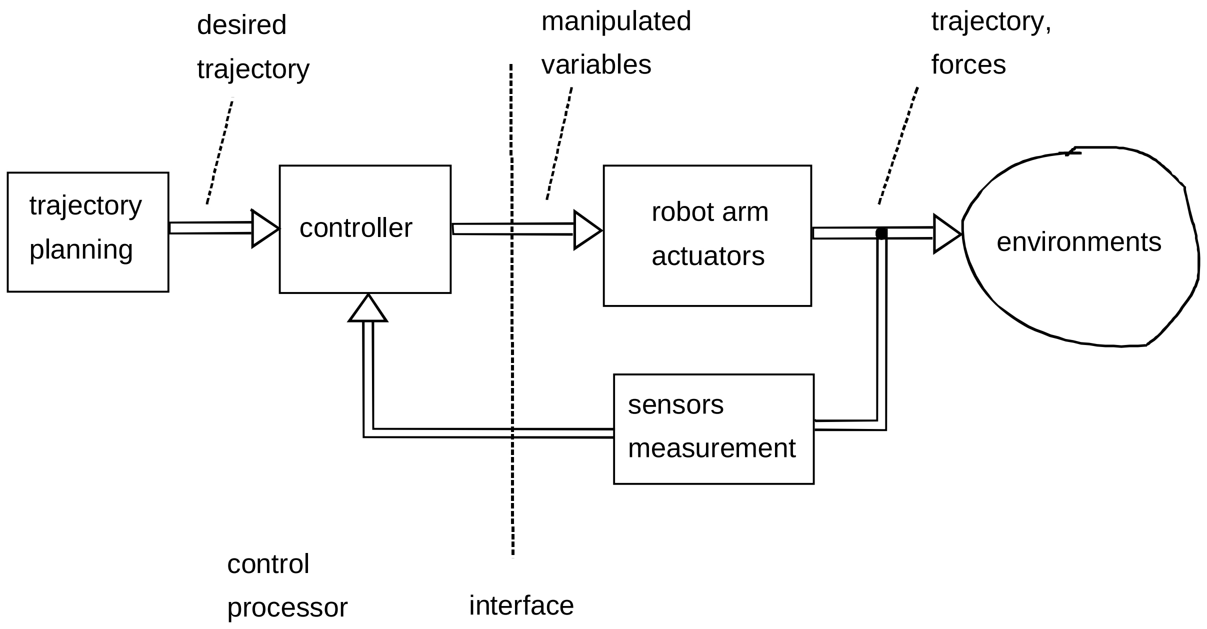
Dr. W. Weber

March 2014

Last modification: 3th of April 2014

Model-based Nonlinear Robot Arm Control



Literature:

- **Siciliano**, B., **Sciavicco**, L., **Villani**, L., **Oriolo**, G.: Robotics: Modelling, Planning and Control. 2nd ed., Springer, London, 2010
- Craig, J.J.: *Introduction to Robotics*. Addison-Wesley, 3rd edition 2004
- Weber, W.: *Industrieroboter – Methoden der Steuerung und Regelung*. Fachbuchverlag Leipzig im Carl Hanser Verlag, München, 2nd ed., 2009 (in German)
- Spong, M.W.; Vidyasagar, M.: *Robot Modelling and Control*. John-Wiley, 2006
- Kozlowski, K.: *Modelling and Identification in Robotics*. Springer, London[u.a.], 1998
- Corke, P.: *Robotics, Vision and Control*. Springer, Berlin/Heidelberg, 2011

Content

1. Introduction to robot arm control

1.1 Definition of an industrial robot and overall robot control system

We begin with some robot terminology. Exactly what constitutes a robot is sometimes debated. Numerically controlled (milling) machines are usually not referred as robots. The distinction lies somewhere in the sophistication of the programmability of the device – if a mechanical device has several joints and can be programmed to perform a wide variety of applications, it is probably an industrial robot. Machines, which are relegated to one task, are considered fixed automation. Industrial robots consist of links that are connected with **joints,** which allow relative motion of neighbouring **links**. These joints are usually instrumented with position sensors that allow the relative position of neighbouring links to be measured. In case of **rotary** or revolute joints, these displacements are called joint angles. Some robots contain sliding or **prismatic** joints (see Fig. 1.1). In Fig. 1.2 two typical industrial robots are depicted.

We will use the following short definition:

Industrial robots are mechanical devices with several links and joints. They are used for the transport of material, tools or special equipment. The motion is user-programmable.

At the free end of the chain of links, which make up the robot, is the **end effector**. Depending on the intended application of the robot, the end effector may be a gripper, welding torch or any other device. A certain point of the end effector is the **TCP** (Tool centre point). We can describe the position of the TCP by a vector $\boldsymbol{p}_0$ and the orientation by the rotation of a tool coordinate system (x_T, y_T, z_T), which is attached to the end effector according to the non-moving base coordinate system K_0 (Fig. 1.3).

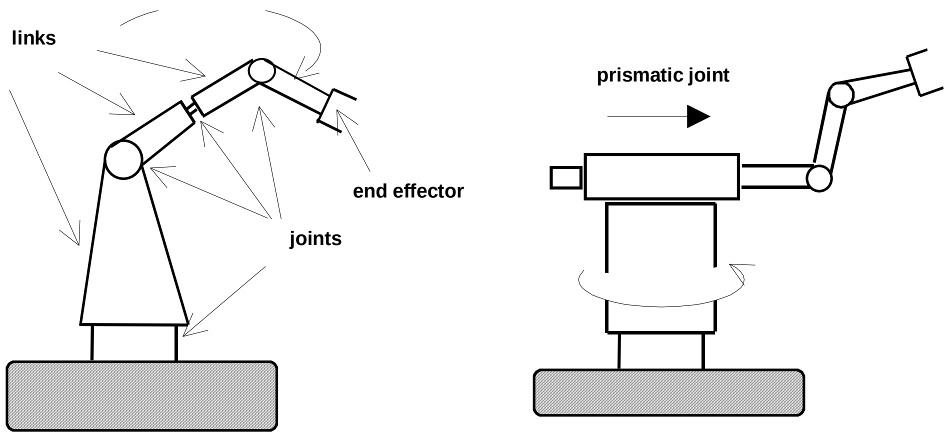


Figure 1.1 Links and joints of an industrial robot

A robot system consists of three main components: electro-mechanical system (actuators, robot arm), programming system, robot control, feedback control and measuring and sensor system. Fig. 1.4 tries to sketch the cooperation between these parts.

- **Programming system**:

A robot user needs to „teach“ the robot the specific task that is to be performed. This can be done with the so-called **Teach In** method. The programmer uses the *teach box* of the robot control and drives the robot to the desired positions and stores them and other values like travel speed, corner smoothing parameters, process parameters, etc. For the programming period which can take a significant time for complex tasks the production is idle. Another possibility is the use of Off-line programming systems.

Figure 1.2 Typical industrial robots (Reis RV6, Stäubli SCARA RS80)

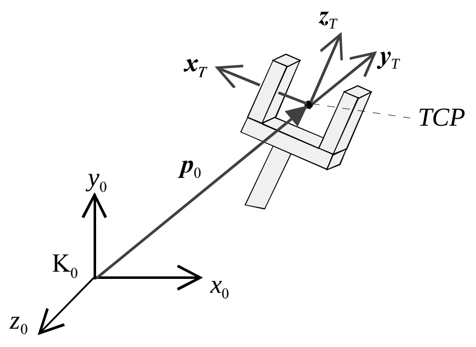


Figure 1.3 Description of position and orientation of the end effector

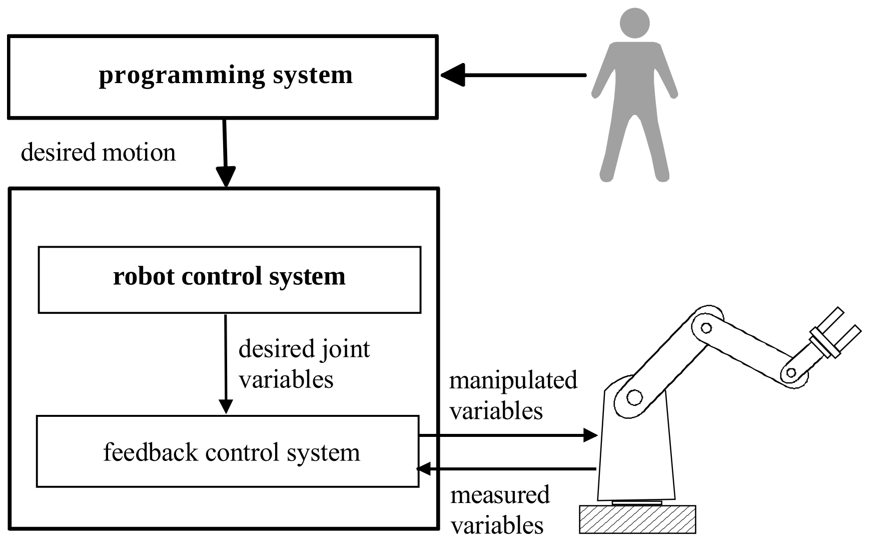


Figure 1.4 Cooperation of components of a robot system

- **Robot control**:

The robot control interprets the robot's application program and generates a series of time dependent joint values and joint velocities (and sometimes accelerations) in an appropriate way as desired values for the feedback control.

- **Feedback control system and actuators**:

The feedback control system causes the motors of each axis to generate a desired torque, which is transformed by the gear train to appropriate torques at the joints. Nowadays brushless DC-motors and special gear trains like Harmonic Drives are used to drive the axis.

- **Measuring and Sensor system**

For feedback control the joint coordinates and joint velocities must be measured. For example resolvers, optical encoders, and so on are used. For advanced applications the robot should “see” and “feel” its environment. Range finder like ultrasonic-sensor laser-radar range finders may be used to detect objects. Image processing is used for recognition of objects, collision avoidance and others. In assembly torque/force sensors detect forces and torques that act on the robot.

1.2 Closed loop control problem of robot arm

In future we will be concerned with the feedback control system or closed loop control system and we will use the following definition for the task closed loop control:

Find appropriate torques / forces to servo all the joints of the manipulator to track a desired position trajectory as closely as possible.

In simple terms, the control problem is to find the necessary manipulated variables to cause the actuators to variate the torques in an appropriate manner. Figure 1.5 shows a high level block diagram of the closed loop control system. The controller reads the vector of joint positions, joint velocities and eventual forces which act upon the environment and in any case the desired trajectory. On the base of these variables the controller calculates manipulated variables to cause the actuators to apply torques on the links.

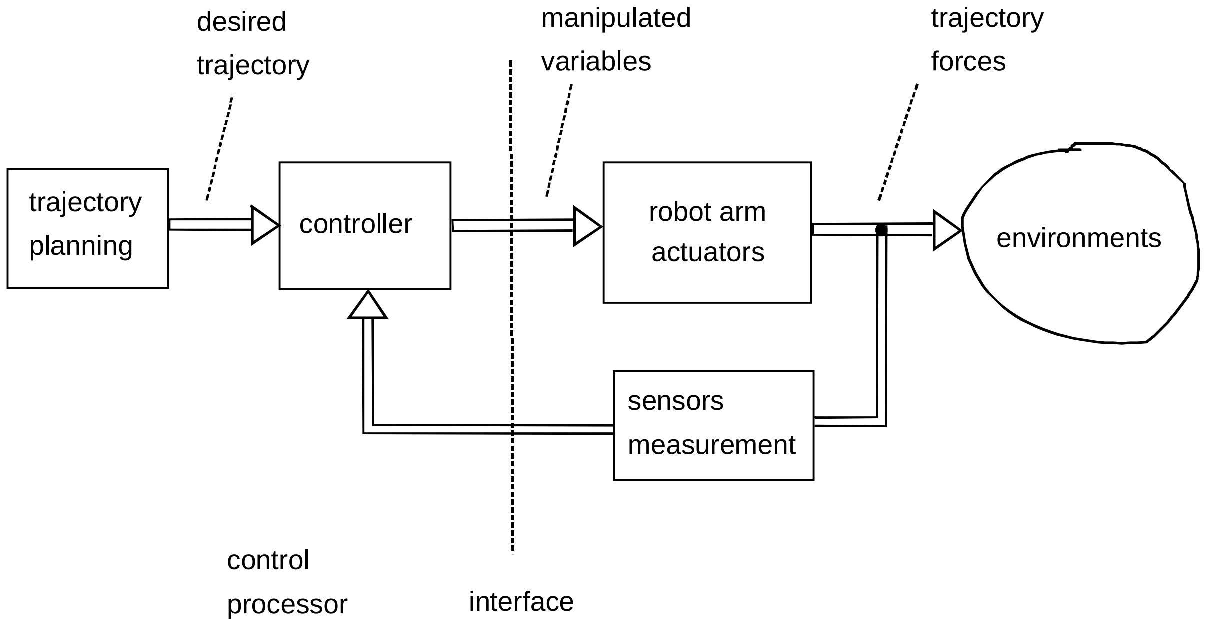


Figure 1.5 High-level block diagram of a robot control system

1.3 Attribute of robot arm control problem

First it is obvious to use a single axis control system to track each joint of the robot arm to the instant desired angle or desired position (Fig. 1.6). The error variable is calculated from the instant desired position $q_{d,i}$ of the *i*-th joint and the measured variable q_i. The single axis controller causes the servo motor to take the (effective) current $I_{A,i}$, which is proportional to the voltage $U_{s,i}$ and stands for a torque or force τ_i.

While the control problem can be stated in a simple manner, the solution is quite complicated because of coupling terms among the link movements by inertial forces, gravity loading and Coriolis and Centrifugal forces. We will roughly discuss such coupling on the base of Figure 1.7.

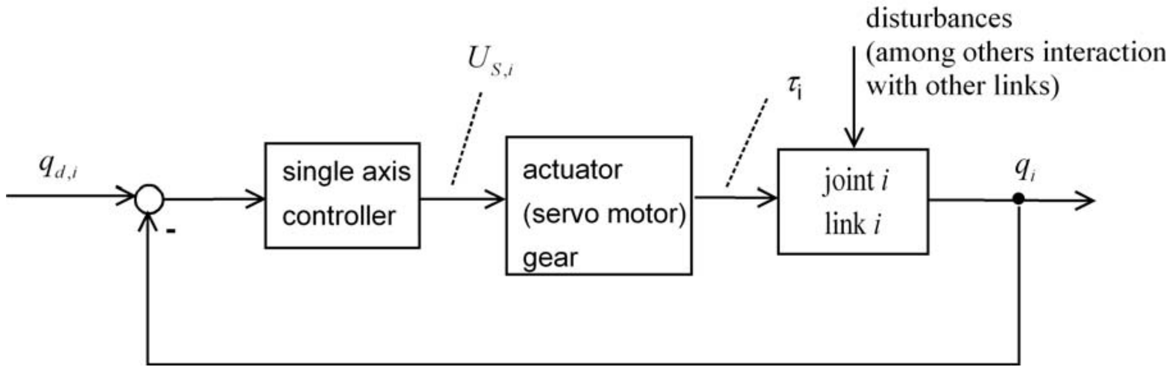


Figure 1.6. Single axis closed loop control.

- The motion $q_1(t)$, which are caused by $\tau_1(t)$, depends on $q_2(t)$ and $q_3(t)$, because the actual inertia moment acting on joint 1, depends on $q_2(t)$ and $q_3(t)$.
- The gravity load causes a torque at joint 2, which depends on the actual position of the joints 2 and 3. It holds: $\tau_2 = \ldots + \cos(q_2+q_3)\cdot k_2 + \cos q_2 \cdot k_3 + \ldots$
- If joint 1 moves, then Centripetal torques act on joints 2 and 3, which depend on the angular velocity of joint 1 and the angles of joint 2 and 3. It holds: $\tau_2 = \cdots + c_1 \cdot \sin q_2 \cdot \cos(q_2+q_3)\cdot \dot{q}_1^2 + \cdots$

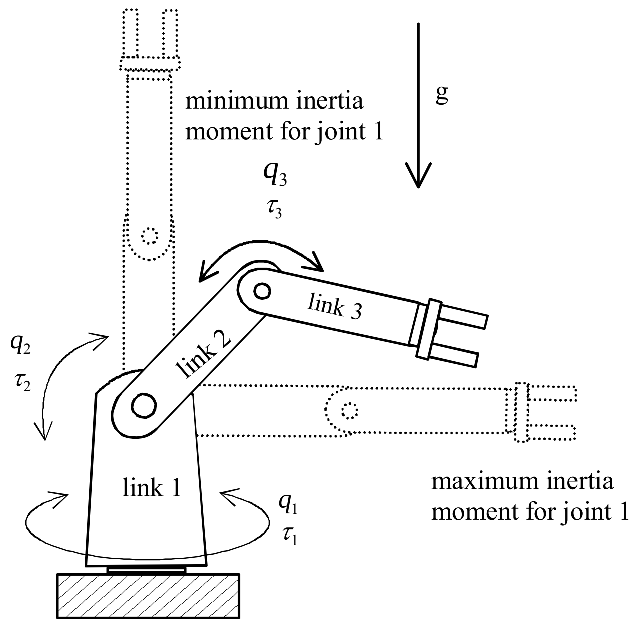

minimum inertia moment for joint 1
g
q_3
τ_3
link 3
link 2
q_2
τ_2
link 1
q_1
τ_1
maximum inertia moment for joint 1

Figure 1.7 Joint angles and torques of a robot with three joints

Therefore the controlled system robot arm is a multi –input /multi-output system.

For this reason a single axis control approach must essentially be viewed as approximate method. Moreover this holds for the class of linear control systems, because in the upper case, τ_2 is a nonlinear function of q_2, but the use of linear control techniques is only valid when the controlled system can be mathematically modelled by linear differential equations. However it is often reasonable to make such approximations, and till now it is the case that these linear methods are often used in industrial practice. But the control performance is limited and with the cost reduction of powerful controllers more and more nonlinear control algorithms are used to improve the control performance. This lecture covers a certain class of nonlinear controllers based on the dynamic model of the robot system.

2. Necessary Basics of Kinematics

Here the Basics of kinematics which are necessary for robot dynamics are given in a rough manner. Test your knowledge by solving exercise 1. For a deeper insight see literature. Please notice: Vector and matrices are written in bold letters. In lecture vectors and matrices are written by underlined letters.

2.1 Cartesian coordinate systems

A Cartesian coordinate system is fully described by its base vectors $\boldsymbol{x}_i, \boldsymbol{x}_i, \boldsymbol{z}_i$ which are orthogonal to each other (Fig. 2.1). These vectors point along the coordinate axes.

$$\boldsymbol{x}_i = \begin{pmatrix} 1 \\ 0 \\ 0 \end{pmatrix},\ \boldsymbol{y}_i = \begin{pmatrix} 0 \\ 1 \\ 0 \end{pmatrix},\ \boldsymbol{z}_i = \begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix},\quad |\boldsymbol{x}_i| = |\boldsymbol{y}_i| = |\boldsymbol{z}_i| = 1 \quad \text{(unit-vectors)} \tag{2.1}$$

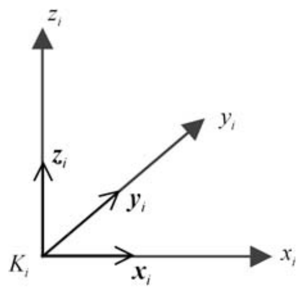


Figure 2.1 Base vectors of a coordinate system

2.2 Free vectors

Geometrically a vector can be interpreted as an arrow describing its length and direction (see vector $\boldsymbol{u}$ in Fig. 2.2). A (free) vector can be parallel shifted without change (two vectors are identical, if they are equal in length and direction).

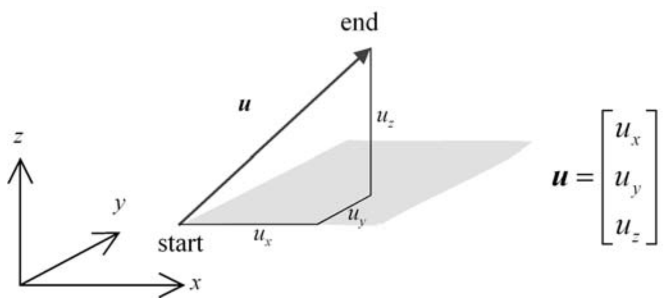

end
$\boldsymbol{u}$
u_z
u_y
u_x
start
z
y
x
$\boldsymbol{u} = \begin{bmatrix} u_x \\ u_y \\ u_z \end{bmatrix}$

Figure 2.2 Geometrical description of a vector

If we have established a coordinate system, we can describe vectors related to this coordinate system. In the context of a coordinate system a (three-dimensional) vector is a three-tupel of numerical values indicating the distances along the axes of the coordinate system. Each of these distances along an axis can be thought of as a result of projecting the vector onto the corresponding axis (Fig. 2.2).

2.3 Position vectors

Once a coordinate system is established we can locate any point *P* in the world with 3x1 position vector. A position vector has its starting point at the origin of a coordinate system and its end point at *P*.(s. Fig. 2.3). Therefore a position vector can not be shifted parallel without change.

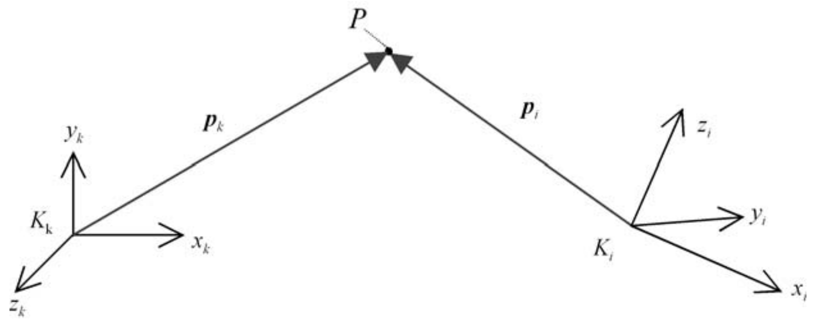


Figure 2.3 Description of position of the end effector by a position vector

2.4 Rotation Matrix and Homogenous Matrix (Frame)

Rotation matrices are used to describe the base vectors of a coordinate system relative to another reference system. A vector ***a*** can be expressed with such a rotation matrix in different coordinate systems.

