.11.

7.2.3 PD control of position with gravity compensation

The PD control of position with gravity compensation was already studied in detail for the internal coordinates. Now we shall derive the analogue control algorithm in

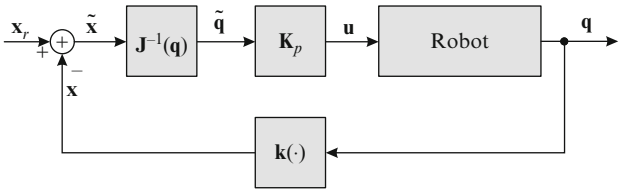

Fig. 7.11 Control based on the inverse Jacobian matrix

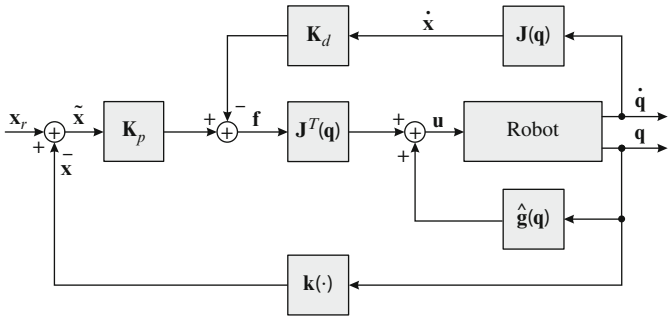

Fig. 7.12 PD control with gravity compensation in external coordinates

the external coordinates. The starting point will be equation (7.21) expressing the error of the pose of the end-effector. The velocity of the robot end-point is calculated with the help of the Jacobian matrix from the joint velocities

$$\dot{\mathbf{x}} = \mathbf{J}(\mathbf{q})\dot{\mathbf{q}}. \tag{7.28}$$

The equation describing the PD controller in external coordinates is analogous to that written in the internal coordinates (7.2)

$$\mathbf{f} = \mathbf{K}_p\tilde{\mathbf{x}} - \mathbf{K}_d\dot{\mathbf{x}}. \tag{7.29}$$

In equation (7.29), the pose error is multiplied by the matrix of the positional gains $\mathbf{K}_p$, while the velocity error is multiplied by the matrix $\mathbf{K}_d$. The negative sign of the velocity error introduces damping into the system. The joint torques are calculated from the force $\mathbf{f}$, acting at the tip of the robot, with the help of the transposed Jacobian matrix (in a similar way as in equation (7.23)) and by adding the component compensating gravity (as in equation (7.7)). The control algorithm is written as

$$\mathbf{u} = \mathbf{J}^T(\mathbf{q})\mathbf{f} + \hat{\mathbf{g}}(\mathbf{q}). \tag{7.30}$$

The complete control scheme is shown in Figure 7.12.

7.2.4 Control of the robot based on inverse dynamics

In the chapter on the control of robots in the internal coordinates, the following controller based on inverse dynamics was introduced

$$\mathbf{u} = \hat{\mathbf{B}}(\mathbf{q})\mathbf{y} + \hat{\mathbf{n}}(\mathbf{q}, \dot{\mathbf{q}}). \tag{7.31}$$

We also learned that the vector $\mathbf{y}$ has the characteristics of acceleration

$$\mathbf{y} = \ddot{\mathbf{q}}, \tag{7.32}$$

which was determined in such a way, that the robot tracked the desired trajectory expressed in the internal coordinates. As it is our aim to develop a control method in the external coordinates, the $\mathbf{y}$ signal must be adequately adapted. Equation (7.31), linearizing the system, remains unchanged.

