Chapter 12
Introduction to Robotics-Mathematical Issues

S.M. Raafat and F.A. Raheem

Abstract Robotic systems play a crucial role in the world and sustainability. Robots presence and our dependencies on them are progressively growing. This chapter brings together mathematical developments in the important fields of robotics, kinematics, dynamics, path planning, control, and vision. Introduction is made on development of robotics in different areas of application (types of robots and applications). The kinematics of a robot manipulator is briefly described. The formulation of dynamics for the manipulator has been obtained based on Lagrange's energy function. Linear Segments with Parabolic Blends and Third-Order Polynomial Trajectory Planning have been described in detail. Different classical control strategies are presented. Finally, basic concepts of Robot Vision are presented.

Keywords Robot kinematic • Robotic dynamics • Path and trajectory planning • Robot control • Robot vision

12.1 Introduction on Robotics; Robot Types and Applications

Through good design practices and thorough consideration of detail, engineers have succeeded in applying robotic systems to a wide variety of industrial, manufacturing, space, domestic or household, social, and medical situations where the environment is structured or predictable. On a practical level, robots are distinguished from other electromechanical motion equipment by their dexterous manipulation capability in that robots can work, position, and move tools and other objects with far greater dexterity than other machines found in the factory. Process robot systems are functional components with grippers, end-effectors, sensors, and process equipment organized to perform a controlled sequence of tasks to execute a process—they require sophisticated control systems. The combined effects of

S.M. Raafat (✉) • F.A. Raheem
Automation and Robotics Research Unit, Control and System Engineering Department, University of Technology, Baghdad, Iraq
e-mail: 60154@uotechnology.edu.iq

kinematic structure, axis drive mechanism design, and real-time motion control determine the major manipulation performance characteristics such as reach and dexterity, payload, quickness, and precision. For Cartesian robots, the range of motion of the first three axes describes the reachable workspace. Some robots will have unusable spaces such as dead zones, singular poses, and wrist-wrap poses inside of the boundaries of their reach. Usually motion test, simulations, or other analyses are used to verify reach and dexterity for each application (Lewis et al. 1999).

All common commercial industrial robots are serial link manipulators with no more than six kinematically coupled axes of motion. By convention, the axes of motion are numbered in sequence as they are encountered from the base on out to the wrist. The first three axes account for the spatial positioning motion of the robot; their configuration determines the shape of the space through which the robot can be positioned. Any subsequent axes in the kinematic chain provide rotational motions to orient the end of the robot arm and are referred to as wrist axes. There are, in principle, two primary types of motion that a robot axis can produce in its robot arm: either revolute (rotational) or prismatic (translational). It is often useful to classify robots according to the orientation and type of their first three axes. As the robot arm has only three degrees of freedom, there exist a limited number of possible combinations resulting all together in 36 different structures of robot arms. There are very common commercial robot configurations: Articulated robots (robotic arms), spherical, Selective Compliance Assembly Robot Arm (SCARA), cylindrical, Cartesian/gantry (as shown in Fig. 12.1), and parallel robots. Cartesian coordinate robots use orthogonal prismatic axes, usually referred to as X, Y, and Z,

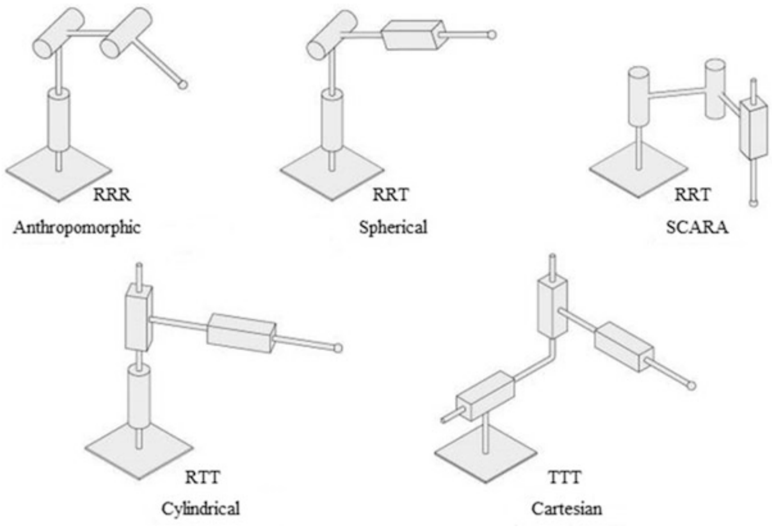

Fig. 12.1 Different types of Robot arms: (RRR) all three joints of the rotational type; (RRT) two joints are rotational and one is translational; (RTT) one rotational and two translational degrees of freedom (Bajd et al. 2010)

to translate their end-effector or payload through their rectangular workspace. One, two, or three revolute wrist axes may be added for orientation. Gantry robots are the most common Cartesian style. Commercial models of spherical and cylindrical robots were originally very common and popular in machine tending and material handling applications (Lewis et al. 1999; Bajd et al. 2010).

Robots are also characterized by the type of actuators employed. Typically manipulators have hydraulic or electric actuation. In some cases, pneumatic actuators are used. A number of successful manipulator designs have emerged, each with a different arrangement of joints and links. Some "elbow" designs, such as the PUMA robots and the SPAR Remote Manipulator System, have a fairly anthropomorphic structure, with revolute joints arranged into "shoulder," "elbow," and "wrist" sections. A mix of revolute and prismatic joints has been adopted in the Stanford Manipulator and the SCARA types of arms. Other arms, such as those produced by IBM, feature prismatic joints for the "shoulder," with a spherical wrist attached. In this case, the prismatic joints are essentially used as positioning devices, with the wrist used for fine motions (Lewis et al. 1999).

The largest number of industrial robot manipulators is found in the car industry (Bajd et al. 2010). They are mainly used for welding. Other important applications of industrial robots are to move objects from point to point. Such examples are found in the process of palletizing. Industrial robots are frequently used in aggressive or dangerous environments, such as spray painting. The request for robot manipulators in the area of industrial assembly of component parts into a functional system is progressively increasing. The interest in robot manipulators in medicine is rapidly increasing as well. They can be found in surgical applications (Boonvisut and Cavusoglu 2013; Keung et al. 2013), drug delivery (Zhou et al. 2013), or in rehabilitation for training of a paralyzed extremity after stroke (Freeman et al. 2012). Exceptional cases of robot manipulators are tele-manipulators. These robots are controlled by a human operator. They are used in dangerous environments or distant places (Bolopion and Régnier 2013).

Wheeled mobile robots can be used on smooth ground. They can effectively be used for vision and other sensors assessing distance or contact with objects in the environment. The biologically inspired legged mobile robots usually have six legs and are used on uneven terrain, as in the forestry robot, which is also capable of cutting trees. Another important class is service robotics where robots are used to help people (predominantly aging populations) in daily activities. The most innovative examples are humanoid robots capable of biped locomotion (Lewis et al. 1999). Tripedal robots, quadrupedal robots, and hexapod robots are other increasingly important legged robots. Flying robots (Nonami et al. 2010) and underwater robots (Javier et al. 2013) are broadly used for observation of distant terrains or for ocean studies.

Finally, humans have sought to establish new dimension of human robot communication, interaction, and collaboration. Sophisticated robotic toys are appreciated by children. Interesting experiments can be found in the art where robots are dancing (Augugliaro et al. 2013), playing musical instruments (Cicconet et al. 2013), and even painting (Lewis et al. 1999). Developing Strategies for Robot soccer competitions has achieved highly advanced stages (Wang et al. 2013).

12.2 Robot Kinematic Modelling

Robot arm kinematics is the science that deals with the analytical study of the motion geometry of the robot manipulator with respect to a fixed and reference coordinate system without regard to the forces or the moments causing the movements. In case of Forward Kinematics (FK) the inputs are the joint angle vectors and the link parameters. The output of the forward kinematics is the orientation and the position of the tool or the gripper. When the joint angles that represent the different robot configuration are computed from the position and orientation of the end-effector, the scheme called as Inverse Kinematics (IK) (Hegde 2008). The representation of FK and IK is shown in Fig. 12.2.

In a serial open loop type of robot manipulators, the links connected to no more than two others via joints at the most. Each pair of a link and a joint gives a single degree of freedom (DOF). Every serial manipulator provides "n" degrees of freedom "n DOF." In general, every link 'k' gets connected at the two ends with the previous link $(k-1)$ and the next link $(k+1)$, forming two joints at the ends of connections (as shown in Fig. 12.3), where O_n is the joint center. However, the kinematic analysis of an n-link manipulator can be solved using Denavit–Hartenberg convention for finding the link parameters (Hegde 2008; Spong et al. 2006):

1. The distance (d_i)
2. The angle (θ_i)
3. The length (a_i)
4. The twist angle (α_k)

After coordinate frames assignment to all robot links, according to Denavit–Hartenberg convention, it is possible to establish the relation between the current frames (i) and the next frame $(i+1)$ by the following transformations in sequence:

- Rotation about Z_i by an angle θ_i
- Translate along Z_i by a distance d_i
- Translate along rotated $X_i = X_{i+1}$ through length a_i
- Rotation about X_i by twist angle α_i

This may result in a product of four homogeneous transformations relating coordinate frames (the current frames (i) and the next frame $(i+1)$) of the serially connected two links. The resulted matrix is known as Arm matrix (A) (Hegde 2008).