$\boldsymbol{a}^{(i)}$: vector ***a*** represented in coordinate system K_i.

The Rotation Matrix ${}^k_i\boldsymbol{A}$ describes the base vectors of coordinate systems *k* in coordinate system K_i:

$$ {}^k_i\boldsymbol{A} = \begin{pmatrix} \boldsymbol{x}_k^{(i)} & \boldsymbol{y}_k^{(i)} & \boldsymbol{z}_k^{(i)} \end{pmatrix} \quad . \tag{2.2a}$$

Hence ${}^i_k\boldsymbol{A}$ is the description of frame *i* relative to *k*:

$$ {}^i_k\boldsymbol{A} = \begin{pmatrix} \boldsymbol{x}_i^{(k)} & \boldsymbol{y}_i^{(k)} & \boldsymbol{z}_i^{(k)} \end{pmatrix} \tag{2.2b}$$

An arbitrary vector ***a*** represented in the coordinate system *k* can be expressed in the coordinate system *i* using the rotation matrix (and vice versa):

$$\boldsymbol{a}^{(i)} = {}^k_i\boldsymbol{A} \cdot \boldsymbol{a}^{(k)} \text{ respectively } \boldsymbol{a}^{(k)} = {}^i_k\boldsymbol{A} \cdot \boldsymbol{a}^{(i)} \tag{2.3}$$

The inverse of a rotation matrix is equal to its transpose. It holds:

$$\left[{}^k_i\boldsymbol{A}\right]^{-1} = {}^k_i\boldsymbol{A}^{\mathrm{T}} = {}^i_k\boldsymbol{A} \tag{2.4}$$

More Properties of rotation matrices:
- det(*A*) = 1
- columns are mutually orthogonal

With a 4x4-matrix the relationship between two coordinate systems can be described completely. This matrix is named Homogeneous Matrix (frame). Assume two general coordinate systems K_i and K_k (Fig. 2.7). In the homogenous matrix ${}^k_i\boldsymbol{T}$ the base vectors of K_k are described in K_i and the position vector $\boldsymbol{p}_{ik}$ from origin of K_i to origin of K_k is described in K_i (see Fig. 2.4):

$$ {}^k_i\boldsymbol{T} = \begin{pmatrix} \boldsymbol{x}_k^{(i)} & \boldsymbol{y}_k^{(i)} & \boldsymbol{z}_k^{(i)} & \boldsymbol{p}_{ik}^{(i)} \\ 0 & 0 & 0 & 1 \end{pmatrix} \qquad (2.5 $$

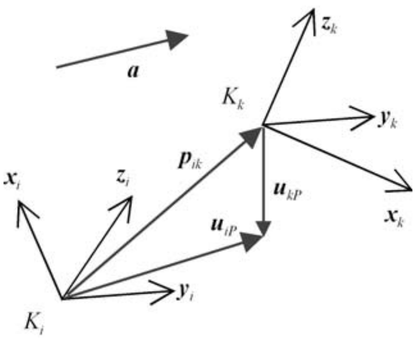


Figure 2.4 Spatial relationship and mapping of vectors between two coordinate systems

With the forth row $(0,0,0,1)$ we can treat free vectors and position vectors in a consistent way.

a) Free vectors

Here the free vector is described by a column vector with 4 components, but the 4th component is always 0. For a general free vector $\boldsymbol{a}$ it holds (see also Fig. 2.4):

$$ \boldsymbol{a}^{(k)} = \begin{pmatrix} a_x \\ a_y \\ a_z \\ 0 \end{pmatrix}^{(k)} \quad \boldsymbol{a}^{(i)} = \begin{pmatrix} a_x \\ a_y \\ a_z \\ 0 \end{pmatrix}^{(i)} = {}^k_i\boldsymbol{T} \cdot \boldsymbol{a}^{(k)} \qquad (2.6) $$

Of course you can also use equation (2.3) for this mapping, if you define $\boldsymbol{a}$ as a (3,1)-vector.

b) Position vectors

Here a position vector is described by a column vector with 4 components, but the 4th component is always 1. If we know the position vector $\boldsymbol{u}_P^{(i)}$ from the origin of K_i to a certain point P, we can calculate the position vector from origin of K_k to this point P (see also Fig. 2.4):

$$ \boldsymbol{u}_{iP}^{(i)} = {}^k_i\boldsymbol{T} \cdot \boldsymbol{u}_{kP}^{(k)} \text{ with } \boldsymbol{u}_{iP}^{(k)} = \begin{pmatrix} u_{kPx} \\ u_{kPy} \\ u_{kPz} \\ 1 \end{pmatrix}^{(k)} \text{ and } \boldsymbol{u}_{kP}^{(i)} = \begin{pmatrix} u_{iPx} \\ u_{iPy} \\ u_{iPz} \\ 1 \end{pmatrix}^{(i)} \qquad (2.7) $$

Notice: In general the both position vectors are not equivalent in contrast to the free vectors in (2.6). Is $\boldsymbol{u}_P^{(i)}$ given and we have to describe $\boldsymbol{u}_P^{(i)}$ and vice versa it holds:

$$ \boldsymbol{u}_{iP}^{(i)} = {}^k_i\boldsymbol{T} \cdot \boldsymbol{u}_{kP}^{(k)},\, \boldsymbol{u}_{kP}^{(k)} = {}^i_k\boldsymbol{T} \cdot \boldsymbol{u}_{iP}^{(i)},\quad {}^i_k\boldsymbol{T} = ({}^k_i\boldsymbol{T})^{-1} = \begin{pmatrix} {}^k_i\boldsymbol{A}^\mathrm{T} & -{}^k_i\boldsymbol{A}^\mathrm{T}\boldsymbol{p}_{ik}^{(i)} \\ \boldsymbol{0} & 1 \end{pmatrix} \qquad (2.8) $$

Several coordinate systems (compound transformations)

Combining two transformations is performed by multiplication of two homogenous transforms. An example is depicted at Fig. 2.5. It holds:

$$\begin{aligned}&{}_1^3\boldsymbol{T} = {}_1^2\boldsymbol{T}\cdot{}_2^3\boldsymbol{T} = \begin{pmatrix} \boldsymbol{x}_2^{(1)} & \boldsymbol{y}_2^{(1)} & \boldsymbol{z}_2^{(1)} & \boldsymbol{p}_{1,2}^{(1)} \\ 0 & 0 & 0 & 1 \end{pmatrix}\cdot\begin{pmatrix} \boldsymbol{x}_3^{(2)} & \boldsymbol{y}_3^{(2)} & \boldsymbol{z}_3^{(2)} & \boldsymbol{p}_{2,3}^{(2)} \\ 0 & 0 & 0 & 1 \end{pmatrix} = \\ &\begin{pmatrix} \boldsymbol{x}_3^{(1)} & \boldsymbol{y}_3^{(1)} & \boldsymbol{z}_3^{(1)} & \boldsymbol{p}_{1,3}^{(1)} \\ 0 & 0 & 0 & 1 \end{pmatrix}\end{aligned} \tag{2.9}$$

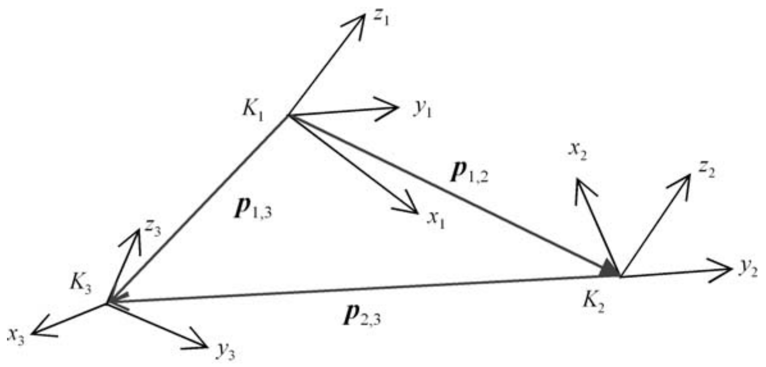


Figure 2.5 Compound transformations

Commonly it holds:

$$\begin{aligned}&{}_i^k\boldsymbol{A} = {}_i^{i+1}\boldsymbol{A}\cdot{}_{i+1}^{i+2}\boldsymbol{A}\ldots{}_{k-2}^{k-1}\boldsymbol{A}\cdot{}_{k-1}^{k}\boldsymbol{A} \\ &{}_k^i\boldsymbol{A} = \left[{}_i^k\boldsymbol{A}\right]^{\mathrm{T}} = {}_{k-1}^{k}\boldsymbol{A}^{\mathrm{T}}\cdot{}_{k-2}^{k-1}\boldsymbol{A}^{\mathrm{T}}\ldots{}_{i+1}^{i+2}\boldsymbol{A}^{\mathrm{T}}\cdot{}_i^{i+1}\boldsymbol{A}^{\mathrm{T}} = {}_k^{k-1}\boldsymbol{A}\cdot{}_{k-1}^{k-2}\boldsymbol{A}\ldots{}_{i+2}^{i+1}\boldsymbol{A}\cdot{}_{i+1}^{i}\boldsymbol{A}\end{aligned} \tag{2.10a}$$

$$\begin{aligned}&{}_i^k\boldsymbol{T} = {}_i^{i+1}\boldsymbol{T}\cdot{}_{i+1}^{i+2}\boldsymbol{T}\ldots{}_{k-2}^{k-1}\boldsymbol{T}\cdot{}_{k-1}^{k}\boldsymbol{T} \\ &{}_k^i\boldsymbol{T} = \left[{}_i^k\boldsymbol{T}\right]^{-1} = {}_{k-1}^{k}\boldsymbol{T}^{-1}\cdot{}_{k-2}^{k-1}\boldsymbol{T}^{-1}\ldots{}_{i+1}^{i+2}\boldsymbol{T}^{-1}\cdot{}_i^{i+1}\boldsymbol{T}^{-1} = {}_k^{k-1}\boldsymbol{T}\cdot{}_{k-1}^{k-2}\boldsymbol{T}\ldots{}_{i+2}^{i+1}\boldsymbol{T}\cdot{}_{i+1}^{i}\boldsymbol{T}\end{aligned} \tag{2.10b}$$

2.5 Orientation defined by Euler-Angles

The orientation of a coordinate system K_W in relation to a reference system K_R can be described by a rotation matrix:

$${}_R^W\boldsymbol{A} = \begin{pmatrix} \boldsymbol{x}_W^{(R)} & \boldsymbol{y}_W^{(R)} & \boldsymbol{z}_W^{(R)} \end{pmatrix} \quad . \tag{2.11}$$

But there are some redundancies. To describe this orientation 3 specifications (3 angles) are enough. Another method to describe the orientation are the Euler-angles.
There are several definitions of Euler-Angles. Here we use the so called Z-Y-X Euler-Angles.

Euler-angles: rotation takes place around a defined axis of the reference frame or the yet moved frame. Here we use one of several definitions.

Definition (Z-Y-X Euler-Angles)

The reference frame is rotated with tree angles *A*, *B*, *C* that the resulting coordinate system has the same orientation as the target coordinate system K_W.

- First the reference system will be rotated **around the z_R-axis with angle *A*.** The *x*'-axis of this new coordinate system *K*' should be perpendicular if possible to the z_W-axis.
- Second we will rotate **about *y*'-axis with angle *B*.** For the new coordinate system *K''* it should be hold: $x''=x_W$. B has the negative direction of *y*'-axis.
- Last we will rotate **about $x''=x_W$ with angle *C*** to get the orientation of K_W (s. Fig. 2.6).

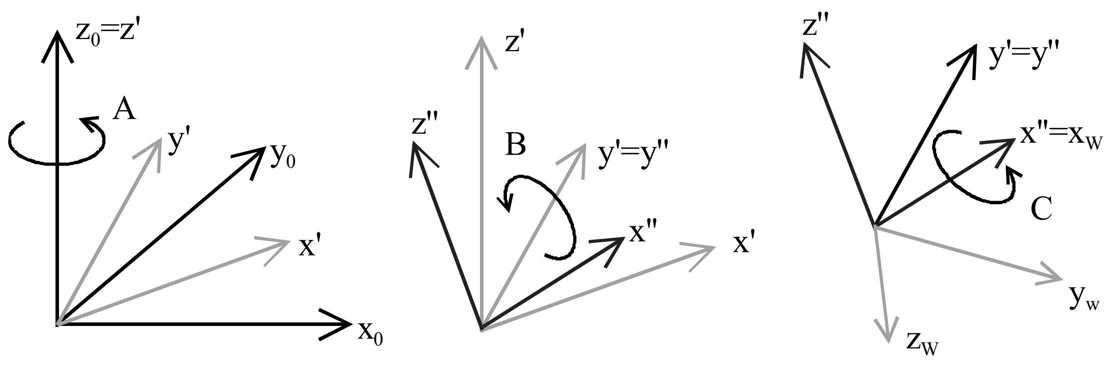


Fig. 2.6. Euler-Angles to definite the orientation (definition B: Z-Y-X-Euler-angles)

It is possible to describe the three rotations by three rotation matrices:

$$ {}_0^{'}\boldsymbol{A} = \begin{pmatrix} \cos A & -\sin A & 0 \\ \sin A & \cos A & 0 \\ 0 & 0 & 1 \end{pmatrix},\ {}_{'}^{''}\boldsymbol{A} = \begin{pmatrix} \cos B & 0 & -\sin B \\ 0 & 1 & 0 \\ \sin B & 0 & \cos B \end{pmatrix},\ {}_{''}^{w}\boldsymbol{A} = \begin{pmatrix} \cos B & 0 & -\sin B \\ 0 & 1 & 0 \\ \sin B & 0 & \cos B \end{pmatrix} $$

The resulting rotation matrix between K_W and the reference coordinate system K_R is :

$$ {}^{*}\boldsymbol{A} = {}^{'}\boldsymbol{A} \cdot {}_{'}^{''}\boldsymbol{A} \cdot {}_{''}^{*}\boldsymbol{A} = \begin{pmatrix} C_A \cdot C_B & -S_A \cdot C_C - C_A \cdot S_B \cdot S_C & S_A \cdot S_C - C_A \cdot S_B \cdot C_C \\ S_A \cdot C_B & C_A \cdot C_C - S_A \cdot S_B \cdot S_C & -C_A \cdot S_C - S_A \cdot S_B \cdot C_C \\ S_B & C_B \cdot S_C & C_B \cdot C_C \end{pmatrix} \tag{2.12} $$

with $C_A = \cos A$, $S_A = \sin A$ and so on. From there between $\boldsymbol{l}=\boldsymbol{x}_w, \boldsymbol{m}=\boldsymbol{y}_w, \boldsymbol{n}=\boldsymbol{z}_w$ and the angles *A,B,C* the following relations are valid:

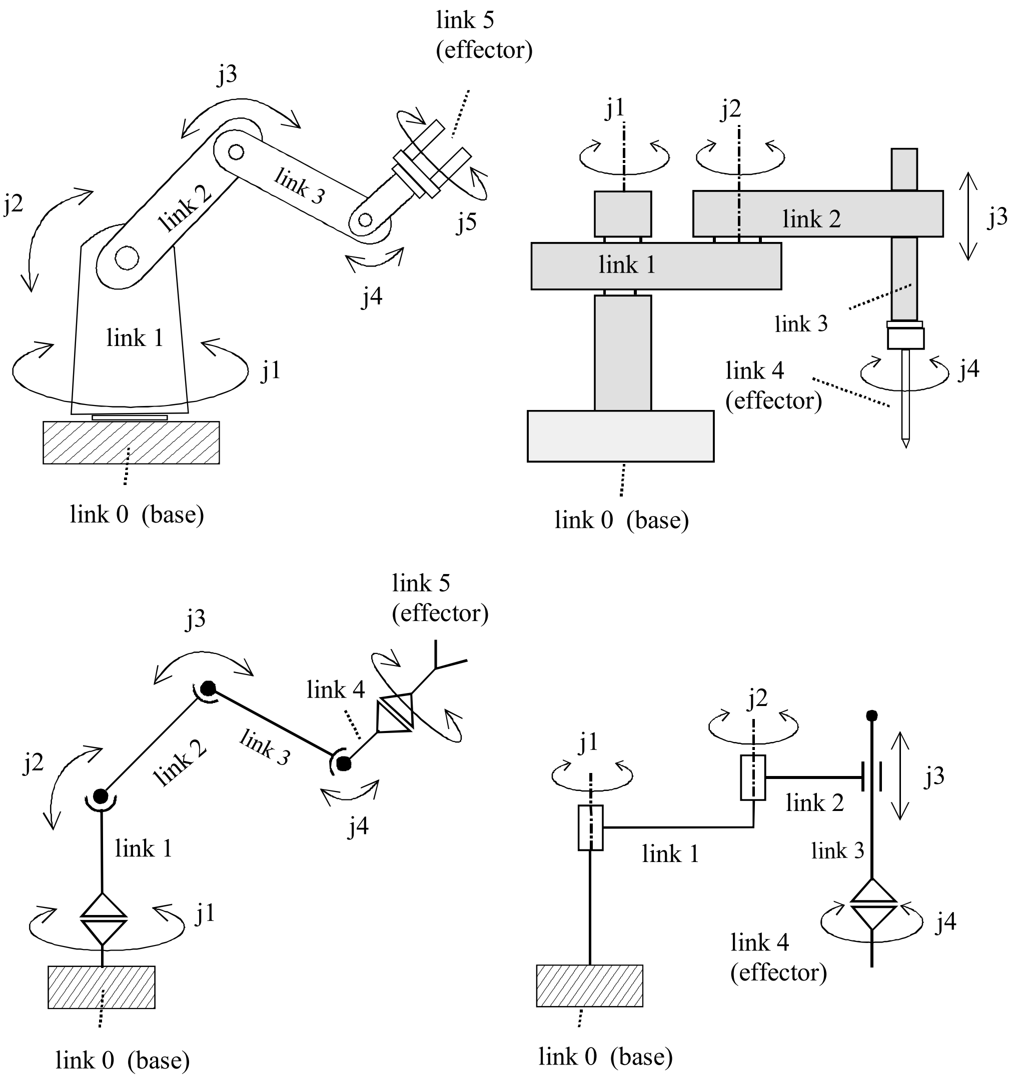
$$ \boldsymbol{l} = \begin{pmatrix} C_A \cdot C_B \\ S_A \cdot C_B \\ S_B \end{pmatrix},\ \boldsymbol{m} = \begin{pmatrix} -S_A \cdot C_C - C_A \cdot S_B \cdot S_C \\ C_A \cdot C_C - S_A \cdot S_B \cdot S_C \\ C_B \cdot S_C \end{pmatrix},\ \boldsymbol{n} = \begin{pmatrix} S_A \cdot S_C - C_A \cdot S_B \cdot C_C \\ -C_A \cdot S_C - S_A \cdot S_B \cdot C_C \\ C_B \cdot C_C \end{pmatrix} \tag{2.13} $$

$$ A=\mathrm{arctan2}(\ell_y, \ell_x),\ B=\arcsin(\ell_z),\ C=\mathrm{arctan2}(m_z, n_z) \tag{2.14} $$

3 Robot kinematics

3.1 Structure of robot arms

We consider only robots with open kinematic chains. Here a link follows a joint. One link has only connection with a predecessor and a successor joint via one joint, respectively. Examples see Fig. 3.1 at the top.



a) robot with 5 revolute joints b) SCARA robot (4 loints)

Figure 3.1 Two possible designs of robots

For sketching the kinematical structure of a robot arm we use very simple signs for joint, links, base of the robot and effector (Fig. 3.2). Examples you can see in Fig. 3.1 at the bottom. We assume that all joints have only one axis and each joint is followed by the corresponding link. If there are in reality joints with more than one axis, there are assumed fictitious links with length 0 and mass 0 between the several joints axes. Fig. 3.3 shows an example.

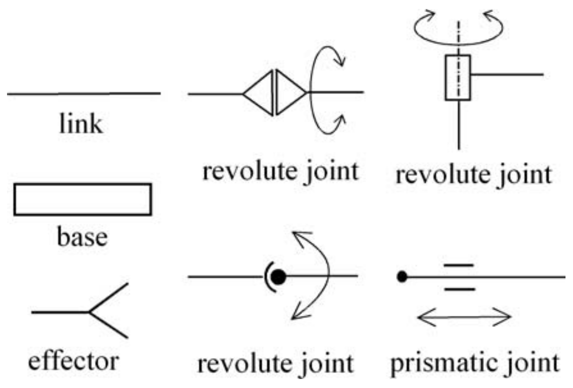


Figure 3.2 Signs for simple sketch of the kinematical structure of a robot arm

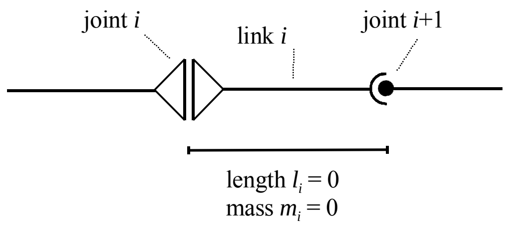


Figure 3.3 Joints with more than one axis: several joints with one axis are assumed

3.2 Fixing the coordinate systems

a) general

At every link $\boldsymbol{i}$ (i=0,1, ... , n) of a robot with n joints a coordinate system K_i must be fixed. There is no relative motion between link i and coordinate system K_i .

b) Defining of non moving base system $\boldsymbol{K_0}$

K_0 is the base-frame or world-frame. It is fixed at the non moving base (link 0).