We shall again start from the equation relating the joint velocities to the robot end-effector velocities

$$\dot{\mathbf{x}} = \mathbf{J}(\mathbf{q})\dot{\mathbf{q}}. \tag{7.33}$$

By calculating the time derivative of equation (7.33), we obtain

$$\ddot{\mathbf{x}} = \mathbf{J}(\mathbf{q})\ddot{\mathbf{q}} + \dot{\mathbf{J}}(\mathbf{q}, \dot{\mathbf{q}})\dot{\mathbf{q}}. \tag{7.34}$$

The error of the pose of the robot end-effector is determined as the difference between its desired and its actual pose

$$\tilde{\mathbf{x}} = \mathbf{x}_r - \mathbf{x} = \mathbf{x}_r - \mathbf{k}(\mathbf{q}). \tag{7.35}$$

In a similar way the velocity error of the robot end-effector is determined

$$\dot{\tilde{\mathbf{x}}} = \dot{\mathbf{x}}_r - \dot{\mathbf{x}} = \dot{\mathbf{x}}_r - \mathbf{J}(\mathbf{q})\dot{\mathbf{q}}. \tag{7.36}$$

The acceleration error is the difference between the desired and the actual acceleration

$$\ddot{\tilde{\mathbf{x}}} = \ddot{\mathbf{x}}_r - \ddot{\mathbf{x}}. \tag{7.37}$$

When developing the inverse dynamics based controller in the internal coordinates, equation (7.19) was derived describing the dynamics of the control error in the form $\ddot{\tilde{\mathbf{q}}} + \mathbf{K}_d\dot{\tilde{\mathbf{q}}} + \mathbf{K}_p\tilde{\mathbf{q}} = \mathbf{0}$. An analogous equation can be written for the error of the end-effector pose. From this equation the acceleration $\ddot{\mathbf{x}}$ of the robot end-effector can be expressed

$$\ddot{\tilde{\mathbf{x}}} + \mathbf{K}_d\dot{\tilde{\mathbf{x}}} + \mathbf{K}_p\tilde{\mathbf{x}} = \mathbf{0} \quad \Rightarrow \quad \ddot{\mathbf{x}} = \ddot{\mathbf{x}}_r + \mathbf{K}_d\dot{\tilde{\mathbf{x}}} + \mathbf{K}_p\tilde{\mathbf{x}}. \tag{7.38}$$

From equation (7.34) we express $\ddot{\mathbf{q}}$ taking into account the equality $\mathbf{y} = \ddot{\mathbf{q}}$

$$\mathbf{y} = \mathbf{J}^{-1}(\mathbf{q})\left(\ddot{\mathbf{x}} - \dot{\mathbf{J}}(\mathbf{q}, \dot{\mathbf{q}})\dot{\mathbf{q}}\right). \tag{7.39}$$

By replacing $\ddot{\mathbf{x}}$ in equation (7.39) with expression (7.38), the control algorithm based on inverse dynamics in the external coordinates is obtained

$$\mathbf{y} = \mathbf{J}^{-1}(\mathbf{q})\left(\ddot{\mathbf{x}}_r + \mathbf{K}_d\dot{\tilde{\mathbf{x}}} + \mathbf{K}_p\tilde{\mathbf{x}} - \dot{\mathbf{J}}(\mathbf{q}, \dot{\mathbf{q}})\dot{\mathbf{q}}\right). \tag{7.40}$$

The control scheme encompassing the linearization of the system based on inverse dynamics (7.31) and the closed loop control (7.40) is shown in Figure 7.13.

7.3 Control of the contact force

The control of position is sufficient when a robot manipulator follows a trajectory in free space. When contact occurs between the robot end-effector and the environment, position control is not an appropriate approach. Let us imagine a robot