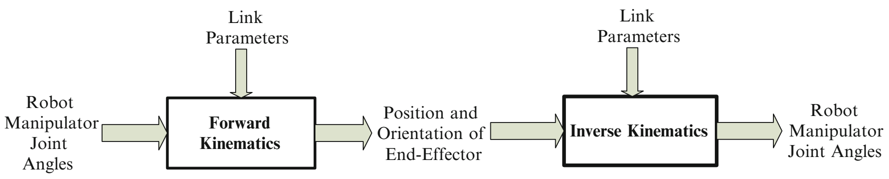
Link Parameters
Robot Manipulator Joint Angles
Forward Kinematics
Position and Orientation of End-Effector
Link Parameters
Inverse Kinematics
Robot Manipulator Joint Angles

Fig. 12.2 Forward and inverse kinematics representation

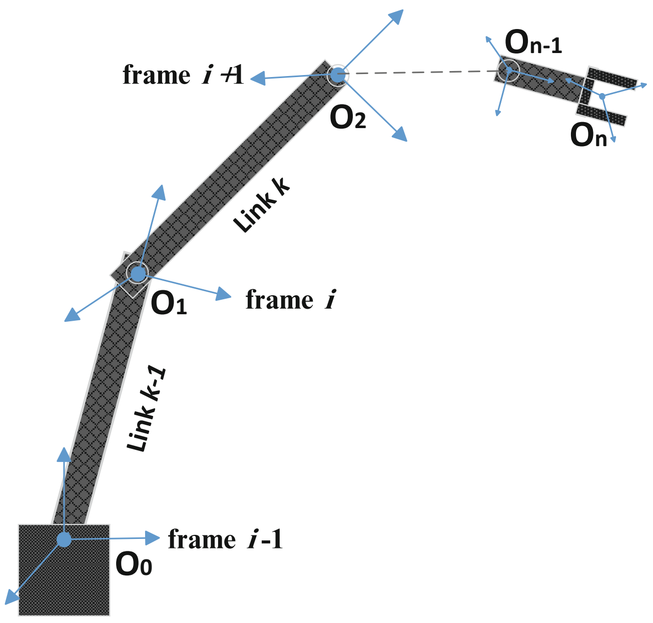

Fig. 12.3 Serial manipulator end-effector frame transformation to base frame

$$A^i_{i+1} = T(z,\ \theta)T_{\text{trans}}(0,\ 0,\ d)T_{\text{trans}}(a,\ 0,\ 0)T(x,\alpha) \tag{12.1}$$

$$= T(z,\ \theta)T_{\text{trans}}(\alpha,\ 0,\ d)T(x,\ \alpha). \tag{12.2}$$

From Equations (12.1) and (12.2) the following matrix may be obtained:

$$\begin{aligned} A^i_{i+1} &= \begin{bmatrix} \cos\theta & -\sin\theta & 0 & 0 \\ \sin\theta & \cos\theta & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & a \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & d \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & \cos\alpha & -\sin\alpha & 0 \\ 0 & \sin\alpha & \cos\alpha & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \\ &= \begin{bmatrix} \cos\theta & -\sin\theta\cos\alpha & \sin\theta\sin\alpha & a\cos\theta \\ \sin\theta & \cos\theta\cos\alpha & -\cos\theta\sin\alpha & a\sin\theta \\ 0 & \sin\alpha & \cos\alpha & d \\ 0 & 0 & 0 & 1 \end{bmatrix}. \end{aligned} \tag{12.3}$$

The coordinate frame at the end-effector of the manipulator is related to the base reference frame by the 'T' matrix in terms of (A) matrices for a six degrees of freedom (6 DOF) robot as example, as follows:

$$T^0_6 = A^0_1\, A^1_2\, A^2_3\, A^3_4\, A^4_5\, A^5_6. \tag{12.4}$$

12.3 Robotic Dynamics: Modelling and Formulations

Dynamics is the science of describing the motion of massive bodies upon application of forces and moments. The motion can be considered as an evolution of the position, orientation, and time derivatives. In robotics, the dynamic equation of motion for manipulators is utilized to set up the fundamental equations for control (Jazar 2007). The dynamic motion of the manipulator arm in a robotic system is produced by the torques generated by the actuators. The relationship between the input torques and the time rates of change of the robot arm components configurations represents the dynamic modelling of the robotic system which is concerned with the derivation of the equations of motion of the manipulator as a function of the forces and moments acting on it. So, the dynamic modelling of a robot manipulator consists of finding the mapping between the forces exerted on the structures and the joint positions, velocities, and accelerations (Canudas et al. 1996).

Beni and Hackwood (1985) note that a dynamic analysis of a manipulator is useful for the following purposes:

1. It determines the joint forces and torques required to produce specified end-effector motions (the direct dynamic problem).
2. It produces a mathematical model which simulates the motion of the manipulator under various loading conditions (the inverse dynamic problem) and/or control schemes.
3. It provides a dynamic model for use in the control of the actual manipulator.

Equations of motion for the manipulator can be obtained by forming Euler–Lagrange's equation on the basis of Lagrange's energy function. The resulting differential equations describe the motion in terms of the joint variables and parameters of the manipulator.

Let K and V be the total kinetic energy and potential energy stored in the dynamic system. The Lagrangian is defined by (Spong et al. 2006; Min et al. 1992) and (Yamamoto 1992):

$$L(q,\dot{q}) = K(q,\dot{q}) - V(q). \tag{12.5}$$

Using the Lagrangian equations of motion as obtained by Yamamoto (1992):

$$\frac{\mathrm{d}}{\mathrm{d}t}\frac{\partial L}{\partial \dot{q}} - \frac{\partial L}{\partial q} = Q_i, \quad i = 1, \ldots, n, \tag{12.6}$$

where Q_i is the generalized force corresponding to the generalized coordinate q_i. The kinetic energy and potential energy for the link i are given (Spong et al. 2006):

$$K_i = \frac{1}{2}\ \text{trace}\left[\sum_{j=1}^{i}\sum_{k=1}^{i}\frac{\partial T}{\partial q_i}J_i\frac{\partial T}{\partial q_k}\ \dot{q}_j\dot{q}_k\right], \tag{12.7}$$

$$V_i = -m_i g^{\mathrm{T}} T_i r^{-(i)}. \tag{12.8}$$

The Lagrangian motion equations for the *n*th link manipulators can be represented as a second-order nonlinear differential equation (Spong et al. 2006):

$$\sum_{j=1}^{n} B_{ij}\ddot{q}_j + \sum_{j=1}^{n}\sum_{k=1}^{n} C_{ijk}\dot{q}_j\dot{q}_j + g_i = Q_i, \quad i = 1, \ldots, n, \tag{12.9}$$

where

$$B_{ij} = \sum_{k=\max(i,j)}^{n} \operatorname{trace}\left[\frac{\partial T_k}{\partial q_i} J_k \frac{\partial T_k^{\mathrm{T}}}{\partial q_i}\right], \tag{12.10}$$

$$C_{ijk} = \sum_{h=\max(i,j,k)}^{n} \operatorname{trace}\left[\frac{\partial T_h}{\partial q_i} J_n \frac{\partial^2 T_h^{\mathrm{T}}}{\partial q_i \partial q_k}\right], \tag{12.11}$$

$$g_i = \sum_{k=i}^{n} m_k g^{\mathrm{T}} \frac{\partial T_k}{\partial q_{ki}} r^{-(i)}. \tag{12.12}$$

Equation (12.9) can be written as a set of second-order vector differential equations:

$$\mathbf{B}(\mathbf{q})\ddot{\mathbf{q}} + \mathbf{C}(\mathbf{q}, \dot{\mathbf{q}}) + \mathbf{g}(\mathbf{q}) = \mathbf{Q_i}, \tag{12.13}$$

where $\mathbf{B}(\mathbf{q})$ is the symmetric inertia matrix $\mathbf{C}(\mathbf{q}, \dot{\mathbf{q}}\cdot)$, the matrix of Coriolis and centrifugal effects, the vector $\mathbf{g}(\mathbf{q})$ denotes the gravity terms, and $\mathbf{Q_i}$ is the generalized force vector.

12.4 Path and Trajectory Planning in Robotics

The definition of the path is the sequence of robot configurations in a particular order without regard to the timing of these configurations. The path planning as a process is the planning of the whole way from the start point to the goal point, including stopping in defined path points. While the trajectory is, the path specified by the time law requirement to move the robot from the starting point to the goal point. In the methodologies of trajectory planning, the task is to attain a specific target from an initial starting point, avoid obstacles, and stay within robot capabilities. Trajectories can be planned either in joint space where the time evolution of the joint angles is specified directly or in Cartesian space specifying the position and orientation of the end-effector frame. In joint space trajectory planning, the joint values that satisfy these robot configurations can be calculated and used by the controller for driving the joints to the desired and new positions. Joint space planning approach is more simple and faster than Cartesian space planning approach because of the inverse kinematics calculations avoidance. Obstacle avoidance is difficult in joint space because of the end-effector pose is not directly controlled, while planning in Cartesian space allows directly satisfying the geometric constraints of the robot work space, but then inverse kinematics must be solved (Niku 2001).