The origin of K_0 lies somewhere at the first joint axis. In most cases it is useful to fix the origin of K_0 near by joint 1 (see examples).

The z_0-axis with unit vector $\boldsymbol{z}_0$ is directed parallel to the axis of the first joint. The x_0- and y_0-axis can then be chosen free under the condition that the vectors $\boldsymbol{x}_0$, $\boldsymbol{y}_0$, $\boldsymbol{z}_0$ define a right handed coordinate system.

c) Defining the coordinate systems $\boldsymbol{K_i}$ ($\boldsymbol{i}$=1,2,..., $\boldsymbol{n}$-1)

- It is valid in generality: The origin of K_i must be fixed on the axis of joint i+1

- If joint axes i and i+1 intersect than the origin of K_i lies on the intersecting point.

- Are the joint axes i and i+1 parallel, than define first the origin of K_{i+1} using these and following rules. Is the origin of K_{i+1} fixed, the origin of K_i is that one point on the axis of joint i+1, which minimizes the distance between the origins of K_i and K_{i+1}.

- Do the joint axes *i* and *i*+1 not intersect and are they not parallel than the following is valid: We look for the common normal vector of axis *i* and axis *i*+1. The origin is then fixed on the intersecting point of this normal vector with the axis of joint *i*+1.

The z_i-axis is parallel to the joint axis *i*+1 (There are two possibilities for the direction, choose one!).

- Defining of the x_i-axis
  - Intersect z_{i-1}-axis and z_i-axis then the x_i-axis is parallel to the direction of the vector $\boldsymbol{z}_{i-1} \times \boldsymbol{z}_i$ (There are two possibilities for the direction, choose one !).
  - Do the z_{i-1}-axis and the z_i-axis not intersect and the z_{i-1}-axis and the z_i-axis are not aligned (collinear), than the x_i-axis is parallel to the common normal vector of the joint axes *i* and *i*+1. The x_i-axis is directed from joint axis *i* to joint axis *i*+1.
  - Are the z_{i-1}-axis and the z_i-axis aligned (collinear), than the x_i-axis is any normal vector to the z_i-axis.

d) Defining K_n

In generality it is valid for K_n: K_n is defined in such way, that the position and orientation of K_n relatively to K_{n-1} can be defined with the four Denavit-Hartenberg parameters (see next pages). If possible, the origin of K_n should be attached to the TCP.
One possibility to proceed is the following: The z_n-axis has the direction of the z_{n-1}-axis. x_n is perpendicular to z_{n-1} and the direction of x_n is from z_{n-1}-axis to z_n-axis. Are z_{n-1}-axis and z_n-axis aligned (collinear), than the direction of x_n has the direction of x_{n-1}.

3.3 The Denavit-Hartenberg parameters and matrices

3.3.1 Denavit-Hartenberg parameters

With four parameters (2 translational parameters, 2 revolute parameters) the relative position and orientation of two neighbouring coordinate systems can be defined. Or in other words: With two sliding motions d_i, a_i and two revolute motions $\boldsymbol{\theta_i}$, α_i we can shift and rotate K_{i-1} to K_i. Precondition is that both coordinate systems are valid in sense of the Denavit-Hartenberg conventions. The four parameters are:

- The **absolute angle $\boldsymbol{\theta_i}$** around joint *i*, this means around $\underline{z}_{i-1}$, to get $\boldsymbol{x}_{i-1}$ in direction of $\boldsymbol{x}_i$.
- **The translational offset $\boldsymbol{d_i}$** from the origin of K_{i-1} in the direction of $\underline{z}_{i-1}$, to minimize the distance between the origin of K_{i-1} and K_i. (notice: the sign is caused by $\boldsymbol{z}_{i-1}$),
- **The translational offset $\boldsymbol{a_i}$** in direction of $\boldsymbol{x}_i$, to minimize the distance between the origines of
  K_{i-1} and K_i (a_i is always positiv).
- The angle α_i rotated about the axis $\boldsymbol{x}_i$, to get $\boldsymbol{z}_{i-1}$ in direction of $\boldsymbol{z}_i$.

3.3.2 Denavit-Hartenberg matrices

The transformation from one link frame to the next can now be defined by homogeneous transformations using this notation:

$$ {}^{i}_{i-1}\boldsymbol{T} = \begin{bmatrix} \boldsymbol{x}_i^{(i-1)} & \boldsymbol{y}_i^{(i-1)} & \boldsymbol{z}_i^{(i-1)} & \boldsymbol{p}_{i-1,i}^{(i-1)} \\ 0 & 0 & 0 & 1 \end{bmatrix} = \begin{pmatrix} \cos\theta_i & -\sin\theta_i\cos\alpha_i & \sin\theta_i\sin\alpha_i & a_i\cos\theta_i \\ \sin\theta_i & \cos\theta_i\cos\alpha_i & -\cos\theta_i\sin\alpha_i & a_i\sin\theta_i \\ 0 & \sin\alpha_i & \cos\alpha_i & d_i \\ 0 & 0 & 0 & 1 \end{pmatrix} \tag{3.1} $$

The homogeneous matrices ${}_{i-1}^{i}\boldsymbol{T}$ are called Denavit-Hartenberg matrices.
The rotation matrix ${}_{i-1}^{i}\boldsymbol{A}$ between K_i and K_{i-1} is the upper left (3,3)-matrix:

$$ {}_{i-1}^{i}\boldsymbol{A} = \left[\boldsymbol{x}_i^{(i-1)} \quad \boldsymbol{y}_i^{(i-1)} \quad \boldsymbol{z}_i^{(i-1)}\right] = \begin{bmatrix} \cos\theta_i & -\sin\theta_i\cos\alpha_i & \sin\theta_i\sin\alpha_i \\ \sin\theta_i & \cos\theta_i\cos\alpha_i & -\cos\theta_i\sin\alpha_i \\ 0 & \sin\alpha_i & \cos\alpha_i \end{bmatrix} \tag{3.2}$$

Concenating link transformations to calculate the single transformation that relates frame *n* (usually the TCP frame, *n* is the number of joints) to frame 0 (usually the base frame):

$$\boldsymbol{T}_W = {}_0^n\boldsymbol{T} = {}_0^1\boldsymbol{T} \cdot {}_1^2\boldsymbol{T} \cdots {}_{n-2}^{n-1}\boldsymbol{T} \cdot {}_{n-1}^{n}\boldsymbol{T} \tag{3.3}$$

3.3.3 Generalized Coordinates

In case of a rotational joint *i* the joint variable (joint coordinate) is the angle θ_i. The DH-matrix ${}_{i-1}^{i}\boldsymbol{T}$ only depends on the joint angles θ_i, since the parameters a_i, d_i, α_i are constants. In the case of a prismatic joint the joint variable is d_i, the other DH-parameters are constant. To get an uniform notation, the generalized coordinate q_i is introduced and all joint coordinates are described as vector $\underline{q}$. It holds:

$q_i = \theta_i$, if joint i is a rotational joint, $q_i = d_i$, if joint i is a prismatic/translational joint,
$\boldsymbol{q} = (q_1, q_2, \cdots, q_n)^{\mathrm{T}}$

3.4 Examples for Denavit-Hartenberg Konvention

Fig. 3.4 shows a robot with two parallel revolute axes. The TCP can be positioned in the $x_0 - y_0$ - plane.

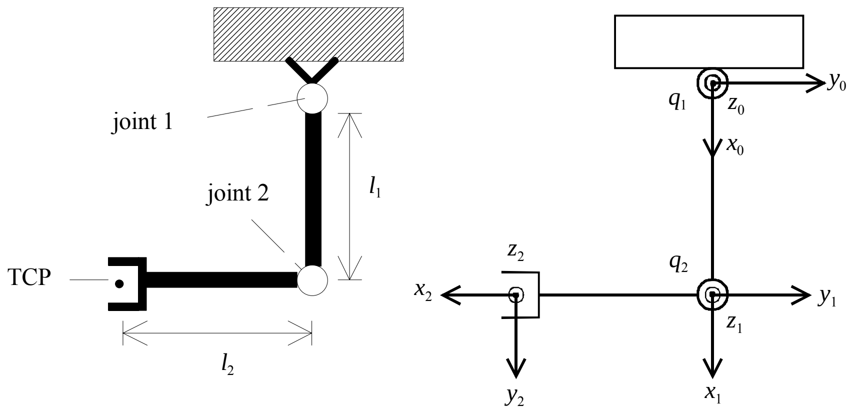


Figure 3.4 Robot with 2 joints (right side as sketch with coordinate systems)

The following table shows the set of DH parameters: $q_1 = \theta_1, q_2 = \theta_2$ are variables (joint angles)

Gelenk	$q_i = \theta_i$ /degree	d_i	a_i	α_i / degree
1	0	0	h1	0
2	-90	0	h2	0

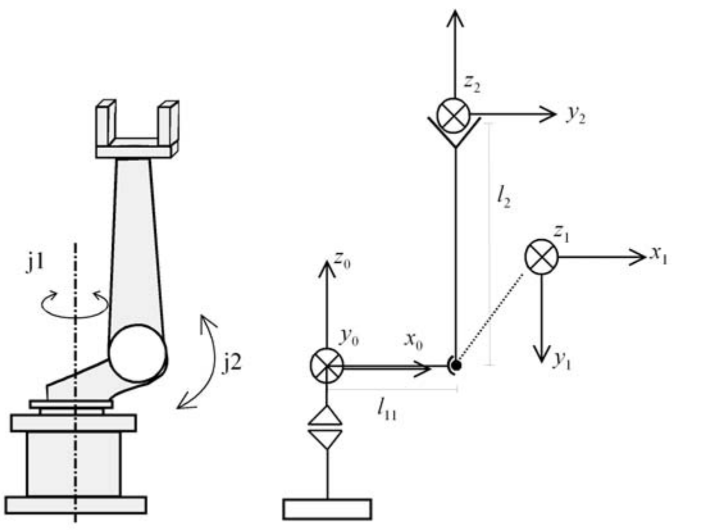
D.H. matrices of the robot in fig. 3.4:

$$ {}^{1}_{0}\boldsymbol{T} = \begin{pmatrix} \cos q_1 & -\sin q_1 & 0 & h_1 \cdot \cos q_1 \\ \sin q_1 & \cos q_1 & 0 & h_1 \cdot \sin q_1 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}, \quad {}^{2}_{1}\boldsymbol{T} = \begin{pmatrix} \cos q_2 & -\sin q_2 & 0 & h_2 \cdot \cos q_2 \\ \sin q_2 & \cos q_2 & 0 & h_2 \cdot \sin q_2 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} $$

$$ {}^{2}_{0}\boldsymbol{T} = \boldsymbol{T}_W(\boldsymbol{q}) = \begin{pmatrix} C_1C_2 - S_1S_2 & -(C_1S_2 + S_1C_2) & 0 & h_2(C_1C_2 - S_1S_2) + h_1C_1 \\ S_1C_2 + C_1S_2 & -S_1S_2 + C_1C_2 & 0 & h_2(S_1C_2 + C_1S_2) + h_1S_1 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \qquad \begin{matrix} C_i = \cos q_i \\ S_i = \sin q_i \end{matrix} $$

Robot with two perpendicular axes („RR-sphere“)

Fig. 3.5 shows a robot with perpendicular joints. It can be considered as a robot which consist of the two first joints of a typical industrial robot with 6 revolute axis.



joint	θ_i	d_i	a_i	α_i
1	0	0	l_{11}	-90°
2	-90°	0	l_2	0

Fig 3.5 Robot with two axes with perpendicular axes (RR-sphere), $q_1 = \theta_1, q_2 = \theta_2$ joint variables a_1, a_2, d_1, d_2, α_1, α_2 constant

robot with 2 rotational joints and one prismatic joint („RRT“, Fig. 3.6)

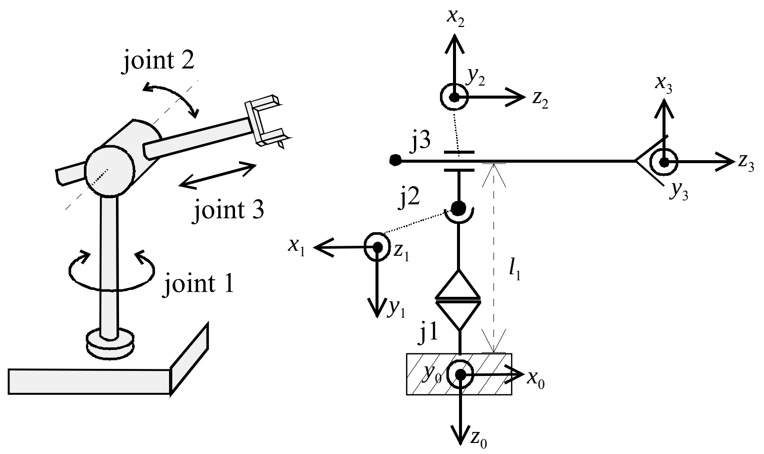


Figure 3.6 Robot with two rotational joints and one prismatic joint (RRT)

DH-parameter of the robot with 2 rotational and one prismatic (sliding) joint

joint	q_i / degree	d_i	a_i	α_i / degree
1	180	-h1	0	+90
2	-90	0	0	+ 90
3	0	d*	0	0

q_1, q_2, d_3: variable, a_1, a_2, a_3, d_1, d_2, α_1, α_2, α_3 constant

robot with 6 revolute joints (articulated robot), see fig. 3.7.

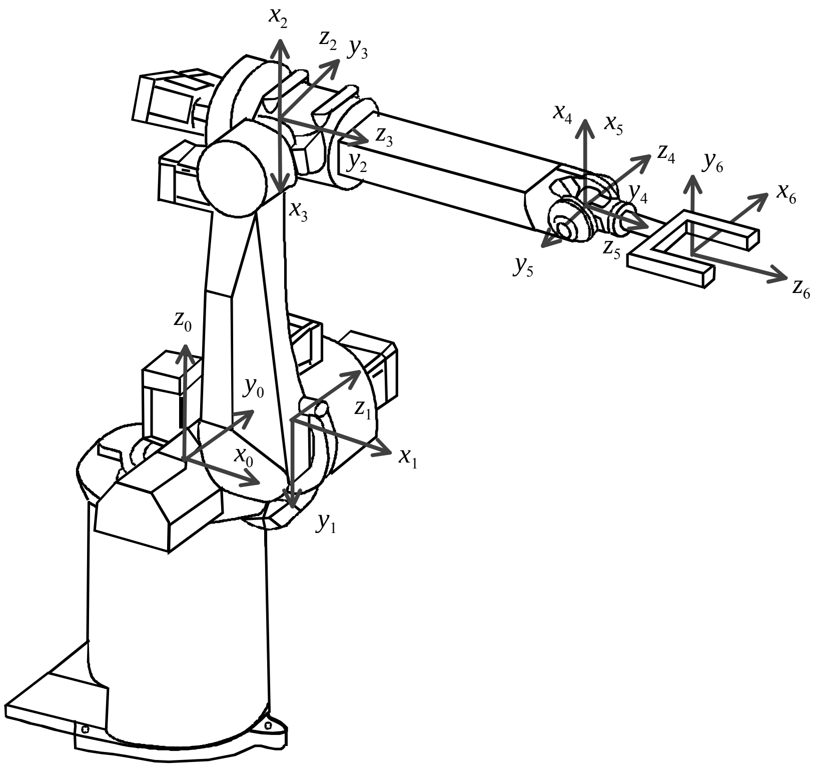


Figure 3.7 articulated robot arm, 6 revolute joints

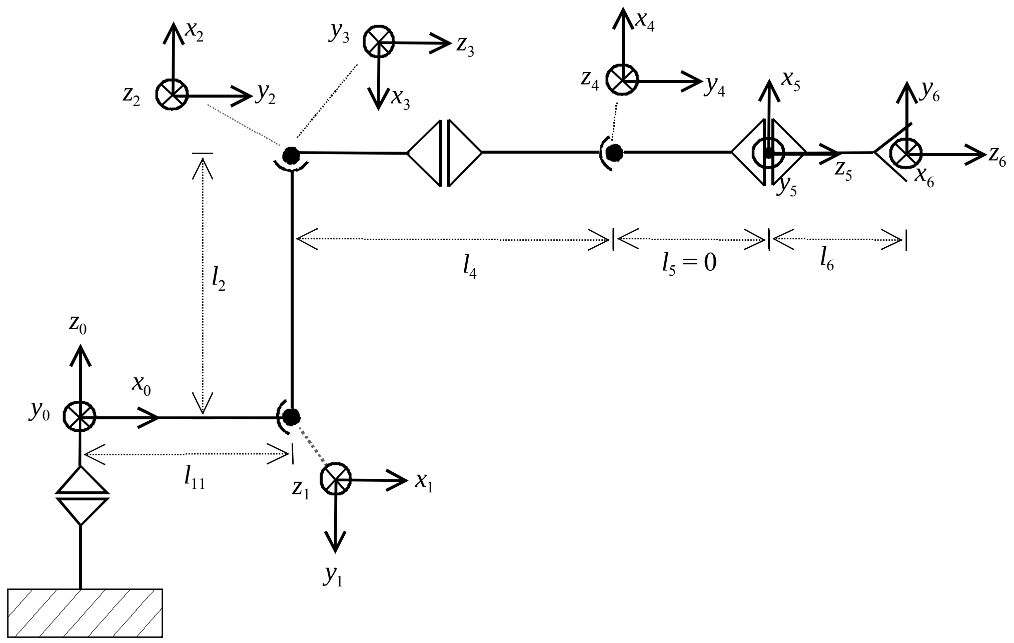


Figure 3.8 articulated robot sketch

3.5 Mapping joint coordinates to Cartesian coordinates and vice versa

3.5.1 Direct and inverse kinematics

The world coordinates are defined by the position vector $\boldsymbol{p}=\boldsymbol{p}_{0w}$ from the base of K_0 to the TCP expressed in K_0 and the orientation of the tool by specification of the unit vectors $\boldsymbol{x}_W, \boldsymbol{y}_W, \boldsymbol{z}_W$ of the tool coordinate system in coordinates of K_0. As example see Fig. 3.9.
The orientation of the tool can also be expressed by three angles. Mostly the Euler-angles are used (but there are several definitions (see for ex. W. Weber: Industrieroboter, Hanser, 2nd ed., 2009 or Craig: Introduction to Robotics, 3rd ed.). For example we can use the Euler-Angles Z-Y-X, which are called A, B, C (see ch. 2.5) Fig. 3.10 shows in a schematic way the mapping from robot coordinates to world coordinates, the **forward** or **direct kinematics**, and the mapping from the world coordinates to the robot coordinates, **the inverse kinematics**.

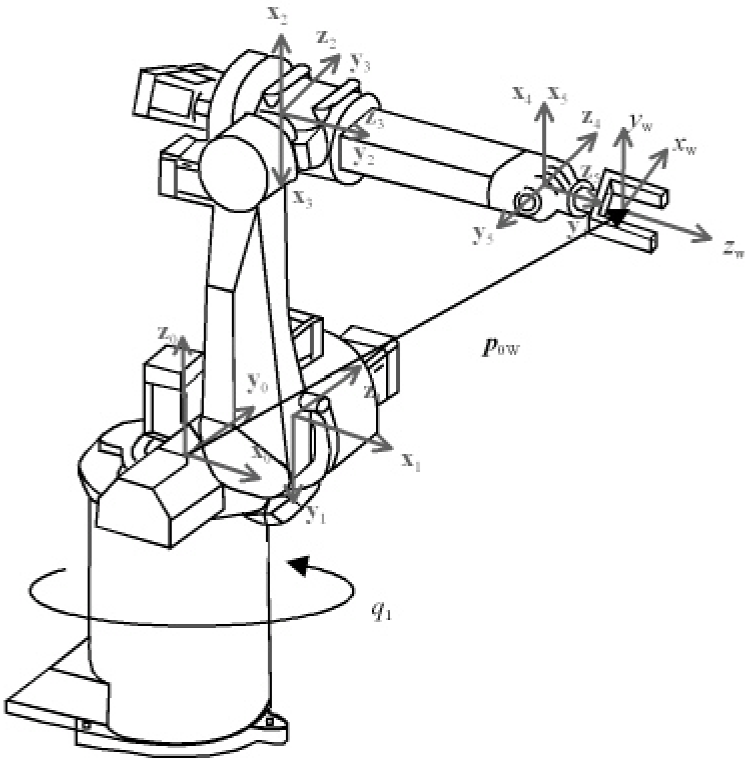


Figure 3.9: Six joint robot with world coordinate system K_0, tool coordinate system K_W and position vector $\boldsymbol{p}_{ow}$

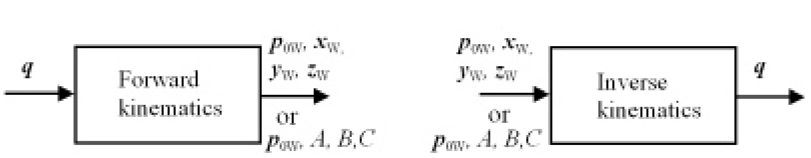


Figure 3.10: Mapping of robot coordinates and world coordinates

While the forward kinematics is a unique transformation the inverse kinematics has the problem of multiple solutions and infinitive solutions (singularities). Figure 3.11 shows an example for multiple solutions (left) and for a singularity. In the sketched case an infinitive number of pairs (q_4, q_6) can perform the demanded orientation of $K_6 = K_W$.