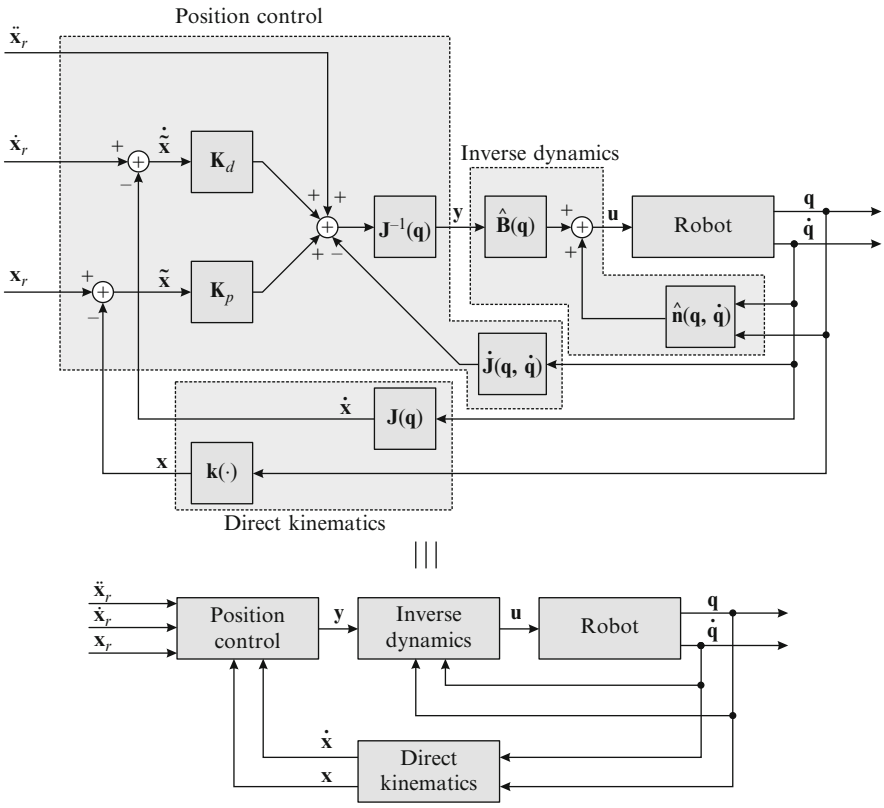

Fig. 7.13 Robot control based on inverse dynamics in external coordinates

manipulator cleaning a window with a sponge. As the sponge is very compliant, it is possible to control the force between the robot and window by controlling the position between the robot gripper and the window. If the sponge is sufficiently compliant and when we know the position of the window accurately enough, the robot will appropriately accomplish the task.

If the compliance of the robot tool or its environment is smaller, then it is not so simple to execute the tasks which require contact between the robot and its environment. Let us now imagine a robot scraping paint from a glassy surface while using a stiff tool. Any uncertainty in the position of the glassy surface or malfunction of the robot control system will prevent satisfactory execution of the task; either the glass will break, or the robot will uselessly wave in the air.

In both robot tasks, i.e. cleaning a window or scraping a smooth surface, it is more reasonable that instead of position of the glassy surface we determine the force that the robot should exert on the environment. Most of the modern industrial robots are carrying out relatively simple tasks, such as spot welding, spray painting and various point-to-point operations. Several robot applications, however, require

control of the contact force. A characteristic example is grinding or a similar robot machining task. An important area of industrial robotics is also robot assembly, where several component parts are to be assembled. In such robot tasks, sensing and controlling the forces is of utmost importance.

Accurate operation of a robot manipulator in an uncertain, non-structured and changeable environment is required for efficient use of robots in an assembly task. Here, several component parts must be brought together with high accuracy. Measurement and control of the contact forces enable the required positional accuracy of the robot manipulator to be reached. As relative measurements are used in robot force control, the absolute errors in positioning of either the manipulator or the object are not as critical as in robot position control. When dealing with stiff objects, already small changes in position produce large contact forces. Measurement and control of those forces can lead to significantly higher positional accuracy of robot movement.

When a robot is exerting force on the environment, we deal with two types of robot tasks. In the first case we would like the robot end-effector to be brought into a desired pose while the robot is in contact with the environment. This is the case of robot assembly. A characteristic example is that of inserting a peg into a hole. The robot movement must be of such nature that the contact force is reduced to zero or to a minimal value allowed. In the second type of robot task, we require of the robot end-effector to exert a predetermined force on the environment. This is the example of robot grinding. Here, the robot movement depend on the difference between the desired and the actually measured contact force.