12.4.1 Third-Order Polynomial Trajectory Planning

In this planning method, the problem is to find a trajectory that connects the initial and final configuration. We find the final joint angles for the desired position and orientation using the inverse kinematic equations. Considering one of the joints, which at the beginning of the motion segment at time t_i is at θ_i and has to move to a new value of θ_f at time t_f. Third-order polynomials can be used to plan the trajectory, such that the velocities at the beginning and the end of the motion are zero or other known values (Niku 2001; Spong et al. 2006).

$$\theta(t) = c_0 + c_1 t + c_2 t^2 + c_3 t^3,$$

where the initial and final conditions are as follows:

$$\theta(t_\mathrm{i}) = \theta_\mathrm{i}; \quad \theta(t_\mathrm{f}) = \theta_\mathrm{f}; \quad \dot{\theta}(t_\mathrm{i}) = 0; \quad \dot{\theta}(t_\mathrm{f}) = 0.$$

Taking the first derivative of the polynomial equation:

$$\theta(t_\mathrm{i}) = c_0 = \theta_\mathrm{i}; \quad \theta(t_\mathrm{f}) = c_0 + c_1 t_\mathrm{f} + c_2 t_\mathrm{f}^2 + c_3 t_\mathrm{f}^3; \quad \dot{\theta}(t_\mathrm{i}) = c_1 = 0; \quad \dot{\theta}(t_\mathrm{f}) \\ = c_1 + 2c_2 t_\mathrm{f} + 3c_3 t_\mathrm{f}^2 = 0.$$

Solving these four equations simultaneously, we get the necessary values for the constants as follows (Niku 2001):

$$\begin{gathered} \theta(t) = c_0 + c_1 t + c_2 t^2 + c_3 t^3; \\ \dot{\theta}(t) = c_1 + 2c_2 t + 3c_3 t^2. \\ \text{At}\, t = 0 \quad (t = t_\mathrm{i} = \text{Zero}) \quad \rightarrow \quad \theta(0) = c_0 = \theta_\mathrm{i}; \quad \dot{\theta}(0) = c_1 \\ = 0. \quad (\text{initial velocity} = \text{Zero}). \\ \text{At}\, t = t_\mathrm{f}: \quad \rightarrow \quad \theta(t_\mathrm{f}) = \theta_\mathrm{f} = \theta_\mathrm{i} + c_2 t_\mathrm{f}^2 + c_3 t_\mathrm{f}^3 \quad \rightarrow \quad c_2 t_\mathrm{f}^2 + c_3 t_\mathrm{f}^3 \\ = \theta_f - \theta_\mathrm{i}. \\ \dot{\theta}(t_\mathrm{f}) = 2c_2 t_\mathrm{f} + 3c_3 t_\mathrm{f}^2 = 0 \quad (\text{Zero velocity at the final time } t_\mathrm{f}) \end{gathered} \tag{12.14}$$

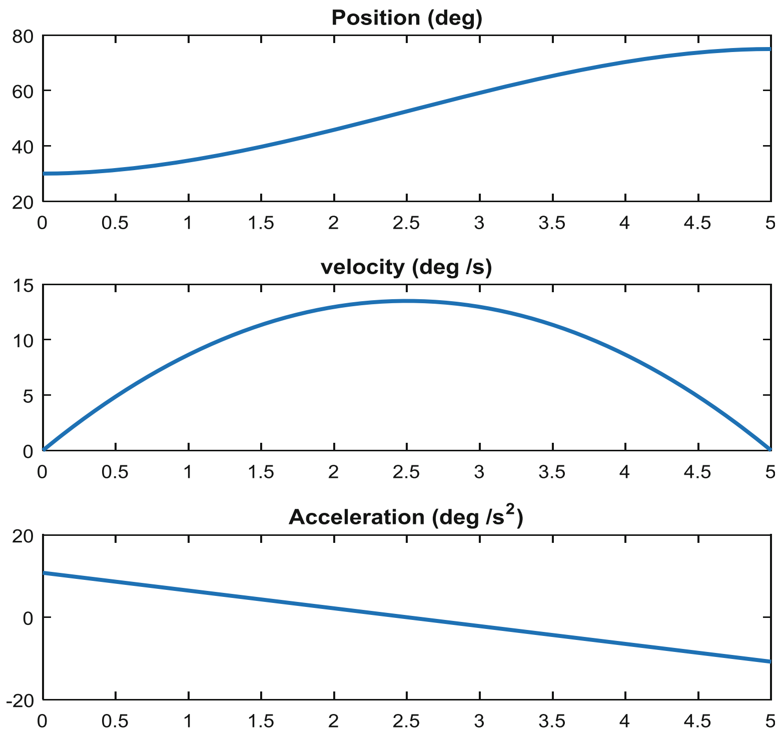

Fig. 12.4 Joint positions, velocities, and accelerations using third-order polynomial equation

$$\rightarrow\ 2c_2 t_\mathrm{f} + 3c_3 t_\mathrm{f}^2 = 0. \tag{12.15}$$

Solving Eqs. (12.14) and (12.15) gives:

$$c_3 = \frac{-2(\theta_\mathrm{f} - \theta_\mathrm{i})}{t_\mathrm{f}^3}; \quad c_2 = \frac{3(\theta_\mathrm{f} - \theta_\mathrm{i})}{t_\mathrm{f}^2}.$$

As an example, Fig. 12.4 shows the joint positions, velocities, and accelerations using third-order polynomial equation.

12.4.2 *Linear Segments with Parabolic Blends*

This method is a joint space trajectory planning with a trapezoidal velocity profile in which to run the joints at constant speed between the initial and final configurations. Achieving this method in order to create a smooth path, we design the desired trajectory in three parts. Starting with the linear segment and adding a parabolic blend region (quadratic polynomial parts) at the beginning and the end of the

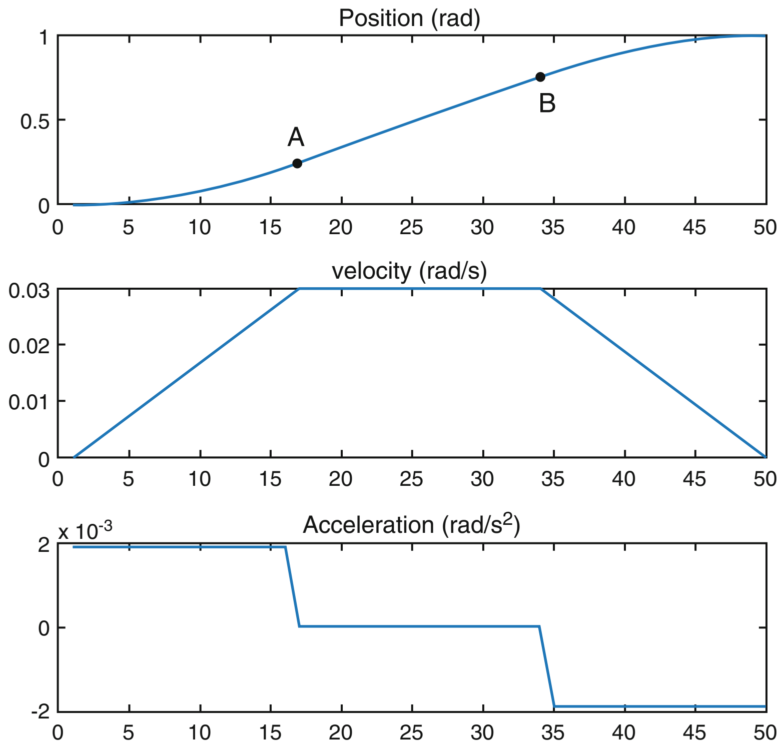

Fig. 12.5 Joint positions, velocities, and accelerations using linear segments with parabolic blends

motion segment for creating continuous position and velocity, as shown in Fig. 12.5. The linear segment is the trajectory segment between points A and B. Assuming that the initial and the final positions are θ_i and θ_f at time $t_\mathrm{i} = 0$ and t_f and that the parabolic segments are symmetrically blended with the linear section at blending times t_b and $t_\mathrm{f} - t_\mathrm{b}$, we can write (Niku 2001; Spong et al. 2006):

$$\theta_\mathrm{i} = \theta(t = 0) = \theta(0) = \text{initial position}, \quad \theta_\mathrm{f} = \theta(t = t_\mathrm{f}) = \theta(t_\mathrm{f}) = \text{final position}.$$
$$t_\mathrm{i} = 0(\text{starting position}), \quad t_\mathrm{f} = \text{final time (ending time)}.$$

For the first parabolic segment:

$$\theta(t) = c_0 + c_1 t + \frac{1}{2} c_2 t^2;$$
$$\dot{\theta}(t) = c_1 + c_2 t;$$
$$\ddot{\theta}(t) = c_2.$$

The acceleration is constant for the parabolic sections, yielding a continuous velocity at the common points (called knot points) ***A*** and ***B***. Substituting the boundary conditions into the parabolic equation segment yields (Niku 2001):

$$\begin{gathered}
\text{At } t=0 \quad \rightarrow \quad \theta(0)=c_0 \quad \rightarrow \quad c_0=\theta_i; \\
\dot{\theta}(0)=c_1=0 \text{ (starting velocity} = 0); \\
\ddot{\theta}(0)=c_2 \quad \rightarrow \quad c_2=\ddot{\theta}.
\end{gathered}$$

Substituting the initial conditions gives parabolic segments in the form:

$$\begin{gathered}
\theta(t)=\theta_i+\frac{1}{2}c_2t^2; \\
\dot{\theta}(t)=c_2t; \\
\ddot{\theta}(t)=c_2.
\end{gathered}$$

For the linear segment the velocity will be constant and can be chosen based on the physical capabilities of the actuators. Substituting zero initial velocity, a constant known joint velocity ω in the linear portion, and zero final velocity, the joint positions and velocities for points A, B and the final point as follows (Niku 2001):

The general linear equation is,

$$\begin{gathered}
\frac{y}{x}=\frac{y_2-y_1}{x_2-x_1} \quad \rightarrow \quad \frac{\theta}{t}=\frac{\theta_B-\theta_A}{t_f-t_b-t_b}; \\
\frac{\theta}{t}=\omega \quad \rightarrow \quad \omega=\frac{\theta_B-\theta_A}{t_f-2t_b} \\
\rightarrow \theta_B=\theta_A+\omega(t_f-2t_b). \\
\text{At } t=t_b.
\end{gathered}$$

Because of point A = the end of the first parabolic segment = the start of the linear segment, then the value of θ_A can be found from the end point of the first parabolic segment, so that:

$$\begin{gathered}
\theta_A=\theta_i+\frac{1}{2}c_2t_b^2; \\
\dot{\theta}_A=c_2t_b=\omega \,(\text{constant velocity at the linear segment}).
\end{gathered}$$