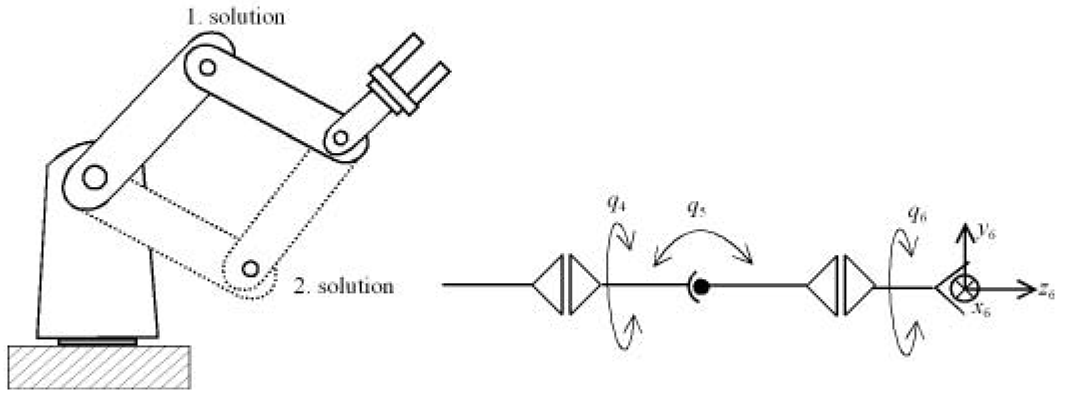


Figure 3.11. Simple example for multiple solutions (left) and example for a singularity

3.5.2 Mapping joint velocities to Cartesian coordinates and vice versa with the Jacobian

If we condense all Cartesian velocities in the vector $\dot{\boldsymbol{x}} = \begin{pmatrix} \dot{p}_x & \dot{p}_y & \dot{p}_z & \dot{A} & \dot{B} & \dot{C} \end{pmatrix}^{\mathrm{T}}$ the mapping between Cartesian velocities and joint velocities can be expressed with

$$\dot{\boldsymbol{x}} = \boldsymbol{J}_0(\boldsymbol{q}) \cdot \dot{\boldsymbol{q}} \tag{3.4}$$

$\boldsymbol{J}_0(\boldsymbol{q})$ is called the Jacobian and has the dimension (6,n) if n is the number of joints. If the manipulator has 6 joints we can solve for the joint velocities if the Cartesian velocities are given:

$$\dot{\boldsymbol{q}} = \boldsymbol{J}_0^{-1}(\boldsymbol{q}) \cdot \dot{\boldsymbol{x}} \tag{3.5}$$

In the case of a singularity position it holds $\det(\boldsymbol{J}_0(\boldsymbol{q})) = 0$ and the inverse does not exist.
See examples in lecture.

3.5.3 Mapping external forces/torques to joint forces/torques

An external force vector $\boldsymbol{f}_{ex}$ and torque momentum vector $\boldsymbol{n}_{ex}$ to the tool of a robot produces equivalent forces torques in the joint condensed in the vector $\boldsymbol{\tau}_{ex}$ (see Fig. 3.12). This relation is written as

$$\boldsymbol{\tau}_{ex} = \boldsymbol{J}_0^{\mathrm{T}}(\boldsymbol{q}) \cdot \begin{pmatrix} \boldsymbol{f}_{ex}^{(0)} \\ \boldsymbol{n}_{ex}^{(0)} \end{pmatrix} \tag{3.6}$$

The matrix $\boldsymbol{J}_0^{\mathrm{T}}(\boldsymbol{q})$ is the transposed Jacobian and has the dimension (n,6). If $\boldsymbol{f}_{ex}$ and $\boldsymbol{n}_{ex}$ are given respectively defined as vectors described in the tool coordinate system K_W we must apply the rotation matrix ${}_0^W\boldsymbol{A}$ and get:

$$\begin{aligned}
\boldsymbol{\tau}_{ex} &= \boldsymbol{J}_0^{\mathrm{T}}(\boldsymbol{q})\cdot\begin{pmatrix}\boldsymbol{f}_{ex}^{(0)}\\ \boldsymbol{n}_{ex}^{(0)}\end{pmatrix} = \boldsymbol{J}_0^{\mathrm{T}}(\boldsymbol{q})\cdot\begin{pmatrix}{}_0^W\boldsymbol{A}\cdot\boldsymbol{f}_{ex}^{(W)}\\ {}_0^W\boldsymbol{A}\cdot\boldsymbol{n}_{ex}^{(W)}\end{pmatrix} = \boldsymbol{J}_0^{\mathrm{T}}(\boldsymbol{q})\cdot\begin{pmatrix}{}_0^W\boldsymbol{A} & \boldsymbol{0}\\ \boldsymbol{0} & {}_0^W\boldsymbol{A}\end{pmatrix}\cdot\begin{pmatrix}\boldsymbol{f}_{ex}^{(W)}\\ \boldsymbol{n}_{ex}^{(W)}\end{pmatrix}\rightarrow\\
\boldsymbol{\tau}_{ex} &= \boldsymbol{J}_W^{\mathrm{T}}(\boldsymbol{q})\cdot\begin{pmatrix}\boldsymbol{f}_{ex}^{(W)}\\ \boldsymbol{n}_{ex}^{(W)}\end{pmatrix},\ \text{with } \boldsymbol{J}_W^{\mathrm{T}}(\boldsymbol{q}) = \boldsymbol{J}_0^{\mathrm{T}}(\boldsymbol{q})\cdot\begin{pmatrix}{}_0^W\boldsymbol{A} & \boldsymbol{0}\\ \boldsymbol{0} & {}_0^W\boldsymbol{A}\end{pmatrix}
\end{aligned} \tag{3.7}$$

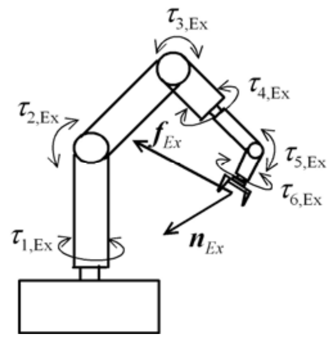


Figure 3.12: Mapping of external forces/torques to joint forces/torques

3.6 Evaluation of the Jacobian matrix

Will be add in next time

4. Model of robot arm and drive system

In chapter 1 we have mentioned that a single axis control scheme is limited in control performance. In chapter 5 we will introduce more advanced control techniques which guarantee robust control responses with better tracking of arbitrary trajectories and approximately equal control responses in the whole workspace. But these control design methods must be based on a mathematical model of the robot arm and drive system which includes the nonlinear effects of couplings among the joints. The main part of such a model is the dynamic model of the robot arm.

4.1 Robot arm dynamics

The dynamic model of a robot arm is a multi-variable system as outlined in Fig. 4.1.

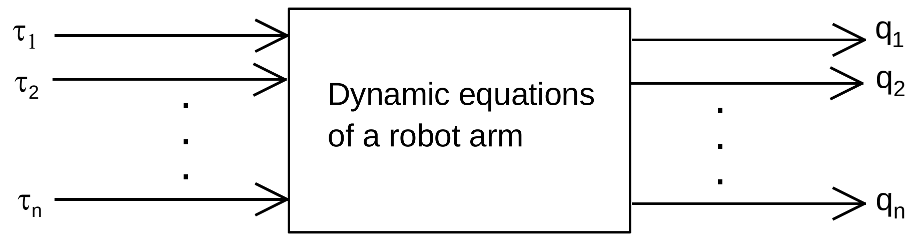


Figure 4.1. Multi-variate dynamics of a robot arm

The mathematical model of the arm dynamics describes how the robot arm is moving under the influence of the torques and forces acting on the joints. The model is set up by application of the energy conservation laws or the Newton's equation. In robotics two methods are important: The Euler-Lagrange Equations, which are based on the difference of the kinetic and potential energy of the system and the Newton-Euler formulation, which is based on the free body diagram of each link. Here we will only consider the Newton-Euler formulation, it seems the easier and more effective method for control tasks.
The relations are nonlinear, because trigonometric functions and squares and multiplications of time depending variables occur (see chap. 1.2). In the next section first the terms „equations of motion" and "inverse model" will be described.

4.1.1 Equations of motion and Inverse Model of mechanical systems

In Fig. 4.2 we consider first a simple mechanical system of a mass *m*. A force *F* is acting in x-direction. Additional a velocity dependent friction with coefficient F_D acts on *m*. The force balance gives the differential equation:

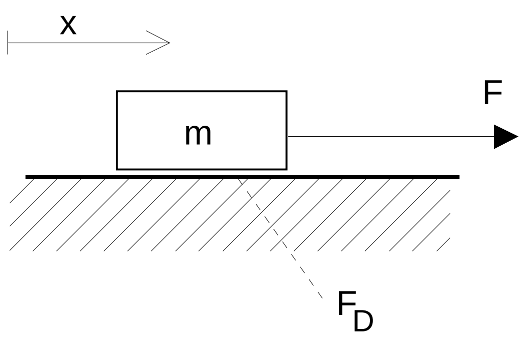
$$m \cdot \ddot{x} = F - F_D \cdot \dot{x}$$

If we assume that the motion $\dot{x}, \ddot{x}$ is measured and we have to calculate the force which causes this motion, we can do that by the **inverse Model**

$$F = m \cdot \ddot{x} + F_D \cdot \dot{x}$$

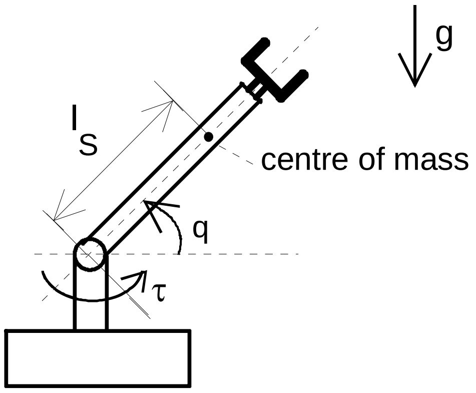
If we know the time evaluation of the force and if we have to calculate the motion, we can do that by solving the **equations of motion**:

$$\ddot{x} = \frac{1}{m} \cdot (F - F_D \cdot \dot{x})$$



inverse model equation of motion

$$F = m \cdot \ddot{x} + F_D \cdot \dot{x} \qquad \ddot{x} = m^{-1}[F - F_D \cdot \dot{x}]$$



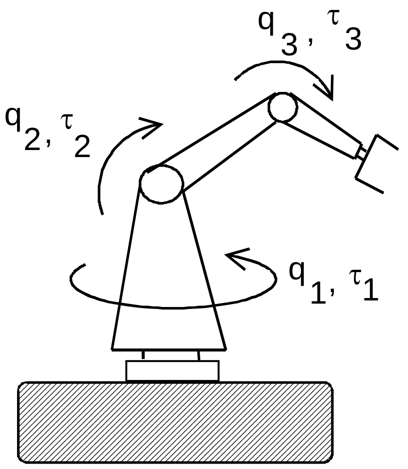
inverse mod*el* :

$$\tau = M \cdot \ddot{q} + F_D \cdot \dot{q} + m \cdot g \cdot l_s \cdot \cos q$$

equation of motion :

$$\ddot{q} = M^{-1}[\tau - F_D \cdot \dot{q} - m \cdot g \cdot l_s \cdot \cos q]$$

M: inertia due to joint
m: mass



inverse mod*el*:

$$\boldsymbol{\tau} = \boldsymbol{M}(\boldsymbol{q}) \cdot \ddot{\boldsymbol{q}} + \boldsymbol{b}(\boldsymbol{q}, \dot{\boldsymbol{q}})$$

equation of motion:

$$\ddot{\boldsymbol{q}} = \boldsymbol{M}(\boldsymbol{q})^{-1}[\boldsymbol{\tau} - \boldsymbol{b}(\boldsymbol{q}, \dot{\boldsymbol{q}})]$$

Figure 4.2 Inverse model and multi-variate dynamics of a robot arm

For the one joint model in fig. 4.2 we get for the momentum balance:

$$M \cdot \ddot{q} = \tau - F_D \cdot \dot{q} - m \cdot g \cdot l_s \cdot \cos q$$

With the torque τ on the left side we get the inverse model and if we solve for the supreme derivation we get the equations of motion (see Fig. 4.2). In summary we can define:

<u>Inverse model:</u>
The forces or torques, required to cause a given or desired motion, are evaluated.

<u>Equation of motion:</u>
The motion, which arises from the acting forces/moments, is evaluated.

In Fig. 4.2 also inverse model and equations of motion of a robot arm is indicated. Due to the multivariate behaviour of a robot arm the matrix form must be used. The mass matrix $\underline{M}$ depends on the joint coordinates written as Vektor $\boldsymbol{q}$.

The objective of inverse model of a robot arm is to design suitable model based control laws, while the equations of motions are mainly used for simulation purposes. Even from the single joint model it can be seen, that the equations are nonlinear because trigonometric functions arises.

4.1.2 Structur of robot arms with open kinematical chains

We will us concentrate on robot arms with open kinematical chains. Each link is connected with the following link by one joint with a single axis. In Fig. 4.3 the joints and links of a simple robot with 3 joints are numerated. The base does not move. The effector is the *n*-th link of a robot with *n* joints.

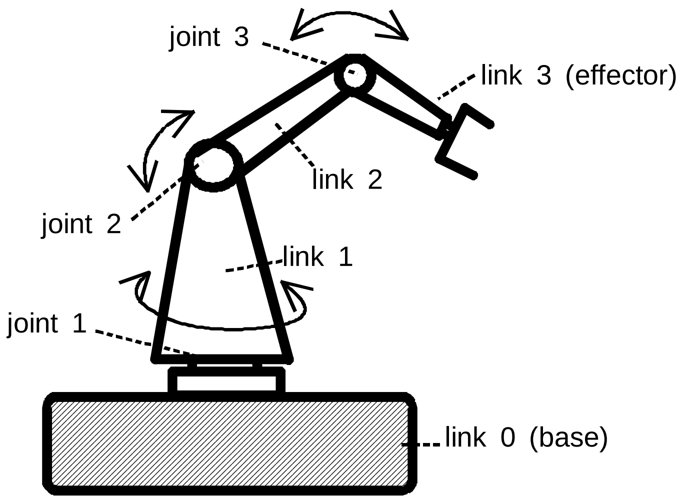


Figure 4.3 Structure of a robot arm with an open kinematic chain .

If there are joints with more than one axis (for example a ball joint), then links with no masses and length 0 between two single joint axis are assumed. In this way the above formulated condition for open kinematical chains is satisfied (see Fig. 4.4).

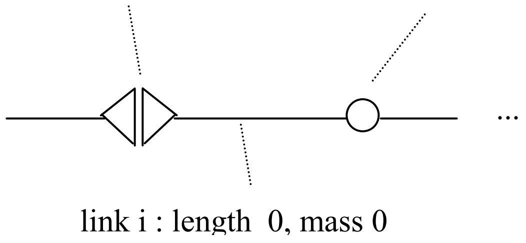


Figure 4.4 Link with no masses and length 0 between two joint axes

For each link a coordinate frame according to the Denavit-Hartenberg notation is established (see chap. 3).

4.1.3 Principles of Newton-Euler method and notations

The Newton-Euler formulation is a method to establish the inverse model of a robot. The objective is to calculate the vector of torques/forces acting on each link dependent on the vectors $\boldsymbol{q}$, $\dot{\boldsymbol{q}}$, $\ddot{\boldsymbol{q}}$ of joint coordinates, velocities of joint coordinates and accelerations of joint coordinates. Of course the acting torques are depending on the constant parameters of length, mass and inertia of each link (Fig. 4.5).

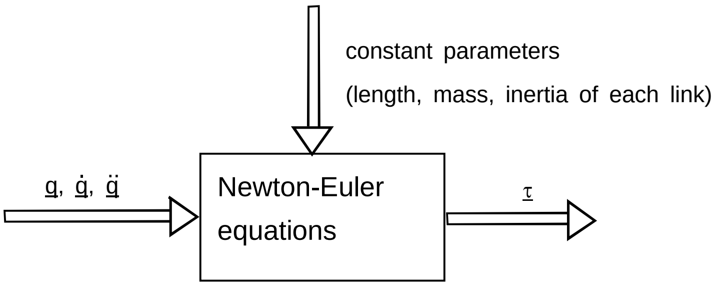


Figure 4.5 Objective of the Newton-Euler formulation

In the Newton-Euler formulation we treat each link in turn. With the kinematical part of the Newton-Euler equations the motion of each link is established. This motion depends on the velocities, angular velocities and accelerations of the joints. The general motion of a rigid body like a link *i* of the robot in relation to a non moving coordinate frame can be described by the velocity vector $\boldsymbol{v}_{S,i}$ of the centre of mass, the angular motion $\boldsymbol{\omega}_i$ and the corresponding accelerations (Fig. 4.6).

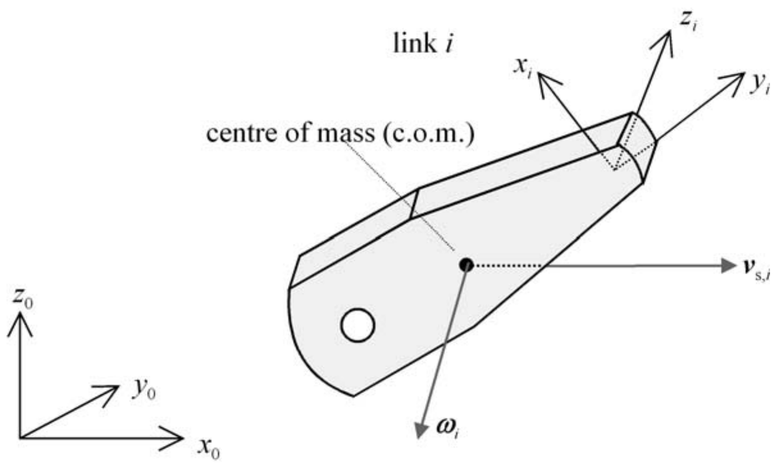


Figure 4.6. General motion of a rigid body (here link *i*)

The direction of $\boldsymbol{\omega}_i$ is the direction of the instantaneous rotary axis. The absolute value $|\boldsymbol{\omega}_i|$ is the angular velocity. All points of a rigid body have the same angular velocity. $\boldsymbol{v}_\mathrm{i}$ is the velocity of the base of frame K_i. $\boldsymbol{\omega}_i$, $\boldsymbol{v}_\mathrm{i}$ and the corresponding accelerations are absolute values in relation to the non moving space.

Agreement at Newton-Euler-equations: all vectors of the *i*-th link are described in the coordinates of the frame K_i, which is fixed at link *i* although they are values in relation to the non moving space. $\boldsymbol{v}_\mathrm{i}$ stands for $\boldsymbol{v}_i^{(i)}$ etc. Especially it holds: $\mathbf{z}_i = \mathbf{z}_k = \mathbf{z}_0 = \begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix} = \mathbf{z}$.

In Fig. 4.7 the kinematical relationship between two neighbouring links is depicted. The vector $\boldsymbol{p}_{i+1}$ is directed from the base of K_i to the base of K_{i+1}. The vector $\boldsymbol{s}_i$ is directed from the base of frame K_i to the centre of mass of link *i*. $\boldsymbol{p}_i$ and $\boldsymbol{s}_i$ are written in coordinates of frame K_i, $\boldsymbol{s}_i$ is a constant vector, if joint i is a revolute joint $\boldsymbol{p}_i$ is constant too. For a prismatic joint $\boldsymbol{p}_i$ depends on the joint coordinate $d_i = q_i$. $\boldsymbol{r}_\mathrm{i}$ and $\boldsymbol{r}_{s,i}$ are vectors from the base of the non moving frame K_0 to the base of frame K_i respectively to the centre of mass of link *i*. These vectors are time dependent.

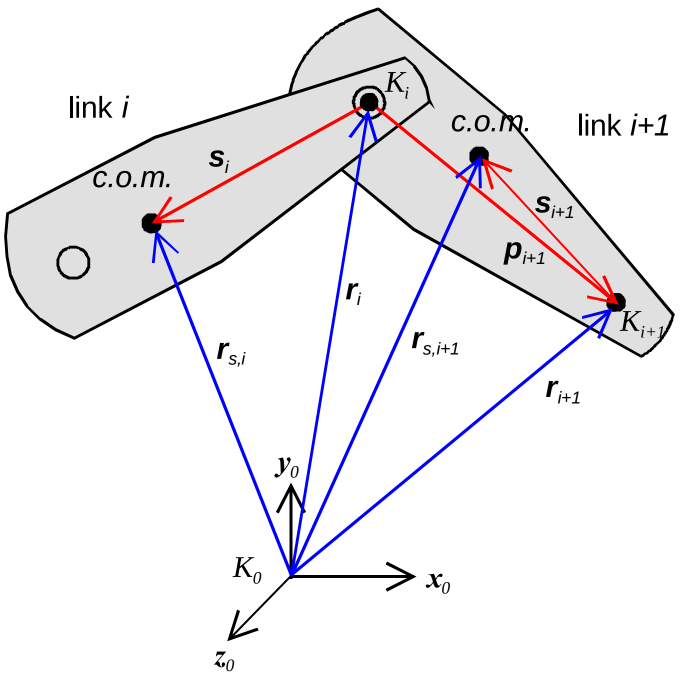


Figure 4.7 Kinematical relations between two neighbouring links.

Fig. 4.8 shows link *i* in a free body diagram. $\boldsymbol{n}_i$ respectively $\boldsymbol{f}_i$ are the vector of force respectively the vector of angular momentum which act on link *i* exerted of link *i*-1. So $\boldsymbol{f}_{i+1}$ and $\boldsymbol{n}_{i+1}$ are the vectors of forces and torques exerted from link *i* to link *i*+1. With the Newton law „actio=reactio" the negative vectors $-\boldsymbol{f}_{i+1}$ and $-\boldsymbol{n}_{i+1}$ are the force vector respectively angular momentum vector which are exerted from link *i*+1 to link *i*.