The robot force control method will be based on control of the robot using inverse dynamics. Because of the interaction of the robot with the environment, an additional component, representing the contact force $\mathbf{f}$, appears in the inverse dynamic model. As the forces acting at the robot end-effector are transformed into the joint torques by the use of the transposed Jacobian matrix (4.17), we can write the robot dynamic model in the following form

$$\mathbf{B}(\mathbf{q})\ddot{\mathbf{q}} + \mathbf{C}(\mathbf{q},\dot{\mathbf{q}})\dot{\mathbf{q}} + \mathbf{F}_v\dot{\mathbf{q}} + \mathbf{g}(\mathbf{q}) = \tau - \mathbf{J}^T(\mathbf{q})\mathbf{f}. \tag{7.41}$$

On the right hand side of the equation (7.5) we added the component $-\mathbf{J}^T(\mathbf{q})\mathbf{f}$ representing the force of interaction with the environment. It can be seen that the force $\mathbf{f}$ acts through the transposed Jacobian matrix in a similar way as the joint torques, i.e. it tries to produce robot motion. The model (7.41) can be rewritten in a shorter form by introducing

$$\mathbf{n}(\mathbf{q},\dot{\mathbf{q}}) = \mathbf{C}(\mathbf{q},\dot{\mathbf{q}})\dot{\mathbf{q}} + \mathbf{F}\dot{\mathbf{q}} + \mathbf{g}(\mathbf{q}), \tag{7.42}$$

which gives us the following dynamic model of a robot in contact with its environment

$$\mathbf{B}(\mathbf{q})\ddot{\mathbf{q}} + \mathbf{n}(\mathbf{q},\dot{\mathbf{q}}) = \tau - \mathbf{J}^T(\mathbf{q})\mathbf{f}. \tag{7.43}$$

7.3.1 Linearization of a robot system through inverse dynamics

Let us denote the control output, representing the desired actuation torques in the robot joints, by the vector $\mathbf{u}$. Equation (7.43) can be written as follows

$$\mathbf{B}(\mathbf{q})\ddot{\mathbf{q}} + \mathbf{n}(\mathbf{q}, \dot{\mathbf{q}}) + \mathbf{J}^T(\mathbf{q})\mathbf{f} = \mathbf{u}. \tag{7.44}$$

From equation (7.44) we express the direct dynamic model

$$\ddot{\mathbf{q}} = \mathbf{B}^{-1}(\mathbf{q})\left(\mathbf{u} - \mathbf{n}(\mathbf{q}, \dot{\mathbf{q}}) - \mathbf{J}^T(\mathbf{q})\mathbf{f}\right). \tag{7.45}$$

Equation (7.45) describes the response of the robot system to the control input $\mathbf{u}$. By integrating the acceleration, while taking into account the initial velocity value, the actual velocity of the robot motion is obtained. By integrating the velocity, while taking into the account the initial position, we calculate the actual positions in the robot joints. The described model is represented by the block *Robot* in Figure 7.14.

In a similar way as when developing the control method based on inverse dynamics, we will linearize the system by including the inverse dynamic model into the closed loop

$$\mathbf{u} = \hat{\mathbf{B}}(\mathbf{q})\mathbf{y} + \hat{\mathbf{n}}(\mathbf{q}, \dot{\mathbf{q}}) + \mathbf{J}^T(\mathbf{q})\mathbf{f}, \tag{7.46}$$

The use of circumflex denotes the estimated parameters of the robot system. The difference between equation (7.46) and equation (7.14), representing the control based