The necessary blending time t_b can be found as follows:

$$\begin{gathered}
\theta_f=\theta_B+(\theta_A-\theta_i); \\
\theta_f=\theta_A+\omega(t_f-2t_b)+\theta_A-\theta_i; \qquad (12.16)
\end{gathered}$$

$$\theta_A = \theta_i + \frac{1}{2}c_2 t_b^2; \tag{12.17}$$

$$\theta_f = 2\left(\theta_i + \frac{1}{2}c_2 t_b^2\right) - \theta_i + \omega(t_f - 2t_b);$$

$$\theta_f = \theta_i + c_2 t_b^2 + \omega(t_f - 2t_b);$$

$$c_2 = \frac{\omega}{t_b} \tag{12.18}$$

$$\rightarrow \theta_f = \theta_i + \left(\frac{\omega}{t_b}\right) t_b^2 + \omega(t_f - 2t_b);$$

$$\theta_f = \theta_i + \omega t_b + \omega t_f - 2\omega t_b.$$

Then calculating the blending time as:

$$t_b = \frac{\theta_i - \theta_f + \omega t_f}{\omega}.$$

The time t_b cannot be bigger than half of the total time t_f which results in a parabolic speedup and a parabolic slowdown. With no linear segment, a corresponding maximum velocity (Niku 2001):

$$\omega_{\max} = 2(\theta_f - \theta_i)/t_f.$$

The final parabolic segment is symmetrical with the initial parabola, but with a negative acceleration, and thus can be expressed as follows (Niku 2001):

$$\theta(t) = \theta_f - \frac{1}{2}c_2(t_f - t)^2, \quad \text{where} \quad c_2 = \frac{\omega}{t_b},$$

$$\rightarrow \begin{cases} \theta(t) = \theta_f - \dfrac{\omega}{2t_b}(t_f - t)^2, \\ \dot{\theta}(t) = \dfrac{\omega}{t_b}(t_f - t), \\ \ddot{\theta}(t) = -\dfrac{\omega}{t_b}. \end{cases}$$

12.5 Classical Control Synthesis and Design

The problem of robot control can be described as a computation of the forces or torques that must be generated by the actuators in order to successfully accomplish the robot task. The robot task can be presented either as the accomplishment 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 (Bajd et al. 2010).

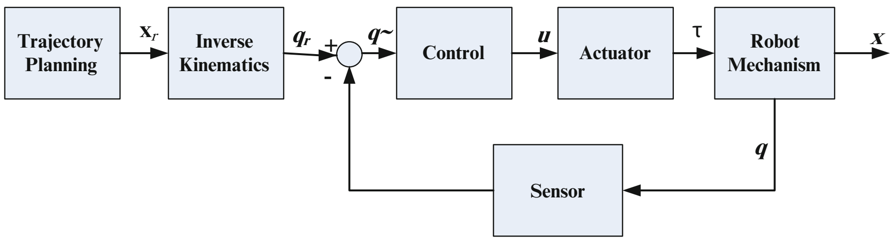

Fig. 12.6 A common robot control system (Bajd et al. 2010)

Control is used to move the robot with respect to the environment as well as to articulate sensor heads, arms, grippers, tools, and implements. Several techniques can be employed for controlling a robot, as in Samad (2001). The choice of the control method depends on the robot task.

A general robot control system consists of the following components: path planning, inverse kinematics, and a closed loop system that contains the controller, actuator, the robot mechanism, and the sensor, as shown in Fig. 12.6. The input to the control system is the desired pose of the robot end-effector, which can be obtained by using trajectory interpolation methods. The variable $\mathbf{x_r}$ represents the reference pose of the robot end-effector. The $\mathbf{x}$ vector, describes the actual pose of the robot end-effector, in general this 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. Accordingly,

$$\mathbf{x} = [x \quad y \quad z \quad \varphi \quad \vartheta \quad \psi]^{\mathrm{T}}.$$

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). The internal coordinates $\boldsymbol{q}_{\mathrm{r}}$ represent the desired end-effector 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. Based on the positional error $\tilde{\boldsymbol{q}}$, the control system output $\boldsymbol{u}$ is calculated. The actuators ensure the forces or torques necessary for the required robot motion. The robot motion is measured by the sensors.

12.5.1 PD Position Control

For position control of a robot, a Proportional Derivative (PD) is commonly designed. For robot control this closed loop is separate for each particular degree of freedom. The control method 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

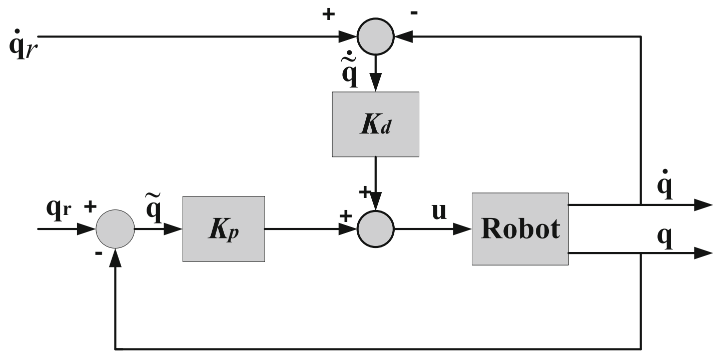

Fig 12.7 PD position control (Bajd et al. 2010)

many controllers are to be developed as there are degrees of freedom. A robot manipulator has several degrees of freedom, therefore the error $\tilde{\boldsymbol{q}} = \boldsymbol{q}_\mathrm{r} - \boldsymbol{q}$ can be stated as a vector, whereas $\boldsymbol{K_p}$ is a diagonal matrix of the gains of all joint controllers. The calculated control input incites robot motion toward reduction of the positional error. The positional error is characterized by the position error $\tilde{\boldsymbol{q}}$ multiplied by $\boldsymbol{K_p}$ (whereas $\boldsymbol{K_p}$ is a diagonal matrix of the gains of all joint controllers). In addition, to confirm safe and stable robot actions, velocity in closed loop mode is presented with a negative sign. The velocity in closed loop mode brings damping into the system. It is characterized by the actual joint velocities $\dot{\boldsymbol{q}}$ multiplied by a diagonal matrix of velocity gains $\boldsymbol{K_d}$. The overall control law can be obtained by combining the positional error and the velocity error as given in the following form:

$$\boldsymbol{u} = \boldsymbol{K}_p(\boldsymbol{q}_r - \boldsymbol{q}) + \boldsymbol{K}_d\left(\dot{\boldsymbol{q}}_r - \dot{\boldsymbol{q}}\right), \tag{12.19}$$

where $\dot{\boldsymbol{q}}_r - \dot{\boldsymbol{q}}$ is the velocity error $\dot{\tilde{\boldsymbol{q}}}$. In Eq. (12.19), the reference velocity signal is included in the PD signal in order to avoid unnecessary high damping at fastest part of the trajectory. The synthesis of the PD position controller involves the determination of the matrices $\boldsymbol{K}_p$ and $\boldsymbol{K}_d$. The $\boldsymbol{K}_p$ gains must be high for faster response. On the other hand by proper choice of the $\mathbf{K}_d$ gains, fast response without overshoot for the robot systems is gained. Figure 12.7 illustrates the PD position control configuration.

12.5.2 PD Control of Position with Gravity Compensation

The robot mechanism is usually known to be under the influence of inertial, Coriolis, centripetal, and gravitational forces. For a simplified model, viscous

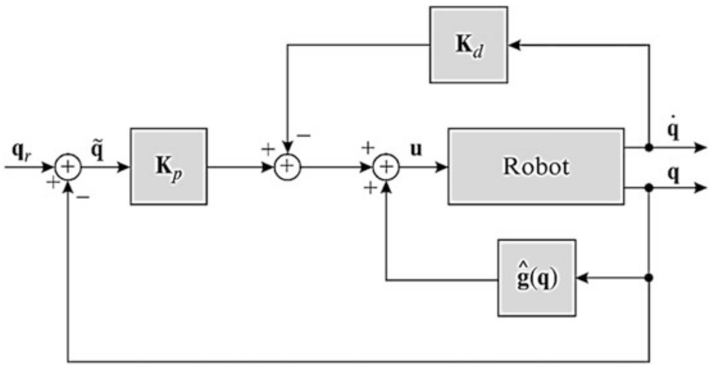

Fig. 12.8 PD control with gravity compensation (Bajd et al. 2010)

friction, which is proportional to the joint velocity, will be considered here. Consequently, robot dynamics of Eq. (12.13) can be rewritten as follows:

$$\boldsymbol{B}(\boldsymbol{q})\ddot{\boldsymbol{q}} + \boldsymbol{C}(\boldsymbol{q},\dot{\boldsymbol{q}})\dot{\boldsymbol{q}} + \boldsymbol{F}_v\dot{\boldsymbol{q}} + \boldsymbol{g}(\boldsymbol{q}) = \boldsymbol{\tau}, \tag{12.20}$$

where $\boldsymbol{\tau}$ is the torques in the robot joints. **B**, **C**, and $\boldsymbol{g}$ are defined in Sect. 11.2, $\mathbf{F}_v$ is a diagonal matrix of the joint friction coefficients (Bajd et al. 2010).