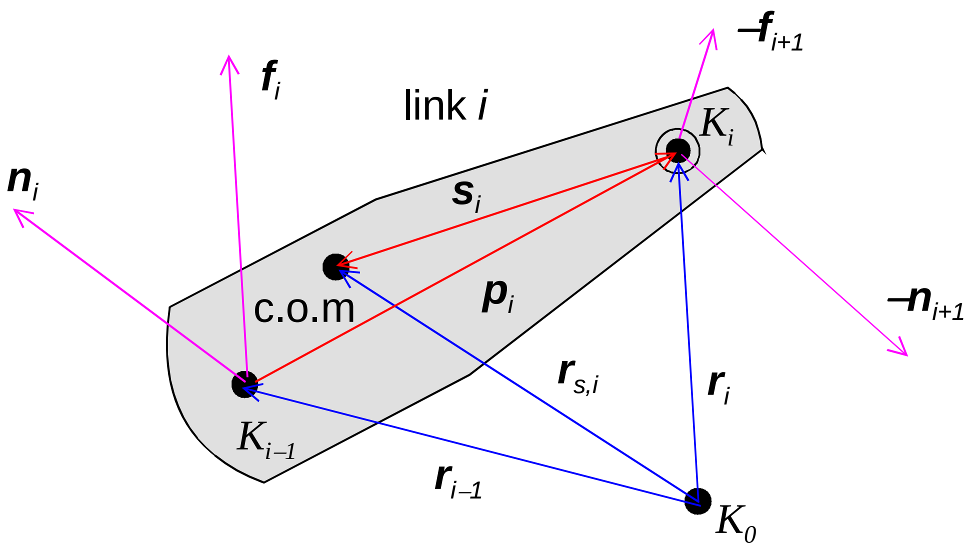


Figure 4.8. Free body diagram of link i

4.1.4 The recursive Newton-Euler equations

First the kinematical calculations are executed. For a revolute joint *i* we set $h_i = 1$ for a prismatic joint $h_i = 0$. Beginning at link 0 (non moving base) the vectors $\boldsymbol{\omega}_i, \dot{\boldsymbol{\omega}}_i, \boldsymbol{v}_i, \dot{\boldsymbol{v}}_i, \dot{\boldsymbol{v}}_{si}$ for each link are calculated in a recursive manner. Going out from Fig. 4.7 we hold:

$$\begin{aligned}
\boldsymbol{\omega}_{i+1} &= {}_{i+1}^{\;\;i}\boldsymbol{A}(\boldsymbol{\omega}_i + h_{i+1}\cdot \boldsymbol{z}\cdot \dot{q}_{i+1}),\\
\dot{\boldsymbol{\omega}}_{i+1} &= {}_{i+1}^{\;\;i}\boldsymbol{A}[\dot{\boldsymbol{\omega}}_i + h_{i+1}\cdot(\boldsymbol{z}\cdot \ddot{q}_{i+1} + \boldsymbol{\omega}_i \times \boldsymbol{z}\cdot \dot{q}_{i+1})],\\
\boldsymbol{v}_{i+1} &= {}_{i+1}^{\;\;i}\boldsymbol{A}\cdot \boldsymbol{v}_i + (1-h_{i+1})\cdot \frac{\mathrm{d}^{(i+1)}}{\mathrm{d}t}\boldsymbol{p}_{i+1} + \boldsymbol{\omega}_{i+1}\times \boldsymbol{p}_{i+1},
\end{aligned}$$

$$\begin{aligned}
\dot{\boldsymbol{v}}_{i+1} = {}&{}_{i+1}^{\;\;i}\boldsymbol{A}\cdot \dot{\boldsymbol{v}}_i + \dot{\boldsymbol{\omega}}_{i+1}\times \boldsymbol{p}_{i+1} + \boldsymbol{\omega}_{i+1}\times(\boldsymbol{\omega}_{i+1}\times \boldsymbol{p}_{i+1}) + \\
&+ (1-h_{i+1})\cdot\left(\frac{\mathrm{d}^{2,(i+1)}}{\mathrm{d}t^2}\boldsymbol{p}_{i+1} + 2\cdot \boldsymbol{\omega}_{i+1}\times \frac{\mathrm{d}^{(i+1)}}{\mathrm{d}t}\boldsymbol{p}_{i+1}\right),
\end{aligned} \tag{4.1}$$

$$\begin{aligned}
\boldsymbol{v}_{s,i+1} &= \boldsymbol{v}_{i+1} + \boldsymbol{\omega}_{i+1}\times \boldsymbol{s}_{i+1},\\
\dot{\boldsymbol{v}}_{s,i+1} &= \dot{\boldsymbol{v}}_{i+1} + \dot{\boldsymbol{\omega}}_{i+1}\times \boldsymbol{s}_{i+1} + \boldsymbol{\omega}_{i+1}\times(\boldsymbol{\omega}_{i+1}\times \boldsymbol{s}_{i+1})
\end{aligned}$$

To calculate the values of link 1 we need the initial values of the base:

$$\boldsymbol{\omega}_0 = \boldsymbol{0}\,,\ \dot{\boldsymbol{\omega}}_0 = \boldsymbol{0},\ \boldsymbol{v}_0 = \boldsymbol{0}\,,\ \dot{\boldsymbol{v}}_0 = -\boldsymbol{g}$$

The acceleration vector of the base has the absolute value of the gravitational acceleration but in negative direction. The purpose of this starting value is explained below. As we have to treat vectors which are written in different coordinate frames, we must use the rotational matrices ${}_{i-1}^{\;\;i}\boldsymbol{A}$ respectively ${}_{\;\;i}^{i-1}\boldsymbol{A} = {}_{i-1}^{\;\;i}\boldsymbol{A}^T$ .This matrices depend on the respective joint coordinates.
After the kinematical equations (4.1) are executed the motion of all links are known. Now the forces, angular momentums and torques caused by the drives can be calculated by (4.2). The equations are the derivative from Fig. 4.8. The recursive calculations start from the last link *n* (effector) and end at link 1:

$$\begin{aligned}
&\boldsymbol{F}_i = m_i\cdot \dot{\boldsymbol{v}}_{s,i} + (1-h_i)\cdot \hat{F}_{D,i}\,{}_{\;\;i}^{i-1}\boldsymbol{A}\cdot \boldsymbol{z}\cdot \dot{q}_i,\\
&\boldsymbol{N}_i = \boldsymbol{I}_{\mathrm{SP},i}\cdot \dot{\boldsymbol{\omega}}_i + \boldsymbol{\omega}_i\times(\boldsymbol{I}_{\mathrm{SP},i}\cdot \boldsymbol{\omega}_i) + h_i\cdot F_{D,i}\cdot {}_{\;\;i}^{i-1}\boldsymbol{A}\cdot \boldsymbol{z}\cdot \dot{q}_i,\\
&\boldsymbol{f}_i = {}_{\;\;i}^{i+1}\boldsymbol{A}\cdot \boldsymbol{f}_{i+1} + \boldsymbol{F}_i,\\
&\boldsymbol{n}_i = {}_{\;\;i}^{i+1}\boldsymbol{A}\cdot[\boldsymbol{n}_{i+1} + ({}_{i+1}^{\;\;i}\boldsymbol{A}\cdot \boldsymbol{p}_i)\times \boldsymbol{f}_{i+1}] + (\boldsymbol{p}_i + \boldsymbol{s}_i)\times \boldsymbol{F}_i + \boldsymbol{N}_i,\\
&\boldsymbol{\tau}_i = [h_i\cdot \boldsymbol{n}_i^T + (1-h_i)\cdot \boldsymbol{f}_i^T]\cdot {}_{\;\;i}^{i-1}\boldsymbol{A}\cdot \boldsymbol{z},\\
&\text{initial values:}\ \boldsymbol{f}_{n+1} = -\boldsymbol{f}_{ex}, \boldsymbol{n}_{n+1} = -\boldsymbol{n}_{ex}
\end{aligned}\ , \tag{4.2}$$

As initial values the external force vector $\boldsymbol{f}_{n+1}$ = - $\boldsymbol{f}_{Ex}$ and the external angular momentum vector $\boldsymbol{n}_{n+1}$ = - $\boldsymbol{n}_{ex}$ which act on the effector can be used (*n*: number of links).

The inertia tensor $\boldsymbol{I}_{SP,i}$ .can be thought as the generalization of the scalar moment of inertia acting on one fixed axis. While inertia tensors may be defined relative to any frame we will always consider the case of the inertia tensor defined for the frame K_i attached at the centre of mass of link *i* (Fig. 4.9).

The inertia tensor

$$\boldsymbol{I}_{\mathrm{SP},i} = \begin{pmatrix} I_{xx,i} & I_{xy,i} & I_{xz,i} \\ I_{yx,i} & I_{yy,i} & I_{yz,i} \\ I_{zx,i} & I_{zy,i} & I_{zz,i} \end{pmatrix}$$

is a symmetric matrix. The index *i* for the elements is omitted and the elements are given by

$$\begin{aligned} I_{xx,i} &= \int (y_i^2 + z_i^2)\,\mathrm{d}m_i, \quad I_{yy,i} = \int (x_i^2 + z_i^2)\,\mathrm{d}m_i, \quad I_{zz,i} = \int (x_i^2 + y_i^2)\,\mathrm{d}m_i, \\ I_{xy,i} &= -\int (x_i \cdot y_i)\,\mathrm{d}m_i, \quad I_{xz,i} = -\int (x_i \cdot z_i)\,\mathrm{d}m_i, \quad I_{yz,i} = -\int (y_i \cdot z_i)\,\mathrm{d}m_i \end{aligned} \tag{4.3}$$

In which the rigid body link *i* is composed of differential volume elements d*v*, containing material elements of density ρ. The elements I_{xx}, I_{yy}, I_{zz} are called mass moments of inertia. The elements of mixed indices are called the mass products of inertia. Since the inertia tensors are defined relative to a body fixed frame, the tensors $\boldsymbol{I}_{Sp,i}$ are constant. In most cases the mass products of inertia can be neglected.

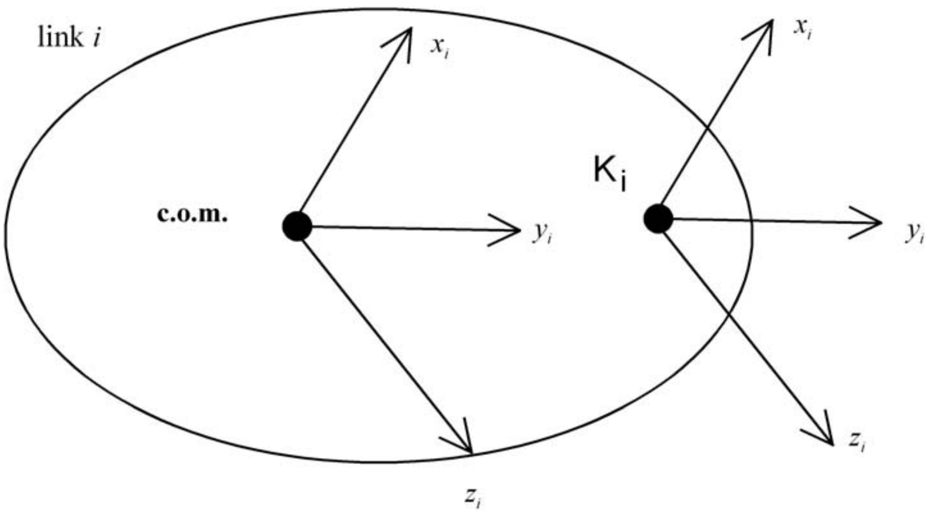


Figure 4.9 Axes of inertia: direction of the axes of frame K_i, origin in the centre of mass

In practice the geometry and composition of the links are somewhat complex, so that the application is difficult. Though it may be better to measure the moments of inertia or to divide the link in simple bodies like cuboids and cylinder and compose the inertia tensors in a adequate manner.

In the dynamic equations of (4.2) no gravity loading is included. This effect can be included quite simply by setting $\dot{\boldsymbol{v}}_0 = -\mathbf{g}$. This is equivalent by saying that the base of the robot is accelerating upward with 1 g (g=9.81 m/sec²). This fictitious upward acceleration causes exactly the same effect on links as gravity would.

With (4.1) and (4.2) the objective of the Newton-Euler method according to Fig. 4.5 is obtained. On the base of the initial values and the joint variables the vector $\boldsymbol{\tau}$ of the torques, acting on the joints, can be numerically calculated. The amount of calculation is 150n-48 multiplications and 131n-48 summations for a robot arm with *n* links. But for the simulation tasks and for investigation of special effects we need the inverse model in a closed form:

$$\boldsymbol{\tau} = \boldsymbol{M}(\boldsymbol{q}) \cdot \ddot{\boldsymbol{q}} + \boldsymbol{b}(\boldsymbol{q}, \dot{\boldsymbol{q}}) \tag{4.4a}$$

with the position dependent inertia matrix ***M***(***q***) and the vector ***b***. The vector ***b*** can be divided in the vector ***G***(***q***) caused by gravity, the vector $\boldsymbol{C}(\boldsymbol{q}, \dot{\boldsymbol{q}})$ of Centripetal- and Coriolis torques, the n◌n-matrix ***R*** weights the angular velocity vector to obtain the friction torques:

$$\boldsymbol{\tau} = \boldsymbol{M}(\boldsymbol{q}) \cdot \ddot{\boldsymbol{q}} + \boldsymbol{b}(\boldsymbol{q}, \dot{\boldsymbol{q}}) = \boldsymbol{M}(\boldsymbol{q}) \cdot \ddot{\boldsymbol{q}} + \boldsymbol{G}(\boldsymbol{q}) + \boldsymbol{C}(\boldsymbol{q}) \cdot \boldsymbol{f}_C(\dot{\boldsymbol{q}}) + \boldsymbol{R}(\boldsymbol{q}) \cdot \dot{\boldsymbol{q}} \tag{4.4b}$$

If $\boldsymbol{b}(\boldsymbol{q}, \dot{\boldsymbol{q}})$ and ***M***(***q***) is explicitly given the equations of motion can be attained in the matrix form

$$\ddot{\boldsymbol{q}} = \boldsymbol{M}^{-1}(\boldsymbol{q})\left[\boldsymbol{\tau} - \boldsymbol{b}(\boldsymbol{q}, \dot{\boldsymbol{q}})\right] \tag{4.5}$$

Here two possibilities to obtain the explicit form of ***M*** and ***b*** are outlined in a short form:

Possibility 1:
On the base of the principle that forces/torques due to different effects can be added, it is possible to calculate the singular elements of ***M*** and ***b***. For example: if $\underline{q}$ is set to **0**, all angular velocities and acceleration are set to 0 except $\ddot{q}_1$ is set to 1 and we numerically execute (4.3), it holds: $\tau_1=M_{11}$, $\tau_2=M_{21}$, $\tau_3=M_{31}$, ...,$\tau_n=M_{n1}$ and the first column of the inertia matrix $\underline{M}$ is determined. In a similar way all elements of ***M***, ***G***, ***C*** and ***R*** can be determined. The main disadvantage is that the recursive Newton-Euler equations (4.1), (4.2) have to be executed several times and no insight in the structure of the system is possible, since the elements are numerically calculated for one point of time. For example there is no possibility to see the form of the gravity terms.

Possibility 2:
The equations (4.1), (4.2) are executed symbolically with the help of a computer language with algebraic or symbolic capabilities like Mathematica or Maple. The constant arm-parameters (parameter of length, center of gravity, mass and inertia) are treated numerically while the joint parameters comprised in $\boldsymbol{q}, \dot{\boldsymbol{q}}, \ddot{\boldsymbol{q}}$ are treated symbolically. In this way we obtain rules to calculate the torques τ_i for a single robot arm in dependence of $\boldsymbol{q}, \dot{\boldsymbol{q}}, \ddot{\boldsymbol{q}}$. The factor which is multiplicative connected to $\ddot{q}_1$ is the matrix component $M_{11}(\boldsymbol{q})$ and so on. In such a way it is possible to extract the components of ***M*** and ***b*** automatically. The main disadvantage of this method is the amount of mathematical on line operations to calculate the vector $\boldsymbol{\tau}$, when $\boldsymbol{q}, \dot{\boldsymbol{q}}, \ddot{\boldsymbol{q}}$ are given, compared to the direct numerical execution of the recursive equations (4.1), (4.2).

4.2 Model of actuators and gears

Part of the whole model of the plant are the actuators and gear train (actuator system), which are used to produce the adequate torques and motion of the joints. The actuator system consists of the servo motors, the power electronic converter including an inner current control loop and the gear train. It is not possible to neglect the dynamic behaviour of the actuator system. The actuator system has a significant influence on the dynamics of the whole system. Here only the essential attributes, necessary for control purposes, are considered.

4.2.1 Model of the actuator system

The servo systems is equipped with a current control loop. Compared with the fastness of the control loops for mechanical values like velocity and position the delay of the current control loop is negligible and the motor moment is approximately proportional to a control voltage U_S. (Fig. 4.10).

$$M_A = (C / K_{MI}) \cdot U_S = K_M \cdot U_S$$

U_S is the variable, which is manipulated by the controller. For all *n* joints we can write above relation in matrix Form:

$$\boldsymbol{M}_A = \boldsymbol{K}_M \cdot \boldsymbol{U}_S, \quad \boldsymbol{K}_M = \begin{bmatrix} C_1 / K_{MI,1} & & \mathbf{0} \\ & \ddots & \\ \mathbf{0} & & C_n / K_{MI,n} \end{bmatrix}, \boldsymbol{M}_A = \begin{bmatrix} M_{A,1} \\ \vdots \\ M_{A,n} \end{bmatrix}, \quad \boldsymbol{U}_S = \begin{bmatrix} U_{S,1} \\ \vdots \\ U_{S,n} \end{bmatrix} \tag{4.6}$$

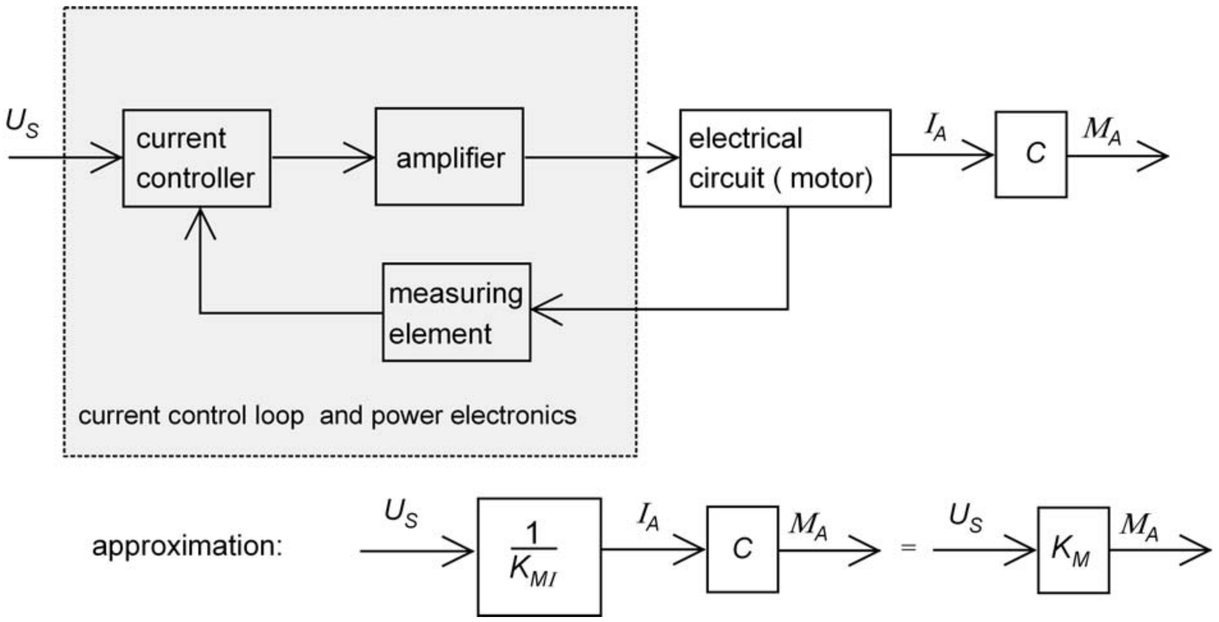


Figure 4.10. Electrical part of the motor and approximation by a proportional factor

The momentum balance related to motor shafts delivers for all joints:

$$\boldsymbol{J}_A \cdot \dot{\boldsymbol{\Omega}} = \boldsymbol{K}_M \cdot \boldsymbol{U}_S - \boldsymbol{M}_R(\boldsymbol{\Omega}) - \boldsymbol{M}_L \tag{4.7}$$

We include in vector $\boldsymbol{M}_R$ velocity dependent as well as static friction losses. $\boldsymbol{J}_A$ is a diagonal matrix of the inertia of the motor shafts.

4.2.2 Model of the gear train

In a robot arm there is a step-down gearing between motors and joints. The vector of the angular velocities $\boldsymbol{\Omega}$ is reduced to the vector $\dot{\boldsymbol{q}}$ of the angular joint velocities or translational velocities. The vector $\boldsymbol{M}_L$ of the working torques of the motors is amplified to adequate joint torques or forces condensed to vector $\boldsymbol{\tau}$. Fig. 4.11 shows a schematic diagram with the gear train, which connects the motors and the mechanics of the robot arm.

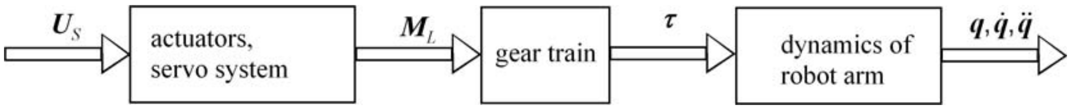


Figure 4.11 Actuator system and robot arm connected by the gear train.

When we describe the gear train, we take the following assumptions:

- the gear train is considered as mechanical stiff,
- friction losses of the gear train are neglected or attribute to $\boldsymbol{M}_R$
- the inertias of the gear wheels, gear shafts etc. are neglected or are added to the inertia matrix $\boldsymbol{J}_A$ of the motor shaft.