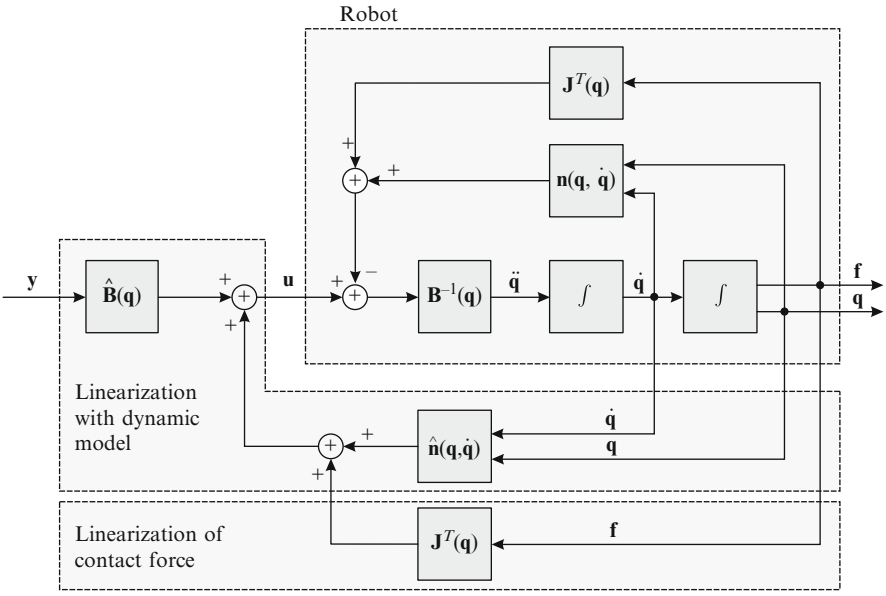

Fig. 7.14 Linearization of the control system by implementing the inverse dynamic model and the measured contact force

on inverse dynamics in internal coordinates, is the component $\mathbf{J}^T(\mathbf{q})\mathbf{f}$, compensating the influence of external forces on the robot mechanism. The control scheme, combining equations (7.45) and (7.46) is shown in Figure 7.14. Assuming that the estimated parameters are equal to the actual robot parameters, it can be observed, that by introducing the closed loop (7.46), the system is linearized because there are only two integrators between the input $\mathbf{y}$ and the output $\mathbf{q}$, as already demonstrated in Figure 7.8.

7.3.2 Force control

After linearizing the control system, the input vector $\mathbf{y}$ must be determined. The force control will be translated to control of the pose of the end-effector. This can be, in a simplified way, explained with the following reasoning: if we wish the robot to increase the force exerted on the environment, the robot end-effector must be displaced in the direction of the action of the force. Now we can use the control system which was developed to control the robot in the external coordinates (7.40). The control scheme of the robot end-effector including the linearization, while taking into account the contact force, is shown in Figure 7.15.

Up to this point we mainly summarized the knowledge of the pose control of the robot end-effector as explained in the previous chapters. In the next step we will determine the desired pose, velocity and acceleration of the robot end-effector, on the basis of the force measured between the robot end-point and its environment.

Let us assume that we wish to control a constant desired force $\mathbf{f}_r$. With the force wrist sensor, the contact force $\mathbf{f}$ is measured. The difference between the desired and measured force represents the force error

$$\tilde{\mathbf{f}} = \mathbf{f}_r - \mathbf{f}. \tag{7.47}$$

The desired robot motion will be calculated based on the assumption that the force $\tilde{\mathbf{f}}$ must displace a virtual object with inertia $\mathbf{B}_c$ and damping $\mathbf{F}_c$. In our case the virtual object is in fact the robot end-effector. For easier understanding, let us consider a system with only one degree of freedom. When a force acts on such a system, an

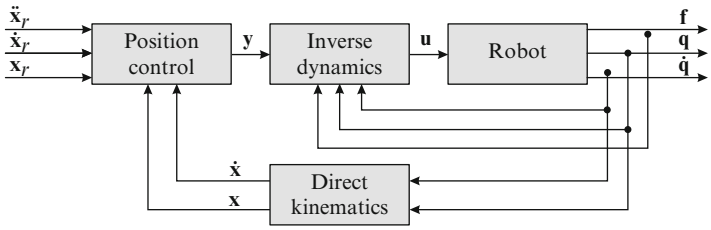

Fig. 7.15 Robot control based on inverse dynamics in external coordinates including the contact force

accelerated movement will start. The movement will be determined by the force, the mass of the object and the damping. The robot end-effector therefore behaves as a system consisting of a mass and a damper, which are under the influence of the force $\tilde{\mathbf{f}}$. For more degrees of freedom we can write the following differential equation describing the movement of the object

$$\tilde{\mathbf{f}} = \mathbf{B}_c \ddot{\mathbf{x}}_c + \mathbf{F}_c \dot{\mathbf{x}}_c. \tag{7.48}$$