In quasi-static conditions, when the robot is moving slowly, it can be assumed that zero accelerations $\ddot{\boldsymbol{q}} \approx 0$ and velocities $\dot{\boldsymbol{q}} \approx 0$. Accordingly, the robot dynamic model is simplified as

$$\boldsymbol{\tau} \approx \boldsymbol{g}(\boldsymbol{q}). \tag{12.21}$$

The model of gravitational effects $\widehat{\mathbf{g}}(\mathbf{q})$ (the circumflex denotes the robot model), which is an acceptable approximation of the actual gravitational forces $\mathbf{g}(\mathbf{q})$. The control algorithm shown in Fig. 12.8 can be written as follows:

$$\boldsymbol{u} = \boldsymbol{K}_p(\boldsymbol{q}_r - \boldsymbol{q}) - \boldsymbol{K}_d\dot{\boldsymbol{q}} + \widehat{\boldsymbol{g}}(\boldsymbol{q}). \tag{12.22}$$

By introducing gravity compensation, the errors in trajectory tracking are significantly reduced. In addition, this control method can be extended to consider the effect of motion of the robot end-effector; starting from the positional error of the robot end-effector which is calculated as:

$$\tilde{\boldsymbol{x}} = \boldsymbol{x}_r - \boldsymbol{x} = \boldsymbol{x}_r - \boldsymbol{k}(\boldsymbol{q}), \tag{12.23}$$

where $\mathbf{x_r}$ is the reference pose of the robot end-effector and **k**(.) represents the equations of direct kinematics.

The velocity of the robot end point is calculated with the help of the Jacobian matrix from the joint velocities. The equation describing the PD controller is:

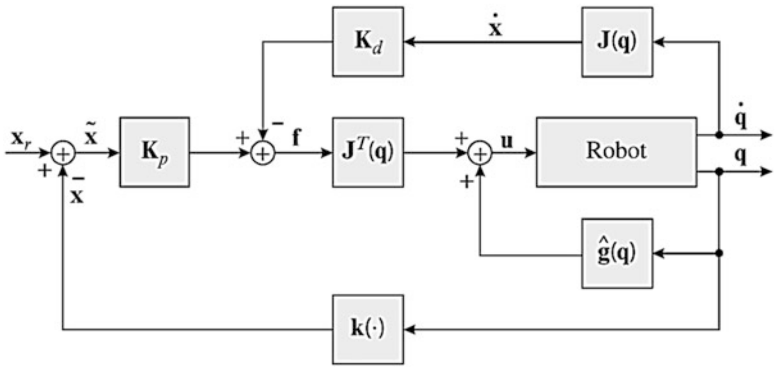

Fig. 12.9 PD control with gravity compensation in external coordinates (Bajd et al. 2010)

$$\mathbf{f} = \mathbf{K_p}\tilde{\mathbf{x}} - \mathbf{K_d}\dot{\mathbf{x}}. \tag{12.24}$$

The negative sign of the velocity error introduces damping into the system. The joint torques are calculated from the force $\boldsymbol{f}$, acting at the tip of the robot, with the help of the transposed Jacobian matrix and by adding the component compensating gravity. The control algorithm is written as follows:

$$\mathbf{u} = \mathbf{J^T}(\mathbf{q})\mathbf{f} + \widehat{\mathbf{g}}(\mathbf{q}). \tag{12.25}$$

The complete control scheme is shown in Fig. 12.9.

12.5.3 Control of the Robot Based on Inverse Dynamics

This control scheme can be derived from the robot dynamic model described by Eq. (12.20). Assume that the torques $\boldsymbol{\tau}$, generated by the motors, are equal to the control outputs $\boldsymbol{u}$. Rewrite Eq. (12.20) in order to determine the direct robot dynamic model, which describes robot motions under the influence of the given joint torques. Accordingly, the acceleration $\ddot{\boldsymbol{q}}$ can be expressed in short as follows (Bajd et al. 2010):

$$\ddot{\boldsymbol{q}} = \boldsymbol{B}^{-1}(\boldsymbol{q})(\boldsymbol{u} - \boldsymbol{n}(\boldsymbol{q}, \dot{\boldsymbol{q}})), \tag{12.26}$$

where $\boldsymbol{n}(\boldsymbol{q}, \dot{\boldsymbol{q}})$ comprising all dynamic components except the inertial component, i.e.

$$\mathbf{n}(\mathbf{q}, \dot{\mathbf{q}}) = \mathbf{C}(\mathbf{q}, \dot{\mathbf{q}})\dot{\mathbf{q}} + \mathbf{F_v}\dot{\mathbf{q}} + \mathbf{g}(\mathbf{q}). \tag{12.27}$$

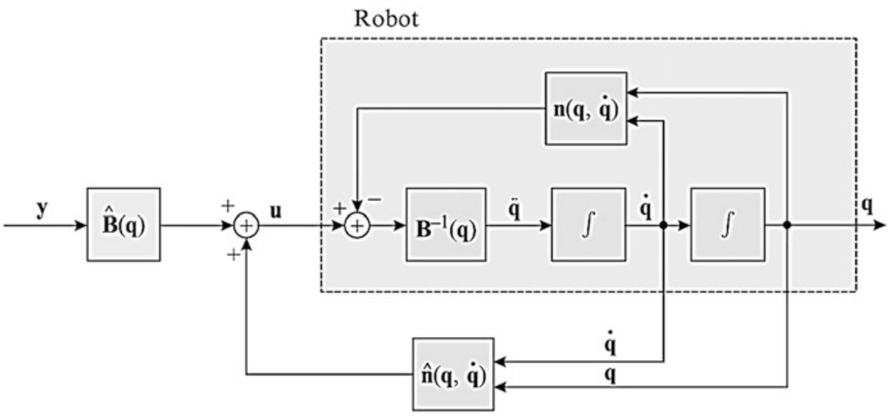

Fig. 12.10 Linearization of the control system by implementing the inverse dynamic model (Bajd et al. 2010)

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

$$\widehat{\mathbf{n}}\ (\mathbf{q}, \dot{\mathbf{q}}) = \widehat{\mathbf{C}}(\mathbf{q}, \dot{\mathbf{q}})\dot{\mathbf{q}} + \widehat{\mathbf{F}}_{\mathbf{v}}\dot{\mathbf{q}} + \widehat{\mathbf{g}}(\mathbf{q}). \tag{12.28}$$

The controller output $\boldsymbol{u}$ is based on inverse dynamics as in the following equation:

$$\mathbf{u} = \widehat{\mathbf{B}}(\mathbf{q})\mathbf{y} + \widehat{\mathbf{n}}(\mathbf{q}, \dot{\mathbf{q}}), \tag{12.29}$$

where the approximate inverse dynamic model of the robot was used. The system, combining Eqs. (12.26) and (12.29), is shown in Fig. 12.10.

By simple substitutions we can write the vector $\mathbf{y}$, having the acceleration characteristics:

$$\mathbf{y} = \ddot{\mathbf{q}}_{\mathbf{r}} + \mathbf{K}_{\mathbf{p}}(\mathbf{q}_{\mathbf{r}} - \mathbf{q}) + \mathbf{K}_{\mathbf{d}}(\dot{\mathbf{q}}_{\mathbf{r}} - \dot{\mathbf{q}}). \tag{12.30}$$

It consists of the reference acceleration $\ddot{\boldsymbol{q}}_r$ and two contributing signals which depend on the errors of position and velocity. These two signals suppress the error arising because of the imperfectly modelled dynamics. The complete control scheme is shown in Fig. 12.11. By considering Eq. (12.30) and the equality $y = \ddot{\boldsymbol{q}}$, the differential equation describing the robot dynamics can be written as follows:

$$\ddot{\tilde{\boldsymbol{q}}} + \boldsymbol{K}_d\dot{\tilde{\boldsymbol{q}}} + \boldsymbol{K}_p\tilde{\boldsymbol{q}} = 0, \tag{12.31}$$

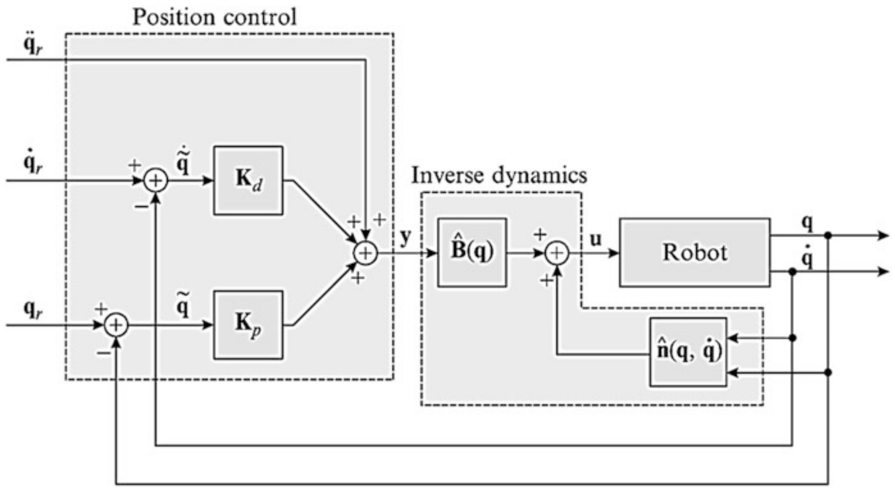

Fig. 12.11 Control of the robot based on inverse dynamics (Bajd et al. 2010)

where the acceleration error $\ddot{\tilde{q}} = \ddot{q}_r - \ddot{q}$ was introduced. The differential equation (12.31) 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 $\mathbf{K}_d$.