Under these assumptions the vector $\boldsymbol{\Omega}$ of the angular velocities of the motor shaft is mapped to the vector of the joint velocities by the constant matrix $\boldsymbol{T}_G$:

$$\dot{\boldsymbol{q}} = \boldsymbol{T}_G \cdot \boldsymbol{\Omega} \tag{4.8}$$

Is one motor directly attached to one joint, than $\boldsymbol{T}_G$ is a diagonal matrix. Often this is not the case. This means, that several actuators together drive several joints (kinematic coupling). For example a differential gear at the effector causes a non diagonal matrix $\boldsymbol{T}_G$ (Fig. 4.12).
On the other side, the vector $\boldsymbol{\tau}$ of useful torque and forces on the joints is connected with the vector of useful torque $\boldsymbol{M}_L$ at the motor side by:

$$\boldsymbol{\tau} = \boldsymbol{S}_G \cdot \boldsymbol{M}_L \tag{4.9}$$

Eq. (4.9) is based on the assumption, that the gear train has no loss. Since the power on side of the motor must then equal to the power on side of the joints, it must hold with (4.8) and (4.9):

$$\boldsymbol{M}_L^{\mathrm{T}} \cdot \boldsymbol{\Omega} = \boldsymbol{\tau}^{\mathrm{T}} \cdot \dot{\boldsymbol{q}} \tag{4.10}$$

When we replace $\boldsymbol{\tau}$ and $\boldsymbol{\Omega}$ in (4.10) with (4.8) and (4.9), we get a useful relation between $\boldsymbol{T}_G$ and $\boldsymbol{S}_G$:

$$\boldsymbol{T}_G = \left(\boldsymbol{S}_G^{-1}\right)^{\mathrm{T}} = \left(\boldsymbol{S}_G^{\mathrm{T}}\right)^{-1} \tag{4.11}$$

As example: $\boldsymbol{S}_{\mathrm{G}}$ and $\boldsymbol{T}_{\mathrm{G}}$ for the Reis-robot RV6 (see Fig. 3.7) is given:

$$\boldsymbol{T}_G = \begin{bmatrix} 1/160 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1/160 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1/160 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1/33.25 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1/15 & 0 \\ 0 & 0 & 0 & 0 & 1/15 & 1/15 \end{bmatrix}$$

$$\boldsymbol{S}_G = \begin{bmatrix} 160 & 0 & 0 & 0 & 0 & 0 \\ 0 & 160 & 0 & 0 & 0 & 0 \\ 0 & 0 & 160 & 0 & 0 & 0 \\ 0 & 0 & 0 & 33.25 & 0 & 0 \\ 0 & 0 & 0 & 0 & 15 & -15 \\ 0 & 0 & 0 & 0 & 0 & 15 \end{bmatrix}$$

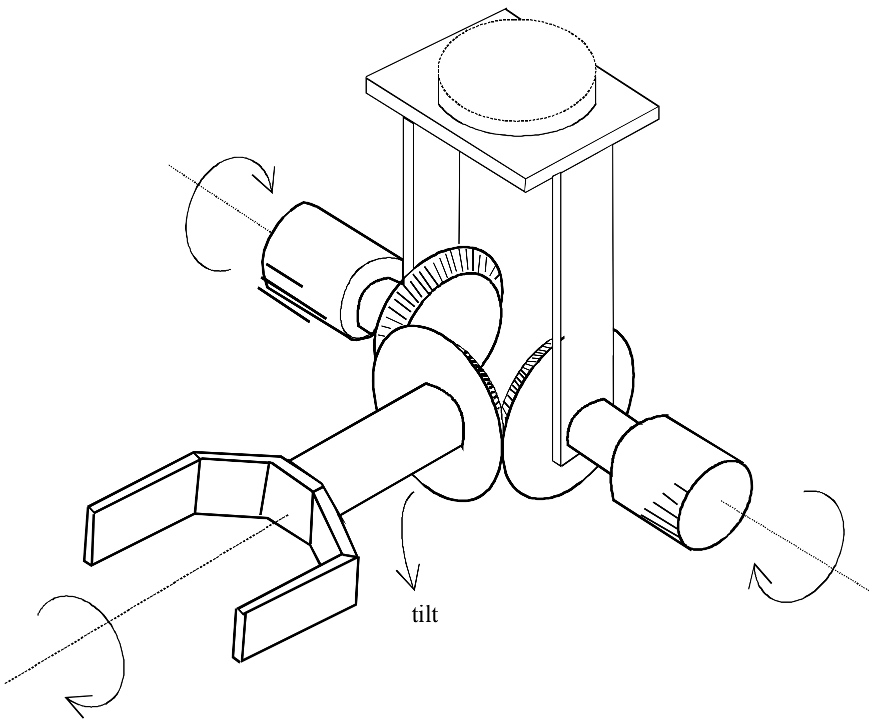


Fig. 4.12 Kinematical coupling by a differential gear

4.3 Dynamics of the whole electromechanics

At this point we are able to set up the dynamics of the whole plant. As the gear train couples the motors and the robot arm, the equations (4.8), (4.9) couple the matrix differential equation (4.7) on the motor side with the matrix differential equation (4.4) on the joint side.
Solving (4.7) for $\boldsymbol{M}_L$ and substituting in (4.9) gives

$$\boldsymbol{\tau} = \boldsymbol{S}_G \cdot [\boldsymbol{K}_M \cdot \boldsymbol{U}_S - \boldsymbol{M}_R(\boldsymbol{\Omega}) - \boldsymbol{J}_A \cdot \dot{\boldsymbol{\Omega}}] \tag{4.12}$$

The input vector for the whole dynamics is $\boldsymbol{U}_S$ not $\boldsymbol{\tau}$. Hence $\boldsymbol{\tau}$ must be substituted in (4.12) by the right side of (4.4a):

$$\boldsymbol{M}(\boldsymbol{q}) \cdot \ddot{\boldsymbol{q}} + \boldsymbol{b}(\boldsymbol{q},\dot{\boldsymbol{q}}) = \boldsymbol{S}_G \cdot \left[\boldsymbol{K}_M \cdot \boldsymbol{U}_S - \boldsymbol{M}_R(\boldsymbol{\Omega}) - \boldsymbol{J}_A \cdot \dot{\boldsymbol{\Omega}}\right] \tag{4.13}$$

Is $\dot{\boldsymbol{\Omega}}$ substituted by $\boldsymbol{T}_G^{-1} \cdot \ddot{\boldsymbol{q}}$ due to (4.8) and is the equation solved for $\boldsymbol{U}_S$, we obtain

$$\boldsymbol{U}_S = \boldsymbol{K}_M^{-1} \cdot [\boldsymbol{S}_G^{-1} \cdot \boldsymbol{M}(\boldsymbol{q}) + \boldsymbol{J}_A \cdot \boldsymbol{T}_G^{-1}] \cdot \ddot{\boldsymbol{q}} + \boldsymbol{K}_M^{-1} \cdot [\boldsymbol{S}_G^{-1} \cdot \boldsymbol{b}(\boldsymbol{q},\dot{\boldsymbol{q}}) + \boldsymbol{M}_R(\dot{\boldsymbol{q}})] \tag{4.14}$$

or

$$\boldsymbol{U}_S = \boldsymbol{M}^*(\boldsymbol{q}) \cdot \ddot{\boldsymbol{q}} + \boldsymbol{b}^*(\boldsymbol{q},\dot{\boldsymbol{q}}) \tag{4.15a}$$

with

$$\begin{aligned} \boldsymbol{M}^*(\boldsymbol{q}) &= \boldsymbol{K}_M^{-1} \cdot [\boldsymbol{S}_G^{-1} \cdot \boldsymbol{M}(\boldsymbol{q}) + \boldsymbol{J}_A \cdot \boldsymbol{T}_G^{-1}], \\ \boldsymbol{b}^*(\boldsymbol{q},\dot{\boldsymbol{q}}) &= \boldsymbol{K}_M^{-1} \cdot [\boldsymbol{S}_G^{-1} \cdot \boldsymbol{b}(\boldsymbol{q},\dot{\boldsymbol{q}}) + \boldsymbol{M}_R(\dot{\boldsymbol{q}})] \end{aligned} \tag{4.15b}$$

Equations (4.15) describe the behaviour of the whole controlled system (plant). (4.15) has the form of an inverse model. If we know the vectors $\boldsymbol{q}, \dot{\boldsymbol{q}}, \ddot{\boldsymbol{q}}$ as a function of time, $\boldsymbol{U}_S$, which causes this motion, can be calculated. Notice that we can write (4.14) with known $\boldsymbol{\tau}$ from (4.4a) as

$$\boldsymbol{U}_S = \boldsymbol{K}_M^{-1} \cdot \boldsymbol{S}_G^{-1} \cdot \boldsymbol{\tau} + \boldsymbol{K}_M^{-1} \cdot [\boldsymbol{J}_A \cdot \boldsymbol{T}_G^{-1} \cdot \ddot{\boldsymbol{q}} + \boldsymbol{M}_R(\dot{\boldsymbol{q}})] \tag{4.16}$$

Hence the motors are mounted on the links of the robot arm, the masses of the motors and gears must be considered by the declaration of the masses m_i and the inertia tensors $\boldsymbol{I}_{SP,i}$.
The model or equation of motion is achieved by solving (4.15a) for the acceleration vector:

$$\ddot{\boldsymbol{q}} = \left[\boldsymbol{M}^*(\boldsymbol{q})\right]^{-1} \cdot \left[\boldsymbol{U}_S - \boldsymbol{b}^*(\boldsymbol{q},\dot{\boldsymbol{q}})\right]. \tag{4.17}$$

In Fig. 4.13 the plant is depicted in a schematic form.

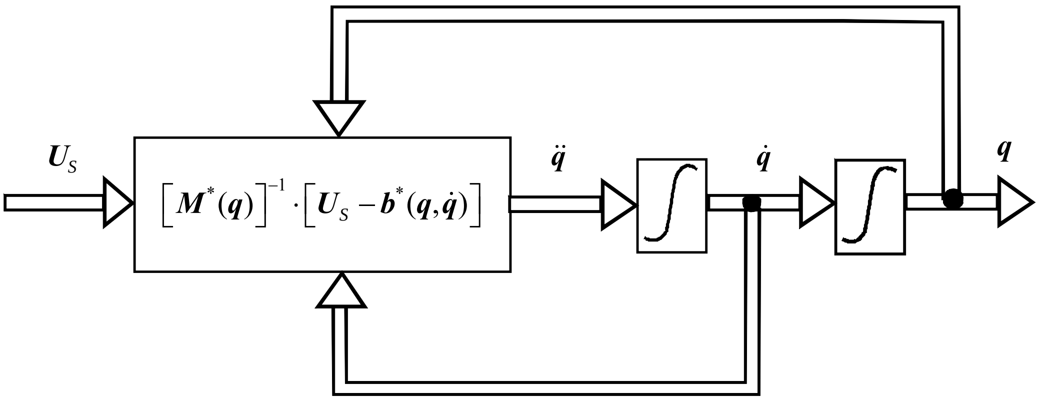

$\boldsymbol{U}_S$
$\left[\boldsymbol{M}^*(\boldsymbol{q})\right]^{-1} \cdot \left[\boldsymbol{U}_S - \boldsymbol{b}^*(\boldsymbol{q},\dot{\boldsymbol{q}})\right]$
$\ddot{\boldsymbol{q}}$
$\dot{\boldsymbol{q}}$
$\boldsymbol{q}$

Fig. 4.13 The whole plant including robot arm and actuator system.

We have now described the whole model of the robot system including arm dynamics, gear train and actuator system (Fig. 4.14).

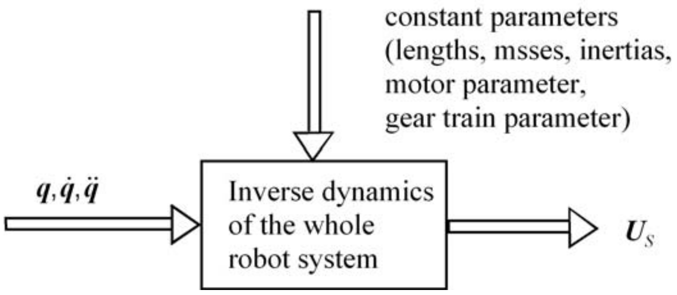


Fig. 4.14 Scheme of the Inverse model

In our case the task of the Inverse model and of the equation of motion can be described as follows:

Inverse Model:

The voltages applied on the servos required to cause a given or desired motion have to be evaluated.

Equation of Motion:

The motion which arises from the acting voltages (input signals of the motor servos) has to be evaluated

5. Model-based Control

In chapter 1 we have established, that the controlled system robot is a multi-input / multi-output system. Now we have described in chapter 4 the robot system including actuators and gear train by a nonlinear matrix equation. The equation of motion (3.17) and Fig. 3.13 describe how the robot arm moves under the effect of the vector $\underline{U}_S$ of control voltages:

$$\ddot{\boldsymbol{q}} = \left[\boldsymbol{M}^*(\boldsymbol{q})\right]^{-1} \cdot \left[\boldsymbol{U}_S - \boldsymbol{b}^*(\boldsymbol{q},\dot{\boldsymbol{q}})\right] \tag{5.1}$$

The control problem is now to find appropriate voltages that the robot arm track a desired position trajectory. In this chapter the control problem is treated considering the nonlinear couplings respectively time variant parameters, if we describe the control as single axis problem. In the first two sections two methods of model based control are considered. All methods of model based control (also named *computed torque method*) use the calculation of the inverse model to decouple and to get linear replacement of dynamics of the robot arm. In chapter 5.3 the Model Reference Adaption Control (MRAC) is presented. Here control parameters of a single axis controller are changed on line due to time varying parameters of the plant.

5.1 Control law to achieve a desired transfer function

Here we use two ideas to design the controller: the reference model method and the computed torque method in a something modified manner.

We develop the control algorithm in two steps:

1. Choosing a linear dynamic input-output behaviour as reference model between the desired motion $\underline{q}_d$ and controlled motion $\underline{q}$.

2. Derivation of the control law to achieve the desired input-output behaviour.

5.1.1 Reference model

Here we define a desired linear dynamic model between the vector of desired joint variables $\underline{q}_d$ and the vector of real joint variables $\underline{q}$. We choose a general second order linear model for each joint i:

$$\ddot{q}_i + a_{1,i} \cdot \dot{q}_i + a_{0,i}\, q_i = b_{2,i} \cdot \ddot{q}_{d,i} + b_{1,i} \cdot \dot{q}_{d,i} + b_{0,i} \cdot q_{d,i} \tag{5.2}$$

The parameters $a_{1,i}$, $a_{0,i}$, $b_{0,i}$, $b_{1,i}$, $b_{2,i}$ are constant parameters, which are chosen by the application engineer. From there we can transform (4.2) in the Laplace domain:

$$Q_i(s) = \frac{b_{0i} + b_{1i}s + b_{2i}s^2}{a_{0i} + a_{1i}s + s^2} \cdot Q_{d,i}(s) \tag{5.3}$$

The input-output behaviour of each joint i can be chosen independently from the other joints. That means we choose a decoupled input-output behaviour. Since the output value q_i has to hold the desired value $q_{d,i}$ in the steady state case, $b_{0,i} = a_{0,i}$ must be valid.
P-T$_2$ behaviour with damping constant d_i and time constant T_i is a suitable desired input-output behaviour. This will be achieved for $b_{1,i} = b_{2,i} = 0$, $a_{0,i} = b_{0,i} = 1/T_i^2$, $a_{1,i} = 2 \cdot d_i / T_i$. In this case we get for the differential equation (5.2):

$$\ddot{q}_i + \frac{2 \cdot d_i}{T_i} \cdot \dot{q}_i + \frac{1}{T_i^2} \cdot q_i = \frac{1}{T_i^2} \cdot q_{d,i} \tag{5.4}$$

and for the transfer function (5.3)

$$Q_i(s) = \frac{1}{1 + 2d_i T_i \cdot s + T_i^2 \cdot s^2} \cdot Q_{d,i}(s) \tag{5.5}$$

It is also possible to select P-T$_1$ behaviour with time constant T_i if we choose $b_{1,i} = 1/T_i$, $b_{2,i} = 0$, $a_{0,i} = b_{0,i} = 1/T_i^2$, $a_{1,i} = 2/T_i$. The differential equation (5.2) and the transfer function (5.3) become

$$\ddot{q}_i + \frac{2}{T_i} \cdot \dot{q}_i + \frac{1}{T_i^2} \cdot q_i = \frac{1}{T_i} \cdot \dot{q}_{d,i} + \frac{1}{T_i^2} \cdot q_{d,i} \tag{5.6}$$

$$Q_i(s) = \frac{\frac{1}{T_i^2} + \frac{1}{T_i} \cdot s}{\frac{1}{T_i^2} + \frac{2}{T_i} \cdot s + s^2} \cdot Q_{d,i}(s) = \frac{1}{1 + T_i \cdot s} \cdot Q_{d,i}(s) \tag{5.7}$$

The desired input-output behaviour (5.2) for all joints can be written in matrix form:

$$\ddot{\boldsymbol{q}} + \boldsymbol{A}_0\, \boldsymbol{q} + \boldsymbol{A}_1\, \dot{\boldsymbol{q}} = \boldsymbol{B}_2\, \ddot{\boldsymbol{q}}_d + \boldsymbol{B}_1\, \dot{\boldsymbol{q}}_d + \boldsymbol{B}_0\, \boldsymbol{q}_d \tag{5.8}$$

The matrices in (4.8) are constant diagonal matrices. For P-T$_2$ behaviour it holds

$$\boldsymbol{A}_0 = \boldsymbol{B}_0 = \begin{bmatrix} \frac{1}{T_1^2} & & \underline{0} \\ & \ddots & \\ \underline{0} & & \frac{1}{T_n^2} \end{bmatrix}, \quad \boldsymbol{A}_1 = \begin{bmatrix} \frac{2 \cdot d_1}{T_1} & & \underline{0} \\ & \ddots & \\ \underline{0} & & \frac{2 \cdot d_n}{T_n} \end{bmatrix}, \quad \boldsymbol{B}_1 = \boldsymbol{B}_2 = \underline{0}. \tag{5.9}$$

PT$_1$ dynamic is given by the matrices

$$\boldsymbol{A}_0 = \boldsymbol{B}_0 = \begin{bmatrix} \frac{1}{T_1^2} & & \underline{0} \\ & \ddots & \\ \underline{0} & & \frac{1}{T_n^2} \end{bmatrix}, \quad \boldsymbol{A}_1 = \begin{bmatrix} \frac{2}{T_1} & & \underline{0} \\ & \ddots & \\ \underline{0} & & \frac{2}{T_n} \end{bmatrix}, \quad \boldsymbol{B}_1 = \begin{bmatrix} \frac{1}{T_1} & & \underline{0} \\ & \ddots & \\ \underline{0} & & \frac{1}{T_n} \end{bmatrix}, \quad \boldsymbol{B}_2 = \underline{0} \tag{5.10}$$

For further handling (5.8) is solved for $\ddot{\boldsymbol{q}}$ and the right side is abbreviated by $\boldsymbol{r}_0$:

$$\ddot{\boldsymbol{q}} = \boldsymbol{B}_2\, \ddot{\boldsymbol{q}}_d + \boldsymbol{B}_1\, \dot{\boldsymbol{q}}_d + \boldsymbol{B}_0\, \boldsymbol{q}_d - \boldsymbol{A}_0\, \boldsymbol{q} - \boldsymbol{A}_1\, \dot{\boldsymbol{q}} = \boldsymbol{r}_0\left(\ddot{\boldsymbol{q}}_d, \dot{\boldsymbol{q}}_d, \boldsymbol{q}_d, \boldsymbol{q}, \dot{\boldsymbol{q}}\right) \tag{5.11}$$

5.1.2 Control law for achieving the desired input-output behaviour

The question is now, how $\boldsymbol{U}_S$, the output vector of the controller, must be calculated to achieve the reference dynamic (5.8), respectively (5.11). The control law can be derived in an easy manner, if we compare (5.11) and the equation of motion (5.1). Eq. (5.1) describes the motion of the robot arm under the effect of the vector $\boldsymbol{U}_S$. If this motion has to obey the desired differential equation (5.11), the right side of the desired behaviour (5.11) and the right side of the real behaviour (5.1) must be equal:

$$\boldsymbol{r}_0(\ddot{\boldsymbol{q}}_d, \dot{\boldsymbol{q}}_d, \boldsymbol{q}_d, \boldsymbol{q}, \dot{\boldsymbol{q}}) = \left[\boldsymbol{M}^*(\boldsymbol{q})\right]^{-1} \cdot \left[\boldsymbol{U}_S - \boldsymbol{b}^*(\boldsymbol{q}, \dot{\boldsymbol{q}})\right] \tag{5.12}$$

Now we can solve for $\boldsymbol{U}_S$ to achieve the control law:

$$\boldsymbol{U}_S = \boldsymbol{M}^*(\boldsymbol{q}) \cdot \boldsymbol{r}_0 + \boldsymbol{b}^*(\boldsymbol{q}, \dot{\boldsymbol{q}}) \tag{5.13}$$

By Comparing (5.13) with (4.15a) we see, that the control law is identical with the inverse model according to (4.15a). But in (5.13) there isn't used the vector $\ddot{\boldsymbol{q}}$ of measured accelerations but the vector $\boldsymbol{r}_0$ which is calculated in dependence of the desired and measured values of the joint variables. (4.13) is a control law. In general there are some deviations between the real matrix $\boldsymbol{M}^*(\boldsymbol{q})$ and real vector $\boldsymbol{b}^*(\boldsymbol{q}, \dot{\boldsymbol{q}})$ and the calculated one, since parameters are not completely available or a reduced model is used. That's why we write instead of (5.13)

$$\boldsymbol{U}_S = \tilde{\boldsymbol{M}}^*(\boldsymbol{q}) \cdot \boldsymbol{r}_0 + \tilde{\boldsymbol{b}}^*(\boldsymbol{q}, \dot{\boldsymbol{q}}) . \tag{5.14}$$

The sign „ ~ „ over $\boldsymbol{M}$ and $\boldsymbol{b}$ indicates, that the values do not completely agree with $\boldsymbol{M}^*$ and $\boldsymbol{b}^*$.