The matrices $\mathbf{B}_c$ and $\mathbf{F}_c$ determine the movement of the object under the influence of the force $\tilde{\mathbf{f}}$. From equation (7.48) the acceleration of the virtual object can be calculated

$$\ddot{\mathbf{x}}_c = \mathbf{B}_c^{-1} \left(\tilde{\mathbf{f}} - \mathbf{F}_c \dot{\mathbf{x}}_c \right). \tag{7.49}$$

By integrating the equation (7.49), the velocities and the pose of the object are calculated, as shown in Figure 7.16. In this way the reference pose $\mathbf{x}_c$, reference velocity $\dot{\mathbf{x}}_c$ and reference acceleration $\ddot{\mathbf{x}}_c$ are determined from the force error. The calculated variables are inputs to the control system, shown in Figure 7.15. In this way the force control was translated into the already known robot control in external coordinates.

In order to simultaneously control also the pose of the robot end-effector, parallel composition is included. Parallel composition assumes that the reference control variables are obtained by summing the references for force control ($\mathbf{x}_c$, $\dot{\mathbf{x}}_c$, $\ddot{\mathbf{x}}_c$) and references for the pose control ($\mathbf{x}_d$, $\dot{\mathbf{x}}_d$, $\ddot{\mathbf{x}}_d$). The parallel composition is defined by equations

$$\begin{aligned} \mathbf{x}_r &= \mathbf{x}_d + \mathbf{x}_c \\ \dot{\mathbf{x}}_r &= \dot{\mathbf{x}}_d + \dot{\mathbf{x}}_c \\ \ddot{\mathbf{x}}_r &= \ddot{\mathbf{x}}_d + \ddot{\mathbf{x}}_c \end{aligned} \tag{7.50}$$

The control system incorporating the contact force control, parallel composition and control of the robot based on inverse dynamics in external coordinates is shown in Figure 7.17. The force control is obtained by selecting

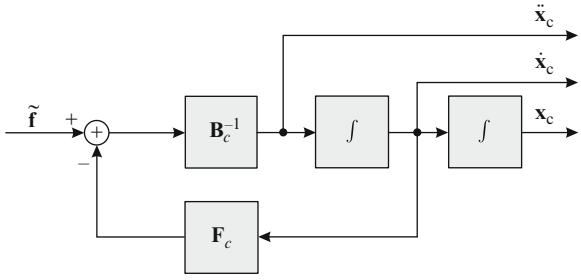

Fig. 7.16 Force control translated into control of the pose of robot end-effector

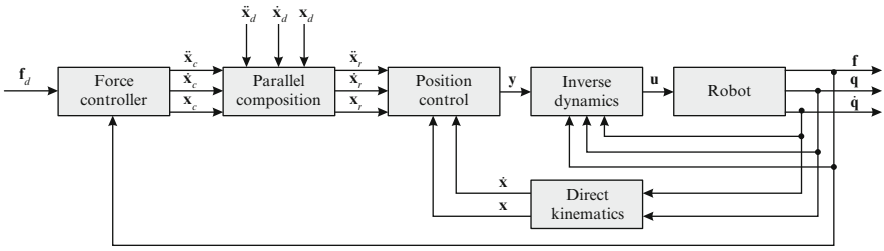

Fig. 7.17 Direct force control in the external coordinates

$$\begin{aligned} \mathbf{x}_r &= \mathbf{x}_c \\ \dot{\mathbf{x}}_r &= \dot{\mathbf{x}}_c \\ \ddot{\mathbf{x}}_r &= \ddot{\mathbf{x}}_c \end{aligned} \tag{7.51}$$

The described control method enables the control of force. However, it does not enable independent control of the pose of the robot end-effector as it is determined by the error in the force signal.