Similar to internal coordinates, the derivation of equation that describe the dynamics of the control error, an analogous equation can be written for the error of the end-effector pose. Accordingly, the acceleration $\ddot{\mathbf{x}}$ of the robot end-effector can be expressed as follows:

$$\ddot{\tilde{\mathbf{x}}} + \mathbf{K_d}\dot{\tilde{\mathbf{x}}} + \mathbf{K_p}\tilde{\mathbf{x}} = 0 \quad \Rightarrow \quad \ddot{\mathbf{x}} = \ddot{\mathbf{x}}_\mathbf{r} + \mathbf{K_d}\dot{\tilde{\mathbf{x}}} + \mathbf{K_p}\tilde{\mathbf{x}}. \tag{12.32}$$

Taking into account the equality $\mathbf{y} = \ddot{\mathbf{q}}$

$$\mathbf{y} = \mathbf{J}^{-1}(\mathbf{q})\left(\ddot{\mathbf{x}} - \dot{\mathbf{J}}(\mathbf{q}, \dot{\mathbf{q}})\dot{\mathbf{q}}\right). \tag{12.33}$$

Substituting $\ddot{\mathbf{x}}$ in Eq. (12.33) with expression (12.32), the control algorithm based on inverse dynamics in the external coordinates is obtained as follows:

$$\mathbf{y} = \mathbf{J}^{-1}(\mathbf{q})\left(\ddot{\mathbf{x}}_r + \boldsymbol{K}_d\dot{\tilde{\mathbf{x}}} + \mathbf{K_p}\tilde{\mathbf{x}} - \dot{\mathbf{J}}(\mathbf{q}, \dot{\mathbf{q}})\dot{\mathbf{q}}\right). \tag{12.34}$$

The control scheme encompassing the linearization of the system based on inverse dynamics (12.29) and the closed-loop control (12.34) is shown in Fig. 12.12.

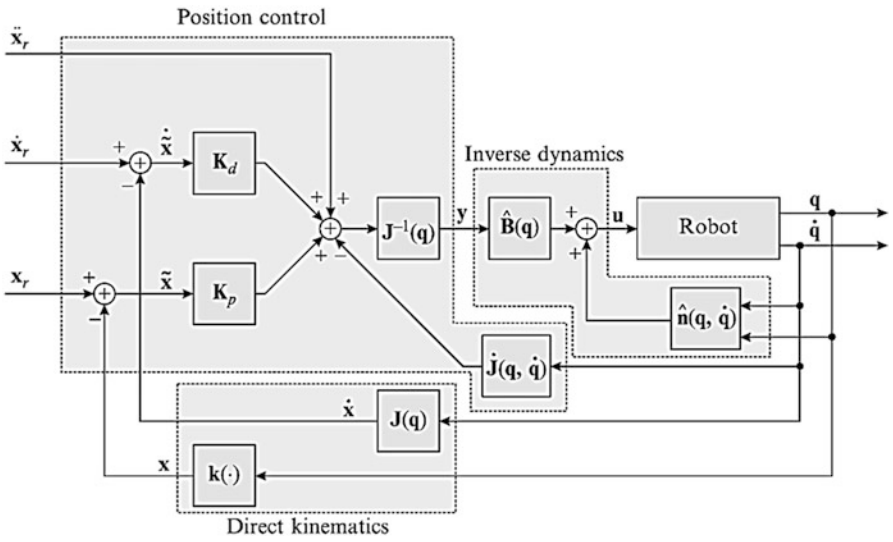

Fig. 12.12 Robot control based on inverse dynamics in external coordinates (Bajd et al. 2010)

12.5.4 Control Based on the Transposed Jacobian Matrix

This control method is based on the relation 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:

$$\boldsymbol{\tau} = \mathbf{J}^{\mathrm{T}}(\mathbf{q})\mathbf{f}, \tag{12.35}$$

where the vector $\boldsymbol{\tau}$ represents the joint torques and $\boldsymbol{f}$ is the force at the robot endpoint. The aim is 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 $\boldsymbol{x}$. Robots are usually provided with sensors that measure the joint variables. The pose of the robot end-effector must be therefore determined by using the direct kinematic model $\boldsymbol{x} = \boldsymbol{k(q)}$, where $\boldsymbol{q}$ indicates the vector of joint variables, $\boldsymbol{x}$ indicates the vector of task variables; usually, three position coordinates and three Euler angles The positional error of the robot end-effector $(\tilde{\boldsymbol{x}} = \boldsymbol{x}_r - \boldsymbol{x})$ must be reduced to zero. A simple proportional control system with the gain matrix $\mathbf{K}_p$ (Bajd et al. 2010) is introduced:

$$\mathbf{f} = \mathbf{K}_{\mathbf{p}}\tilde{\boldsymbol{x}}\,. \tag{12.36}$$

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 $\boldsymbol{f}$. This calculation is performed using the transposed Jacobian matrix in Eq. (12.35).

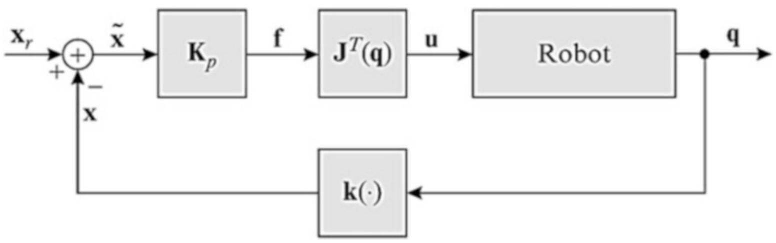

Fig. 12.13 Control based on the transposed Jacobian matrix (Bajd et al. 2010)

$$\mathbf{u} = \mathbf{J}^{\mathbf{T}}(\mathbf{q})\mathbf{f}. \tag{12.37}$$

The vector $\boldsymbol{u}$ represents the desired joint torques. The control method based on the transposed Jacobian matrix is shown in Fig. 12.13.

12.5.5 Control Based on the Inverse Jacobian Matrix

In this method, the control is based on the relation between the joint velocities and the velocities of the robot end point, which is known as the Jacobian matrix (Bajd et al. 2010).

$$\dot{\mathbf{x}} = \mathbf{J}(\mathbf{q})\dot{\mathbf{q}}. \tag{12.38}$$

For small displacements, the relation between changes of the internal coordinates and changes of the pose of the robot end point can be expressed as follows:

$$\mathbf{dx} = \mathbf{J}(\mathbf{q})\mathbf{dq}. \tag{12.39}$$

For small error in the pose, we can calculate the positional error in the internal coordinates by the inverse relation (Eq. 12.39).

$$\tilde{\mathbf{q}} = \mathbf{J}^{-1}(\mathbf{q})\tilde{\mathbf{x}}. \tag{12.40}$$

In this way, the control method is translated to the known method of robot control in the internal coordinates. The control method, based on the inverse Jacobian matrix, is shown in Fig. 12.14.

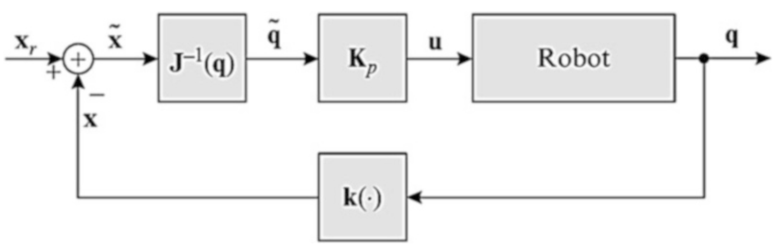

Fig. 12.14 Control based on the inverse Jacobian matrix (Bajd et al. 2010)

12.5.6 *Control of the Contact Force*

The robot force control method is based on control of the robot using inverse dynamics. A contact force f appears in the inverse dynamic model due to the interaction of the robot with the environment. As the forces acting at the robot end-effector are transformed into the joint torques by the use of the transposed Jacobian matrix, we can write the robot dynamic model in the following form (Bajd et al. 2010)

$$\mathbf{B}(\mathbf{q})\ddot{\mathbf{q}} + \mathbf{C}(\mathbf{q},\dot{\mathbf{q}})\dot{\mathbf{q}} + \mathbf{F}_v\dot{\mathbf{q}} + \mathbf{g}(\mathbf{q}) = \boldsymbol{\tau} - \mathbf{J}^T(\mathbf{q})\mathbf{f}. \tag{12.41}$$

It can be seen that the force f acts through the transposed Jacobian matrix in a similar way as the joint torques, i.e., it tries to produce robot motion.

12.5.6.1 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 $\boldsymbol{u}$. The direct dynamic model was described by Bajd et al. (2010)

$$\ddot{\mathbf{q}} = \mathbf{B}^{-1}(\mathbf{q})\big(\mathbf{u} - \mathbf{n}(\mathbf{q},\dot{\mathbf{q}}) - \mathbf{J}^T(\mathbf{q})\mathbf{f}\big). \tag{12.42}$$

Equation (12.24) describes the response of the robot system to the control input $\boldsymbol{u}$. Taking into account the initial velocity value, the actual velocity of the robot motion is obtained by integrating the acceleration. While taking into the account the initial position, the actual positions in the robot joints are calculated by integrating the velocity. The described model is represented by the block *Robot* in Fig. 12.15. The system is linearized by including the inverse dynamic model into the closed loop:

$$\mathbf{u} = \widehat{\mathbf{B}}(\mathbf{q})\mathbf{y} + \widehat{\mathbf{n}}(\mathbf{q},\dot{\mathbf{q}}) + \mathbf{J}^T(\mathbf{q})\mathbf{f}. \tag{12.43}$$

The use of circumflex denotes the estimated parameters of the robot system.