If we have a look on our method to calculate the vector $\boldsymbol{U}_S$ and that means managing the control design, we see, that we get the control rules without the usual control design methods. The vector $\boldsymbol{U}_S$ of the manipulated variables can be achieved by the reference model of the input-output behaviour and the model (equations of motion) of the robot system (Fig. 4.1).

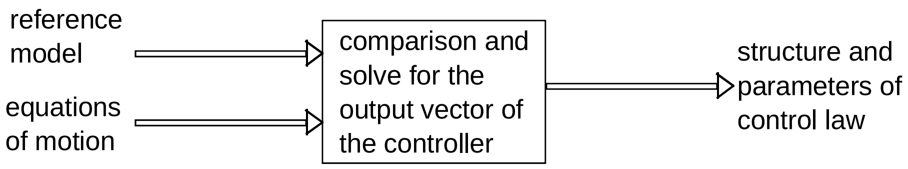

reference model
equations of motion
comparison and solve for the output vector of the controller
structure and parameters of control law

Figure 5.1. Principle of the model based control design

Fig. 5.2 shows the general structure of the control system, while Fig. 4.3 shows the block diagram if PT_2-dynamic is used for the reference model (see equation 5.9).

The model based control has the following advantages:

- The essential design of the control system bases on the choice of a reference model, which can be adapted to the application.
- The nonlinear system will be (approximately) decoupled and linearized by (5.13). The control quality can be predicted.
- For all joint motions the same dynamic can be chosen. This improves the tracking quality of a path.

Disadvantages:

- The control quality depends on the quality of the robot model. The effort to identify the parameters of a model may be high.
- The real time effort to calculate the vector $\underline{U}_S$ of the manipulated variables is very high, since the inverse model must be calculated on line. A powerful hardware is necessary.
- If we divide the control algorithm into controller and inverse model, the controller (5.11) $\boldsymbol{r}_0 = \boldsymbol{B}_2\,\ddot{\boldsymbol{q}}_d + \boldsymbol{B}_1\,\dot{\boldsymbol{q}}_d + \boldsymbol{B}_0\,\boldsymbol{q}_d - \boldsymbol{A}_0\,\boldsymbol{q} - \boldsymbol{A}_1\,\dot{\boldsymbol{q}}$ has no I-term. For example a steady state error occurs, if the robot carries an unknown load.

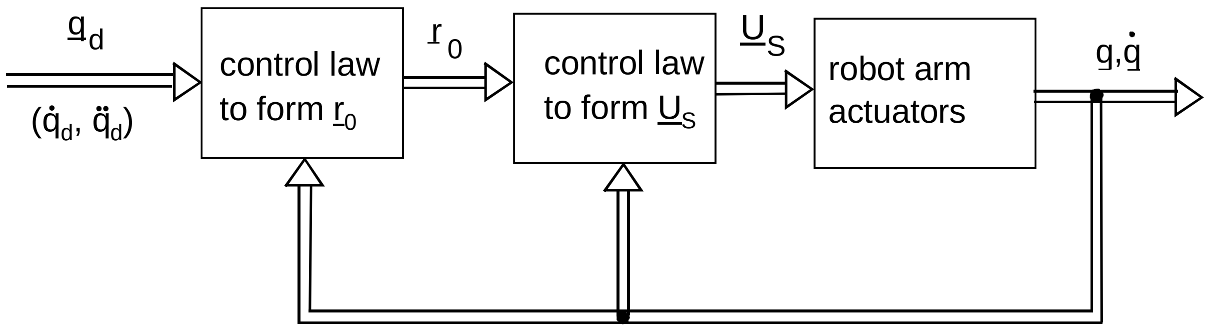

$\underline{q}_d$
$(\dot{\underline{q}}_d, \ddot{\underline{q}}_d)$
control law to form $\underline{r}_0$
$\underline{r}_0$
control law to form $\underline{U}_S$
$\underline{U}_S$
robot arm actuators
$\underline{q}, \dot{\underline{q}}$

Figure 5.2. General structure of the control system with the model based controller.

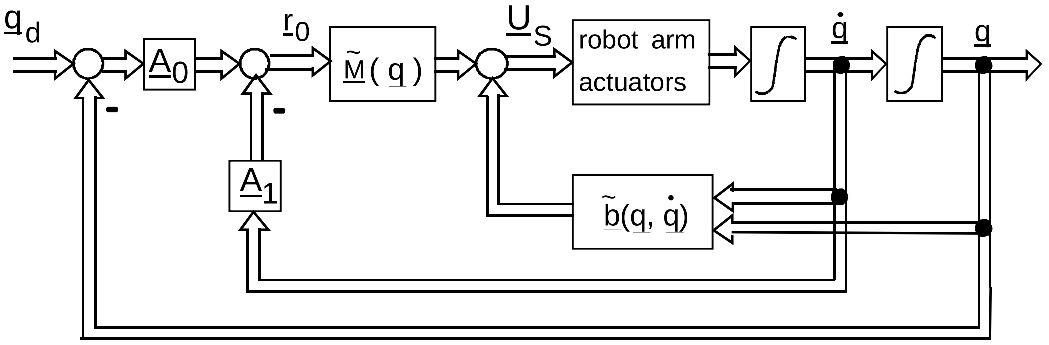


Figure 5.3. Structure of the model-based control system with PT$_2$ reference model.

5.2 Model based cascade control

This new method differs from the method in chapter 5.1 only in the calculation of the control vector $\boldsymbol{r}_0$. The objective is to avoid the last disadvantage mentioned above. First we don't consider how $\boldsymbol{r}_0$ can be calculated, but we use (5.13) respectively (5.14) as part of the control law in the same way as in chapter 4.1. Furthermore we assume that our inverse model is exact, so that (5.13) and (5.14) are identical. Now we use (5.13) in the equation of motion in (5.1)

$$\ddot{\boldsymbol{q}} = \left[\boldsymbol{M}^*(\boldsymbol{q})\right]^{-1} \cdot \left[\boldsymbol{M}^*(\boldsymbol{q}) \cdot \boldsymbol{r}_0 + \boldsymbol{b}^*(\boldsymbol{q},\dot{\boldsymbol{q}}) - \boldsymbol{b}^*(\boldsymbol{q},\dot{\boldsymbol{q}})\right]$$

and get the virtual model

$$\ddot{\boldsymbol{q}} = \boldsymbol{r}_0 \tag{5.15}$$

The task is now, to calculate $\boldsymbol{r}_0$ in such a manner, that the control task is satisfied. In the literature often the vector of control error $\boldsymbol{e}$=$\boldsymbol{q}_d$-$\boldsymbol{q}$ and its first two deviations are used to calculate $\boldsymbol{r}_0$ to

$$\boldsymbol{r}_0 = \boldsymbol{K}_P \cdot \boldsymbol{e} + \boldsymbol{K}_V \cdot \dot{\boldsymbol{e}} + \ddot{\boldsymbol{e}} \ . \tag{5.16}$$

$\boldsymbol{K}_P$ and $\boldsymbol{K}_V$ are diagonal matrices.[1] But this control law does not guarantees steady-state accuracy and the simple and well known transfer behaviour like PT$_2$- or PT$_1$-characteristic will be not achieved. Here we will follow up the idea to realize a cascade control concept with REDUS velocity control and position control (see appendix C). In (5.16) all joints are

[1] See for example: J.J. Craig: Introduction to Robotics. Addison-Wesley, 3. ed. 2005.

decoupled. Between the input $r_{0,i}$ and the output q_i of one joint there is a double I-term(Fig. 5.4), the plant for the inner velocity control loop can be described by a single I-term. To control the inner velocity control loop the REDUS controller will be used (see Appendix C, here the

input signal is $r_{0,i}$).

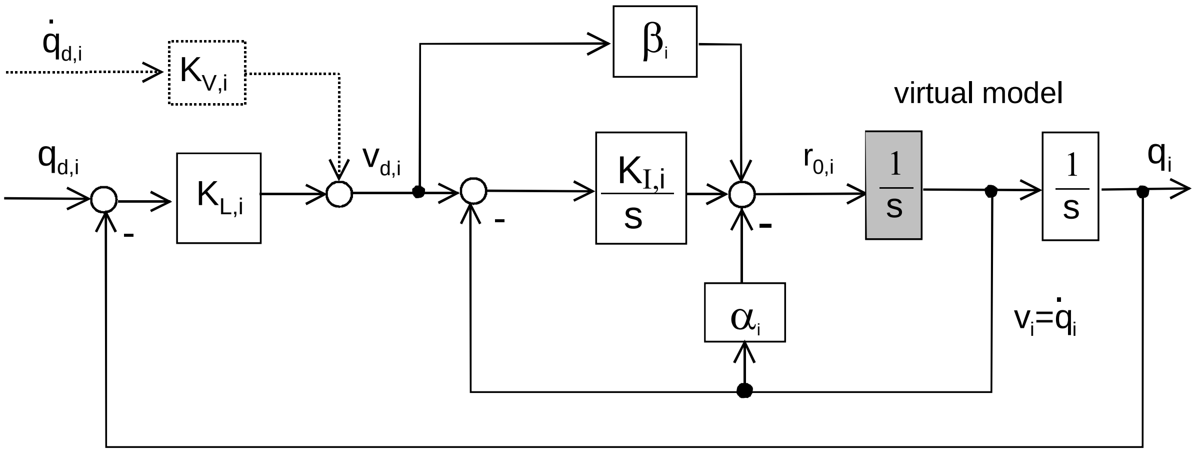


Figure 5.4 Cascade control of the virtual model for the i-th joint.

The control design is to do for all joints in the same way, though the index *i* will be omitted in the following. Due to equation (A1) and figure A1 it holds for all joints:

$$a_0 = 0 \ , \ a_1 = 1 \ , \ a_2 = 0 \ . \tag{5.17}$$

If a PT$_2$ transfer function with time constant T_R and damping constant d_R for each joint is desired, the design rules become due to (A4):

$$\beta = 0, \quad K_I = \frac{1}{T_R^2}, \quad \alpha = \frac{2 \cdot d_R}{T_R} \ . \tag{5.18}$$

In the case of this simple (virtual) plant it is possible to set up PT$_1$ behaviour between v_d and v with the time constant T_V:

$$V(s) = \frac{1}{1 + T_V \cdot s} \cdot V_d(s) \tag{5.19}$$

Due to Fig. 5.4 the transfer function between v_d and v can be calculated to

$$V(s) = \frac{K_I + \beta \cdot s}{K_I + \alpha \cdot s + s^2} \cdot V_d(s) \ . \tag{5.20}$$

This transfer function has to satisfy (5.19). This leads to two equations for the control parameters:

$$T_V \cdot K_I + \beta = \alpha, \quad \beta \cdot T_V = 1 \tag{5.21}$$

There are two equations in (5.21) for the three control parameters α, β and K_I. While β is determined by the choice of T_V, there is a linear dependence between K_I and α. We get the rules

$$\beta = \frac{1}{T_V}\ ,\ \alpha = V_R \cdot \beta,\ K_\mathrm{I} = \frac{1}{T_V^2}\cdot (V_R - 1)\ ,\ V_R > 1 \tag{5.22}$$

The parameter V_R must be larger than 1, but it is otherwise free (e.g. $V_R = 2$).

The position control loop for each joint can be designed with $G_V(s) = \frac{V(s)}{R_0(s)}$ as PT$_2$ or PT$_1$-term for the secondary control loop. State of the art is to use a P-term as position controller.

The general structure of the control system is identical with Fig. 5.2. Since the control system in 5.4 has I-Terms, weighted with the coefficients K_I, the last disadvantage mentioned for the structure in 5.1 is no longer valid. Furthermore the cascade control system to control joints of robots and machines is often used in practice.

5.3 Adaptive control with the MRAC-method

If the parameters of the plant vary with time and the parameters of the controller are adapted in a suitable way, we speak of adaptive control. Fig. 5.5 shows the general structure of an adaptive control system. On base of the desired value *w*, the controlled variable *x* and the manipulated variable *u,* the adaptive law adapts the parameters of the controller to achieve a performance index.

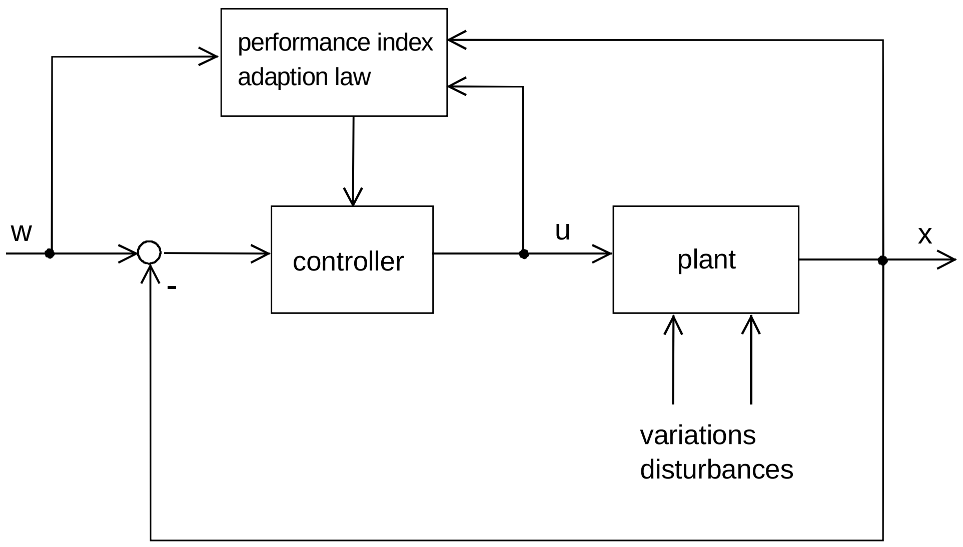


Figure 5.5 General structure of an adaptive control system.

One way to realize a performance index is to define a reference model and to compare the calculated answer of the reference model x_R with the output variable *x*. In the literature this method is named Model-Referenced Adaptive Control (MRAC, Fig. 5.6).

We will apply the idea of MRAC to the control problem of the robot. A single axis control system is assumed. The components of the vector ***b**** are considered as disturbances, only the corresponding diagonal element of ***M****(***q***) is viewed as part of the plant (Fig. 5.7). The position dependence of the diagonal element $M_{ii}(\boldsymbol{q})$ is considered as time-dependence $M_{ii}^*(t)$. Velocity dependent friction losses may also be considered.

advantage: masses of loads are indirectly considered too, since they are altering $M_{ii}^*(\boldsymbol{q})$
disadvantage: the nonlinear coupling by $b_i^*(\boldsymbol{q},\dot{\boldsymbol{q}})$ and $M_{ik}^* \cdot \ddot{q}_k$ are not considered.

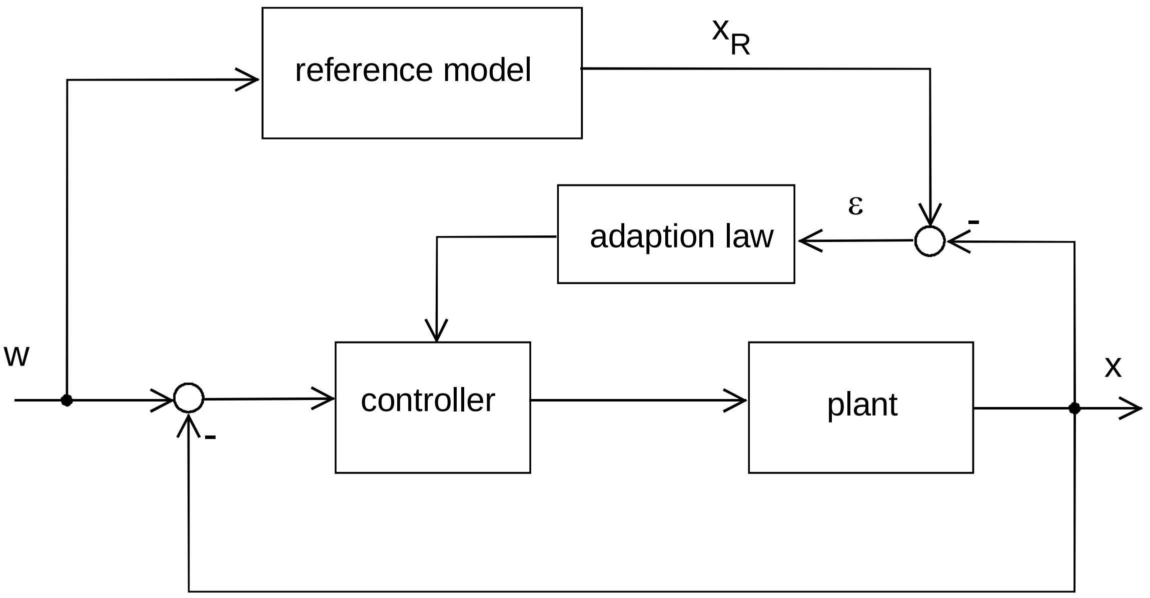


Figure 5.6. Structure of the Model-Referenced Adaptive Control.

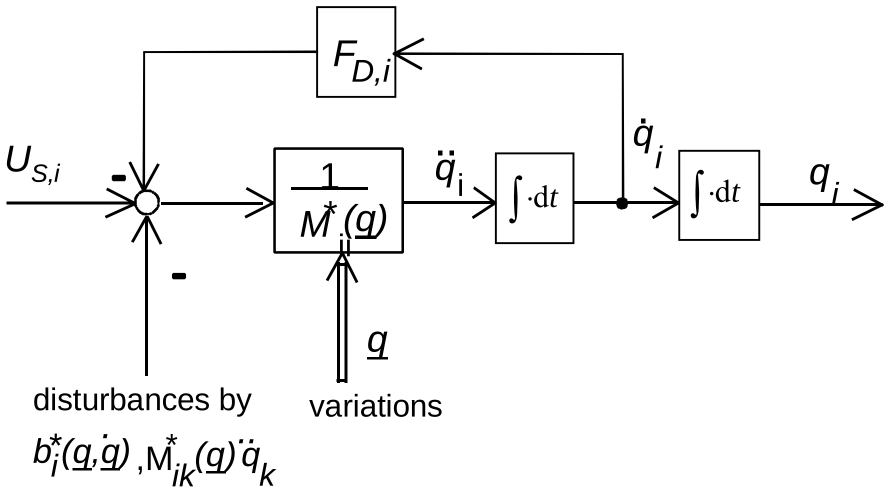


Figure 5.7 Time dependent plant of a single joint

Since the structure is the same for each joint the index *i* is not written in the following. P-T$_2$ dynamic is chosen as reference model, which describes the desired transfer behaviour between q_d and q:

$$a \cdot \ddot{q}_R + b \cdot \dot{q}_R + q_R = q_d \quad \circ\!-\!-\!-\!-\!-\!\bullet \quad Q_R(s) = \frac{1}{1 + bs + as^2} \cdot Q_d(s) \tag{5.23}$$

In Fig. 5.8 the control loop with two time dependent control parameters $K_P(t)$ and $K_V(t)$ is depicted. $K_P(t)$ and $K_V(t)$ are altered by the adaption rule on the base of the deviation $\varepsilon = q_R - q$. From Fig. 5.7 the differential equation

$$\frac{M^*(t)}{K_P(t)} \cdot \ddot{q} + \frac{K_V(t) + F_D}{K_P(t)} \cdot \dot{q}(t) + q = q_d$$

can be set up. It is desired that (5.24) is nearly identical with (5.23):

$$\alpha(t) = \frac{M^*(t)}{K_P(t)} \approx a\,, \quad \beta(t) = \frac{K_V(t) + F_D^*}{K_P(t)} \approx b$$

In this case it holds $x \approx x_R$. The adaption rule has to change K_P and K_V on the base of ε (see fig 5.6), that equation (5.25) is valid. To solve this problem, optimization methods are used (for further studies see Dubowsky,S. and Des Forges, D.T.: *The application of Model Referenced Adaptive Control to Robotic Manipulators*. Journ. of Dyn. Systems, Meas. and Control 101(1979), p. 193-200. and Süss, U. Weber, W.: *Untersuchung und Realisierung einer adaptiven Gelenkregelung nach dem Referenzmodellkonzept*, VDI Berichte Nr. 598, 1986, in German).

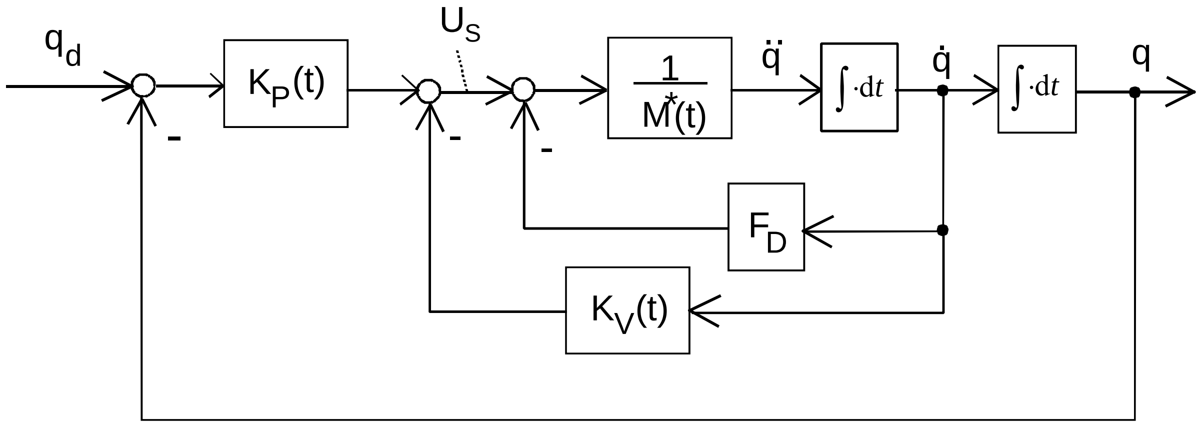


Figure 5.8 Scheme of the control loop for MRAC

Notice that the assumption must take place, that $M^*(t)$ varies slowly compared with the other time constants of the whole system. This causes the main problem of the most adaptive control schemes: Stability and damping in case of fast changes of plant parameters. Experimental tests show that the MRAC scheme works very well, but if the load is dropped then weakly damped oscillations appear.