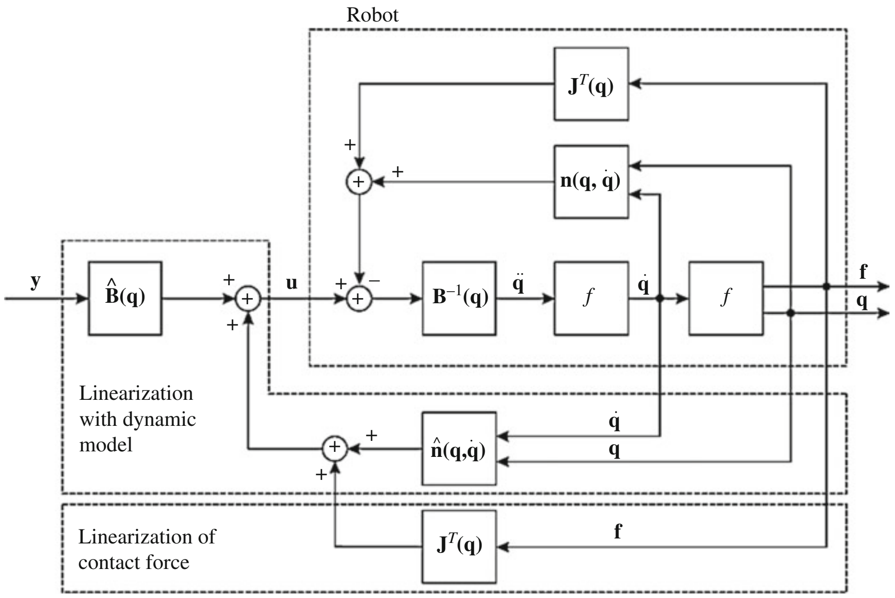

Fig. 12.15 Linearization of the control system by implementing the inverse dynamic model and the measured contact force (Bajd et al. 2010)

12.5.6.2 Force Control

Based on linearized model of the control system, the force control is translated to control the pose of the end-effector. If it is required from 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. The following control system by Bajd et al. (2010) can be used:

$$\boldsymbol{y} = \boldsymbol{J}^{-1}(\boldsymbol{q})\left(\ddot{\boldsymbol{x}}_r + \boldsymbol{K}_d\dot{\tilde{\mathbf{x}}} + \boldsymbol{K}_P\tilde{\mathbf{x}} - \dot{\boldsymbol{J}}(\boldsymbol{q},\dot{\boldsymbol{q}})\dot{\boldsymbol{q}}\right). \tag{12.44}$$

Accordingly, the control of the robot end-effector (including the linearization) while taking into account the contact force can be determined. This is summarized in Fig. 12.16.

Generally, for appropriate handling of interactions between robot and environment, it is necessary to consider force control strategies, either in an indirect way by means of a suitable use of position control laws or in a direct way by means of true force control laws (Samad 2001).

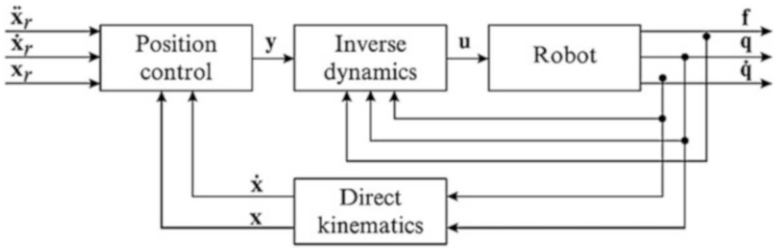

Fig. 12.16 Robot control based on inverse dynamics in external coordinates including the contact force (Bajd et al. 2010)

12.6 Robot Vision and Visual Servoing

Vision technology gives robots intelligent eyes. Using these eyes, robots can recognize the position of objects in space and adjust their working steps accordingly. The benefits of sophisticated vision technology include savings, improved quality, reliability, safety, and productivity. Robot vision is used for part identification and navigation. Vision applications generally deal with finding a part and orienting it for robotic handling or inspection before an application is performed. Sometimes vision-guided robots can replace multiple mechanical tools with a single robot station.

12.6.1 Robot Vision

Recognizing the geometry of the robot workspace from a digital image (Fig. 12.17) is the main task of robot vision which is solved by finding the relation between the coordinates of a point in the two-dimensional (2D) image and the coordinates of the point in the real three-dimensional (3D) robot environment. The basic equations of optics determine the position of a point in the image plane with respect to the corresponding point in 3D space. We will therefore find the geometrical relation between the coordinates of the point $P(x_c,y_c,z_c)$ in space and the coordinates of the point $p(u,v)$ in the image.

Studying the robot geometry and kinematics by attaching the coordinate frame to each rigid robot segments or to objects manipulated by the robot where, the camera itself represents a rigid body and a coordinate frame should be assigned to it. A corresponding coordinate frame will describe the pose of the camera. The z_c axis of the camera frame is directed along the optical axis, while the origin of the frame is positioned at the center of projection. Using a right-handed frame where the x_c axis is parallel to the rows of the imaging sensor and the y_c axis is parallel with its columns. The image plane is in the camera, which is placed behind the center of projection. The focal length is the distance (f_c) between the image and the

$$P = \begin{bmatrix} 0 \\ y_c \\ z_c \end{bmatrix}. \qquad\qquad p = \begin{bmatrix} 0 \\ y \end{bmatrix}.$$

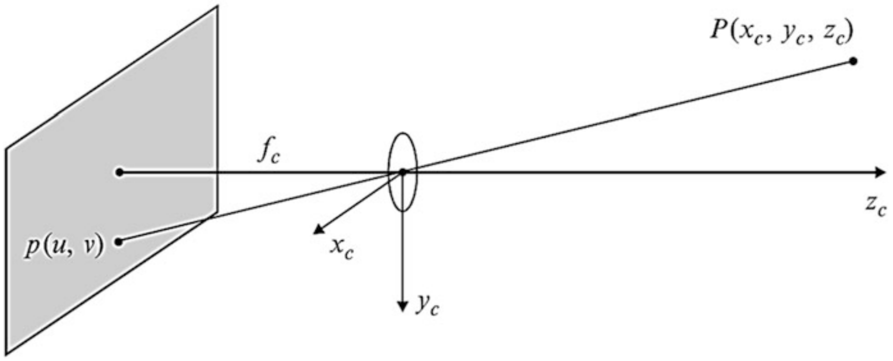

Fig. 12.17 Perspective projection (Bajd et al. 2010)

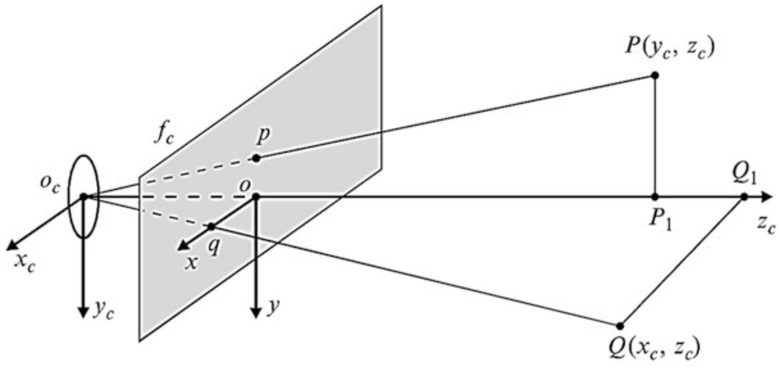

Fig. 12.18 Equivalent image plane (Bajd et al. 2010)

center of projection and has a negative value, as the image plane intercepts the negative z_c axis. The equivalent image plane placed at a positive z_c value (Fig. 12.18). Both the equivalent image plane and the real image plane are symmetrical with respect to the origin of the camera frame. The origin of this frame is placed in the intersection of the optical axis with the image plane. The x and y axes are parallel to the x_c and y_c axes of the camera frame. The camera has two coordinate frames, the camera frame and the image frame. The point P be expressed in the camera frame, while the point p represents its projection onto the image plane. The point P is located in the y_c, z_c plane of the camera frame (Bajd et al. 2010).

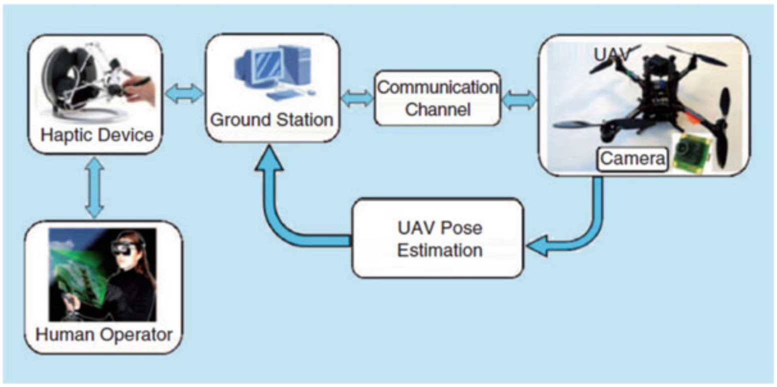

Fig. 12.19 Flying robot system with vision technology (Carloni et al. 2013)

$$P = \begin{bmatrix} 0 \\ y_c \\ z_c \end{bmatrix}, \quad p = \begin{bmatrix} 0 \\ y \end{bmatrix}.$$

A famous case study for using vision technology with robotics is the flying robots such as quadcopters (Carloni et al. 2013). These robots have gained increased interest in research. To navigate safely, flying robots need the ability to localize themselves autonomously using their onboard sensors. Potential applications of such systems include the usage as a flying camera, for example to record sport movies or to inspect bridges from the air, as well as surveillance tasks and applications in agriculture. The main idea of the developed flying robot system is described in Fig. 12.19.

12.6.2 Robot Control Using Visual Servoing Technique

The task is to control the pose of the robot's end-effector using visual information, features, extracted from the image. The Pose is represented by a six element vector encoding position and orientation in 3D space. The camera may be fixed or mounted on the robot's end-effector in which case there exists a constant relationship, between the pose of the camera and the pose of the end-effector. The image of the target is a function of the relative pose between the camera and the target. The relationship between these poses is shown in Fig. 12.17. The distance between the camera and target is frequently referred to as depth or range. The relevant frames required are shown in Fig. 12.20.

The camera contains a lens, which forms a 2D projection of the scene on the image plane where the sensor is located. This projection causes direct depth information to be lost, and each point on the image plane corresponds to a ray in

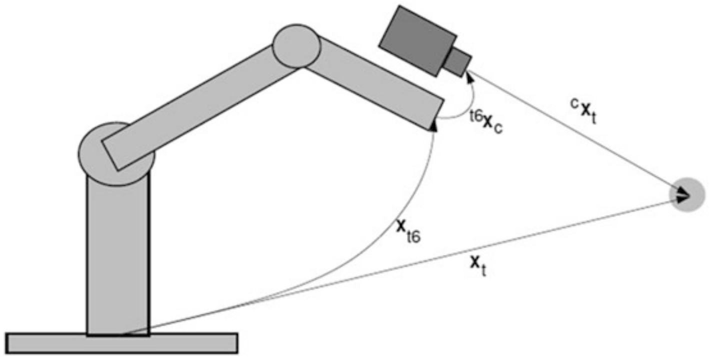

Fig. 12.20 Relevant coordinate frames; world, end-effector, camera, and target

3D space. Some additional information is needed to determine the 3D coordinate corresponding to an image plane point. This information may come from multiple views or knowledge of the geometric relationship between several feature points on the target (Corke 1994).

Robots typically have six degrees of freedom (DOF), allowing the end-effector to achieve, within limits, any pose in 3D space. Visual servoing systems may control six or fewer DOF. Motion so as to keep one point in the scene, the interest point, at the same location in the image plane is referred to as fixation. Animals use fixation to direct the high-resolution fovea of the eye toward regions of interest in the scene. In humans, this low-level, unconscious, fixation motion is controlled by the brain's medulla region using visual feedback from the retina. Fixation may be achieved by controlling the pan/tilt angles of the camera like a human eye or by moving the camera in a plane normal to the optical axis. High performance fixation control is an important component of many active vision strategies. Keeping the target centered in the field of view has a number of advantages that include:

- Eliminating motion blur since the target is not moving with respect to the camera
- Reducing the effect of geometric distortion in the lens by keeping the optical axis pointed at the target

Visual servoing can be classified into position-based visual servoing (Fig. 12.21) and image-based visual servoing. In position-based control (PBVS), the geometric model of the target object is used in conjunction with visual features extracted from the image to estimate the pose with respect to the camera frame, computing the control law by reducing the error in pose space. In this way (Fig. 12.8), the estimation problem involved in computing the object location can be studied separately from the problem of calculate the feedback signal required by the control algorithm (Corke 1994; Miura 2004).

In image-based servoing (IBVS), the last step is omitted, and servoing is done on the basis of image features directly. The structures referred to as dynamic look and move make use of joint feedback, whereas the PBVS and IBVS structures use no joint position information at all. The image-based approach (as shown in

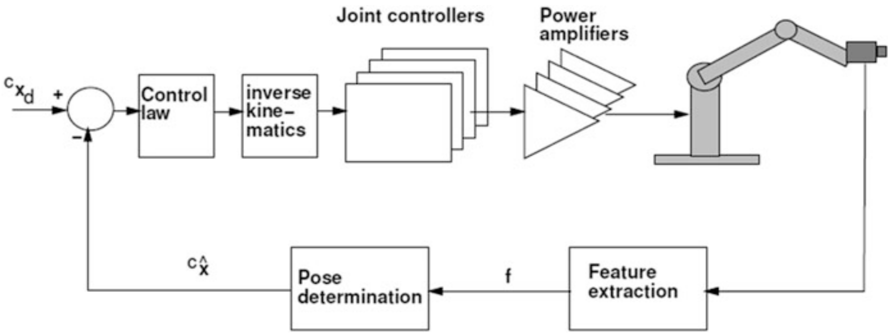

Fig. 12.21 Position-based visual servo (PBVS) structure

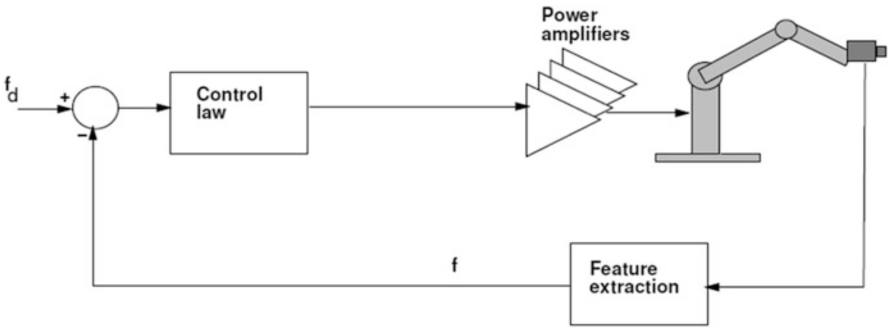

Fig. 12.22 Image-based visual servo (IBVS) structure

Fig. 12.22) may reduce computational delay, eliminate the necessity for image interpretation, and eliminate errors in sensor modelling and camera calibration. However, it does present a significant challenge to controller design since the plant is nonlinear and highly coupled.

12.7 Conclusion

This chapter sheds light on the essentials of a robotic system. First, we exhibited types and applications of robots emphasizing the modelling of the kinematic of a robot manipulator. Then, we formulate the dynamics of a robot manipulator. A third-order polynomial trajectory planning was illustrated as well as linear Segments with Parabolic Blends. Suggestions of some practical control structures have been given; PD position control in different configurations can be realized. Control that based on inverse dynamics is widely used for robots. In order to consider the relation between the joint velocities and the velocities of the robot end point, control based on the (or inverse) Jacobian matrix can be applied. Force control method can find many areas of applications as well. Finally, vision technology for robotic system has been illustrated. The study of robotics and its mathematics has great application in sustainability for the future.

References

Augugliaro F, Schoellig AP, D'Andrea R (2013) Dance of the flying machines. Methods of designing and executing an aerial dance choreography. IEEE Robot Autom Mag 20(4):96–104

Bajd T, Mihelj M, Lenarčič J, Stanovnik A, Munih M (2010) Robotics. Springer, Dordrecht, pp 5–7, 58–60, 77–95

Beni G, Hackwood S (1985) Recent advanced in robotics. Wiley, New York

Bolopion A, Régnier S (2013) A review of Haptic feedback teleoperation systems for micromanipulation and micro-assembly. IEEE Trans Autom Sci Eng 10(3):496–502

Boonvisut P, Cavusoglu MC (2013) Estimation of soft tissue mechanical parameters from robotic manipulation data. IEEE/ASME Trans Mechatron 18(5):1602–1611

Canudas C, Siciliano B, Bastin G (1996) Theory of robot control. Springer, London, pp 21–29

Carloni R, Lippiello V, D'Auria M, Fumagalli M, Mersha AY, Stramigioli S, Siciliano B (2013) Robot vision obstacle-avoidance techniques for unmanned aerial vehicles. IEEE Robot Autom Mag 20(4):22–31

Cicconet M, Bretan M, Weinberg G (2013) Human-robot percussion ensemble, application on the basis of visual cues. IEEE Robot Autom Mag 20(4):105–110

Corke PI (1994) High-performance visual closed-loop robot control. PhD thesis, Mechanical and Manufacturing Engineering, University of Melbourne, pp 186–189

Freeman CT, Rogers E, Huges A-M, Burridge JH, Meadmore KL (2012) Iterative learning control in health care, electrical stimulation and robotic- assisted upper-limb stroke rehabilitation. IEEE Control Syst Mag 32(1):18–43

Hegde GS (2008) A text book on industrial robotics, 2nd edn. University Science Press, New Delhi

Javier J, Prats M, Sanz PJ, Garcia JC, Marin R, Robinson M, Ribas D, Ridao P (2013) Grasping for the seabed. IEEE Robot Autom Mag 20(4):121–130

Jazar RN (2007) Theory of applied robotics: kinematics, dynamics, and control. Springer, New York, pp 507–546

Keung W, Yang B, Liu C, Poignet P (2013) A quasi- spherical triangle- based approach for efficient 3-D soft tissue motion tracking. IEEE/ASME Trans Mechatron 18(5):1472–1484

Lewis FL, Zhou C, Stevens GT, Fitzgeral JJM, Liu K (1999) Robotics. In: Kreith F (ed) Mechanical engineering handbook. CRC Press LLC, Boca Raton

Min Z, Nirwan A, Edwin SH (1992) Mobile manipulator path planning by genetic algorithm. In: Proceedings of the IEEE/RSJ international conference on Intelligent Robot and System, 7–10 July 1992

Miura K (2004) Robot hand positioning and grasping using vision. PhD thesis, Strasbourg I University, Strasbourg

Niku SB (2001) Introduction to robotics: analysis, systems and applications. Prentice Hall, Upper Saddle River

Nonami K, Kendoul F, Suzuki S, Wang W, Nakazawa D (2010) Autonomous flying robots: unmanned aerial vehicles and micro aerial vehicles. Springer, Berlin, pp 2–60

Samad T (2001) Robot control—perspectives in control engineering technologies, applications, and new directions. Wiley-IEEE Press eBook, New York, pp 442–461. doi:10.1109/9780470545485.ch18

Spong MW, Hutchinson S, Vidyasagar M (2006) Robot modeling and control, 1st edn. Wiley, New York, pp 149–185, 215–221

Wang M-L, Wu J-R, Kao L-W, Lin H-Y (2013) Development of a vision system and a strategy simulator for middle size soccer robot. In: International conference on advanced robotics and intelligent systems, May 31–June 2, Tainan, Taiwan, pp 54–58

Yamamoto Y (1992) Coordination locomotion and manipulation of mobile robot manipulator. IEEE Trans Autom Control 39(6):1326–1332

Zhou H, Alici G, Than TD, Li W (2013) Modeling and experimental investigation of rotational resistance of spiral-type robotic capsule inside a real intestine. IEEE/ASME Trans Mechatron 18(5):1555–1562