- Additional concepts for adaptive control of robot arms can be found in

  Siciliano, B., **Sciavicco**, L., **Villani**, L., **Oriolo**, G.: Robotics- Modelling, Planning and Control Springer, London, 2009, ISBN: 978-1-84628-641-4
  and in
  J. Craig: *Adaptive Control of Mechanical Manipulators*. Addison-Wesley, Reading(Mass.), 2004.

Appendix A

Model of the dynamics of a two joint robot by the method of free body diagram

As an introductive example we consider a robot with a prismatic joint (joint 1) and a revolute joint (joint 2), see Fig. A1. His robot can also be considered as an inverse pendulum.
Here we will use the free body diagram, the law of conservation of linear momentum and the law of conservation of angular momentum to set up the model. On the base of this two body model we will discuss important aspects of the dynamics of open chain robots.

Fig. A1 shows the robot with prismatic joint (q_1) and revolute joint (q_2). Link 1 has the mass m_1 and its centre of mass is born on first axis. The centre of mass of link 2 has the distance l_{Sp2} to the joint axis 2.

.

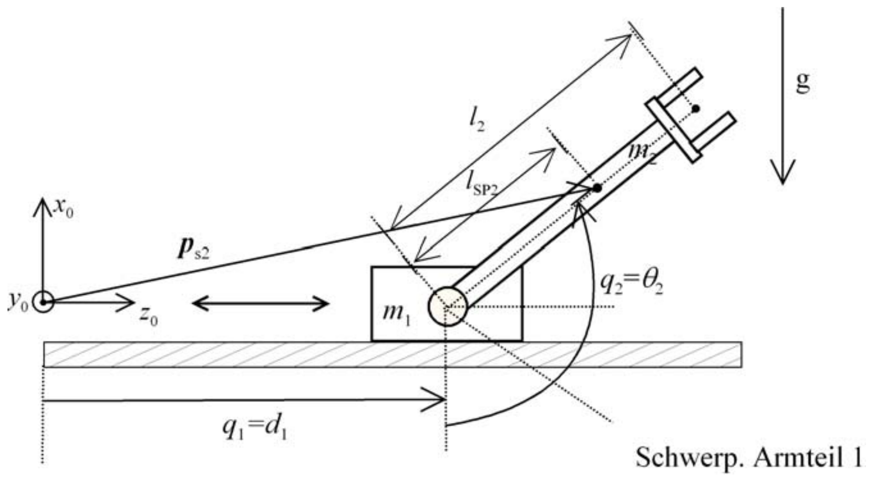


Fig. A1. TR-robot with prismatic joint (q_1) and revolute joint (q_2)

First the position vector $\boldsymbol{p}_{s1}, \boldsymbol{p}_{s2}$ and its first derivations due time of link 1 respectively link 2 related to the non moving coordinate system K_0 is established:

$$\boldsymbol{p}_{s1} = \begin{bmatrix} 0 \\ 0 \\ q_1 \end{bmatrix}, \dot{\boldsymbol{p}}_{s1} = \begin{bmatrix} 0 \\ 0 \\ \dot{q}_1 \end{bmatrix}, \ddot{\boldsymbol{p}}_{s1} = \begin{bmatrix} 0 \\ 0 \\ \ddot{q}_1 \end{bmatrix} \tag{A1}$$

The position vector to the centre of link 2 depends on q_1 and q_2

$$\boldsymbol{p}_{s2} = \begin{pmatrix} l_{SP2} \cdot \sin(q_2 - 90°) \\ 0 \\ q_1 + l_{SP2} \cdot \cos(q_2 - 90°) \end{pmatrix} = \begin{pmatrix} -l_{SP2} \cdot \cos(q_2) \\ 0 \\ q_1 + l_{SP2} \cdot \sin(q_2) \end{pmatrix}$$

$$\dot{\boldsymbol{p}}_{s2} = \begin{pmatrix} l_{SP2} \cdot \sin(q_2) \cdot \dot{q}_2 \\ 0 \\ \dot{q}_1 + l_{SP2} \cdot \cos(q_2) \cdot \dot{q}_2 \end{pmatrix}, \quad \ddot{\boldsymbol{p}}_{s2} = \begin{pmatrix} l_{SP2} \cdot \cos(q_2) \cdot \dot{q}_2^2 + l_{SP2} \cdot \sin(q_2) \cdot \ddot{q}_2 \\ 0 \\ \ddot{q}_1 - l_{SP2} \cdot \sin(q_2) \cdot \dot{q}_2^2 + l_{SP2} \cdot \cos(q_2) \cdot \ddot{q}_2 \end{pmatrix} \tag{A2}$$

Now we evaluate the vector of angular motion $\boldsymbol{\omega}$ and the vector of angular acceleration $\dot{\boldsymbol{\omega}}$ for both links.
Link 1 has a restricted guidance in q_1 direction and therefore no rotation of this body take place:

$$\boldsymbol{\omega}_1 = \dot{\boldsymbol{\omega}}_1 = 0 \tag{A3}$$

The angular velocity of link 2 depends exclusively on the alteration of q_2. The rotation is in direction of y_0 :

$$\boldsymbol{\omega} = \begin{bmatrix} 0 \\ \dot{q}_2 \\ 0 \end{bmatrix}, \quad \dot{\boldsymbol{\omega}} = \begin{bmatrix} 0 \\ \ddot{q}_2 \\ 0 \end{bmatrix} \tag{A4}$$

We apply the free body diagram to link 2 (Fig. A2) and we establish the law of conservation of linear momentum and the law of conservation of angular momentum. Vector $\boldsymbol{f}_1$ describes the force which is caused by link 1 and act to link 2.

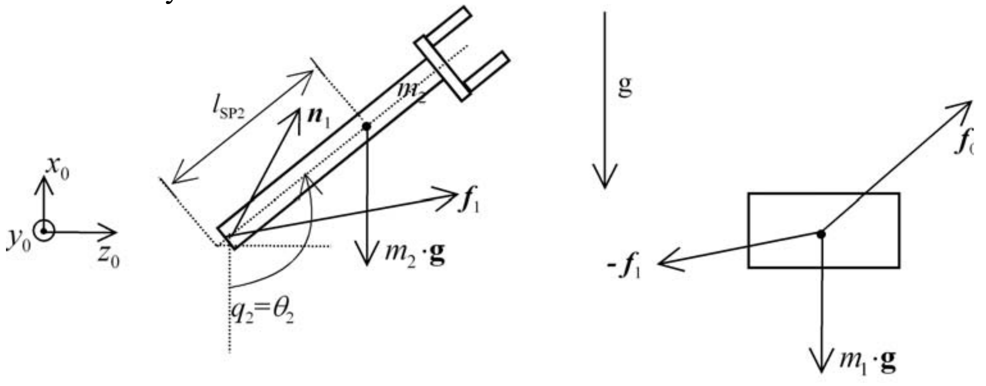


Fig. A2. Free body diagram of link 2(left) and link 1 (right)

For the balance of forces in the centre of mass we get:

$$m_2 \cdot \ddot{\boldsymbol{p}}_{s2} = m_2 \cdot \begin{pmatrix} -g \\ 0 \\ 0 \end{pmatrix} + \boldsymbol{f}_1 \rightarrow \boldsymbol{f}_1 = m_2 \cdot \begin{pmatrix} g + l_{SP2} \cdot \cos(q_2) \cdot \dot{q}_2^2 + l_{SP2} \cdot \sin(q_2) \cdot \ddot{q}_2 \\ 0 \\ \ddot{q}_1 - l_{SP2} \cdot \sin(q_2) \cdot \dot{q}_2^2 + l_{SP2} \cdot \cos(q_2) \cdot \ddot{q}_2 \end{pmatrix} \tag{A5}$$

For the balance of momentum we need in fact the inertia tensor related to the centre of mass. But link 2 can only rotate around an axis perpendicular to the $x_0 - z_0$-plane. The inertia related to that axis through the centre of mass we notate as $I_{zz,2}$. The force $\boldsymbol{f}_1$ is producing a momentum in the centre of mass. The vector from the centre of mass to the force application point we notate as $\boldsymbol{r}_{Sp2}$.

$$\boldsymbol{r}_{Sp2} = \left[-l_{SP2} \cdot \sin(q_2 - 90°) \quad 0 \quad -l_{SP2} \cdot \cos(q_2 - 90°)\right]^{\mathrm{T}} = \left[l_{SP2} \cdot \cos(q_2) \quad 0 \quad -l_{SP2} \cdot \sin(q_2)\right]^{\mathrm{T}} \tag{A6}$$

If we additionally consider a velocity dependent friction loss in direction of the axis and denominate the momentum vector which is produced by link 1 and acts upon link 2 as $\boldsymbol{n}_1$, we get:

$$\begin{pmatrix} 0 \\ I_{zz,2}\cdot\ddot{q}_2 \\ 0 \end{pmatrix} + \begin{pmatrix} 0 \\ F_{D,2}\cdot\dot{q}_2 \\ 0 \end{pmatrix} = \boldsymbol{r}_{SP2}\times\boldsymbol{f}_1 + \boldsymbol{n}_1 \rightarrow \begin{pmatrix} 0 \\ I_{zz,2}\cdot\ddot{q}_2 + F_{D,2}\cdot\dot{q}_2 \\ 0 \end{pmatrix} = \begin{pmatrix} 0 \\ r_{Sp2,z}\cdot f_{1x} - r_{Sp2,x}\cdot f_{1z} \\ 0 \end{pmatrix} + \boldsymbol{n}_1$$

$$\boldsymbol{n}_1 = \begin{pmatrix} 0 \\ I_{zz,2}\cdot\ddot{q}_2 + F_{D,2}\cdot\dot{q}_2 - r_{Sp2,z}\cdot f_{1x} + r_{Sp2,x}\cdot f_{1z} \\ 0 \end{pmatrix} \qquad \text{(A7)}$$

If we use eq. (A6) it holds:

$$-r_{Sp2,z}\cdot f_{1x} + r_{Sp2,x}\cdot f_{1z} =$$

$$m_2\cdot l_{SP2}\cdot\left(\sin q_2\cdot\mathrm{g} + l_{SP2}\cdot\cos q_2\cdot\sin q_2\cdot\dot{q}_2^2 + l_{SP2}\cdot\sin^2 q_2\cdot\ddot{q}_2 + \cos q_2\cdot\ddot{q}_1 - l_{SP2}\cdot\cos q_2\cdot\sin q_2\cdot\dot{q}_2^2 + l_{SP2}\cdot\cos^2 q_2\cdot\ddot{q}_2\right) =$$

$$m_2\cdot l_{SP2}\cdot\left(\sin q_2\cdot\mathrm{g} + \cos q_2\cdot\ddot{q}_1 + l_{SP2}\cdot\ddot{q}_2\right)$$

$$\boldsymbol{n}_1 = \begin{pmatrix} 0 \\ \left(I_{zz,2} + m_2\cdot l_{SP2}^2\right)\cdot\ddot{q}_2 + F_{D,2}\cdot\dot{q}_2 + m_2\cdot l_{SP2}\cdot\left(\sin q_2\cdot\mathrm{g} + \cos q_2\cdot\ddot{q}_1\right) \\ 0 \end{pmatrix} \qquad \text{(A8)}$$

The component n_{1y} acts in direction of joint 2, it must be produced by the actuator of joint 2, hence it holds:

$$\tau_2 = n_{1y}: \quad \tau_2 = \left(I_{zz,2} + m_2\cdot l_{SP2}^2\right)\cdot\ddot{q}_2 + F_{D,2}\cdot\dot{q}_2 + m_2\cdot l_{SP2}\cdot\left(\sin q_2\cdot\mathrm{g} + \cos q_2\cdot\ddot{q}_1\right) \qquad \text{(A9)}$$

By applying the free body diagram to link 1 we must consider that the force $-\boldsymbol{f}_1$ is acting on link 1 (Newtons law action=reaction) and a force $\boldsymbol{f}_0$ is acting on link 1 caused by the actuator of joint 1 and the bearings. As well as for joint 2 we take in account a velocity dependent friction loss in direction of the joint axis (z_0-direction) and we get.

$$m_1\cdot\ddot{\boldsymbol{p}}_{s1} + \begin{bmatrix} 0 \\ 0 \\ \hat{F}_{D,1}\cdot\dot{q}_1 \end{bmatrix} = m_1\cdot\mathbf{g} - \boldsymbol{f}_1 + \boldsymbol{f}_0 \rightarrow \begin{bmatrix} 0 \\ 0 \\ m_1\cdot\ddot{q}_1 + \hat{F}_{D,1}\cdot\dot{q}_1 \end{bmatrix} = \begin{bmatrix} -m_1\cdot\mathrm{g} \\ 0 \\ 0 \end{bmatrix} - \begin{bmatrix} f_{1x} \\ 0 \\ f_{1z} \end{bmatrix} + \begin{bmatrix} f_{0,x} \\ 0 \\ f_{0,z} \end{bmatrix}$$

(A10)

The component $f_{0,z}$ is produce by the actuator of joint 1 therefore it holds $\tau_1 = f_{0,z}$:

$$\tau_1 = f_{0,z} = m_1\cdot\ddot{q}_1 + \hat{F}_{D,1}\cdot\dot{q}_1 + f_{1,z} \rightarrow$$
$$\tau_2 = \left(m_1 + m_2\right)\cdot\ddot{q}_1 + \hat{F}_{D,1}\cdot\dot{q}_1 - m_2\cdot l_{SP2}\cdot\sin\left(q_2\right)\cdot\dot{q}_2^2 + m_2\cdot l_{SP2}\cdot\cos\left(q_2\right)\cdot\ddot{q}_2 \qquad \text{(A11)}$$

Now we have solved the problem of the inverse model and we can give equation (A11) and (A9) in matrix form:

$$\boldsymbol{\tau} = \boldsymbol{M}(\boldsymbol{q})\cdot\ddot{\boldsymbol{q}} + \boldsymbol{b}(\boldsymbol{q},\dot{\boldsymbol{q}}) \qquad \text{(A12)}$$

with

$$\tau_1 = M_{11} \cdot \ddot{q}_1 + M_{12} \cdot \ddot{q}_2 + b_1(\boldsymbol{q}, \dot{\boldsymbol{q}})$$
$$\tau_2 = M_{21} \cdot \ddot{q}_1 + M_{22} \cdot \ddot{q}_2 + b_2(\boldsymbol{q}, \dot{\boldsymbol{q}})$$
$$\boldsymbol{M}(\boldsymbol{q}) = \begin{bmatrix} M_{11} & M_{12} \\ M_{21} & M_{22} \end{bmatrix} = \begin{bmatrix} m_1 + m_2 & m_2 \cdot l_{SP2} \cdot \cos(q_2) \\ m_2 \cdot l_{SP2} \cdot \cos(q_2) & I_{zz,2} + m_2 \cdot l_{SP2}^2 \end{bmatrix}$$
$$\boldsymbol{b}(\boldsymbol{q}, \dot{\boldsymbol{q}}) = \begin{bmatrix} b_1(\boldsymbol{q}, \dot{\boldsymbol{q}}) \\ b_2(\boldsymbol{q}, \dot{\boldsymbol{q}}) \end{bmatrix} = \begin{bmatrix} \hat{F}_{D,1} \cdot \dot{q}_1 - m_2 \cdot l_{SP2} \cdot \sin(q_2) \cdot \dot{q}_2^2 \\ F_{D,2} \cdot \dot{q}_2 + m_2 \cdot l_{SP2} \cdot \sin q_2 \cdot \mathrm{g} \end{bmatrix} \qquad \text{(A13)}$$

Following equations (A12), (A13) we can establish the equations of motion in matrix form:

$$\ddot{\boldsymbol{q}} = [\boldsymbol{M}(\boldsymbol{q})]^{-1} \cdot [\boldsymbol{\tau} - \boldsymbol{b}(\boldsymbol{q}, \dot{\boldsymbol{q}})]. \qquad \text{(A14)}$$

The effort to establish the model of multi-joint robots with more than 3 links is very high. We need a methodical procedure. One systematic way to establish such models can be the recursive Newton-Euler method.

Appendix B: Derivation of recursive Newton-Euler equations

Kinematical equations

Notice: All vectors are described in its own coordinate systems. It means: $\boldsymbol{\omega}_i = \boldsymbol{\omega}_i^{(i)}$ and so on. The coordinate system K_i is fixed on link *i* (it moves with link *i*). The vectors of the rotational velocities can be added like other vectors. The direction of $\boldsymbol{\omega}_i$ is the direction of the present rotary axis. It holds:

$$\boldsymbol{\omega}_{i+1} = \boldsymbol{\omega}_i + h_{i+1} \cdot \boldsymbol{z} \cdot \dot{q}_{i+1}$$

To get the vector of rotational accelerations we derive the vector of rotational velocity:

$$\dot{\boldsymbol{\omega}}_{i+1} = \dot{\boldsymbol{\omega}}_i + \boldsymbol{\omega}_i \times \boldsymbol{\omega}_i + h_{i+1} \cdot \left(\boldsymbol{z} \cdot \ddot{q}_{i+1} + \boldsymbol{\omega}_i \times \boldsymbol{z} \cdot \dot{q}_{i+1} \right)$$

The cross product of a vector with it self is the zero-vector and we get:

$$\dot{\boldsymbol{\omega}}_{i+1} = \dot{\boldsymbol{\omega}}_i + h_{i+1} \cdot \left(\boldsymbol{z} \cdot \ddot{q}_{i+1} + \boldsymbol{\omega}_i \times \boldsymbol{z} \cdot \dot{q}_{i+1} \right).$$

The linear velocity of the origin of K_i is given, when we derivate the vector $\boldsymbol{r}_{i+1}$ (see script or Fig. B1 in this paper). Denote that $\boldsymbol{p}_{i+1}$ is constant in its own coordinate system K_{i+1} if joint *i*+1 is a revolute joint ($h_{i+1} = 1$)

$$\boldsymbol{r}_{i+1} = \boldsymbol{r}_i + \boldsymbol{p}_{i+1}$$

$$\boldsymbol{v}_{i+1} = \frac{d}{dt}\boldsymbol{r}_{i+1} = \frac{d}{dt}\boldsymbol{r}_i + \frac{d}{dt}(\boldsymbol{p}_{i+1}) = v_i + \frac{d^{(i+1)}}{dt}(\boldsymbol{p}_{i+1}^{(i+1)}) + \boldsymbol{\omega}_{i+1} \times \boldsymbol{p}_{i+1}$$

$$\boldsymbol{v}_{i+1} = \boldsymbol{v}_i + (1 - h_{i+1}) \cdot \frac{d^{(i+1)}}{dt}(\boldsymbol{p}_{i+1}^{(i+1)}) + \boldsymbol{\omega}_{i+1} \times \boldsymbol{p}_{i+1}\ .$$

The linear acceleration is the derivation of the linear velocity:

$$\dot{\boldsymbol{v}}_{i+1} = \frac{\mathrm{d}}{\mathrm{d}t}\boldsymbol{v}_{i+1} = \frac{\mathrm{d}}{\mathrm{d}t}\boldsymbol{v}_i + (1 - h_{i+1}) \cdot \left(\frac{\mathrm{d}^{2,(i+1)}}{\mathrm{d}t^2}\boldsymbol{p}_{i+1} + \boldsymbol{\omega}_{i+1} \times \frac{\mathrm{d}^{(i+1)}}{\mathrm{d}t}\boldsymbol{p}_{i+1} \right)$$
$$+ \dot{\boldsymbol{\omega}}_{i+1} \times \boldsymbol{p}_{i+1} + \boldsymbol{\omega}_{i+1} \times (\boldsymbol{\omega}_{i+1} \times \boldsymbol{p}_{i+1}) + \boldsymbol{\omega}_{i+1} \times \frac{\mathrm{d}^{(i+1)}}{\mathrm{d}t}\boldsymbol{p}_{i+1}$$

$$\dot{\boldsymbol{v}}_{i+1} = \frac{\mathrm{d}}{\mathrm{d}t}\boldsymbol{v}_{i+1} = \frac{\mathrm{d}}{\mathrm{d}t}\boldsymbol{v}_i + (1 - h_{i+1}) \cdot \left(\frac{\mathrm{d}^{2,(i+1)}}{\mathrm{d}t^2}\boldsymbol{p}_{i+1} + \boldsymbol{\omega}_{i+1} \times \frac{\mathrm{d}^{(i+1)}}{\mathrm{d}t}\boldsymbol{p}_{i+1} + \boldsymbol{\omega}_{i+1} \times \frac{\mathrm{d}^{(i+1)}}{\mathrm{d}t}\boldsymbol{p}_{i+1} \right)$$
$$+ \dot{\boldsymbol{\omega}}_{i+1} \times \boldsymbol{p}_{i+1} + \boldsymbol{\omega}_{i+1} \times (\boldsymbol{\omega}_{i+1} \times \boldsymbol{p}_{i+1})$$

We can simplify the equation again and get

$$\dot{\boldsymbol{v}}_{i+1} = \dot{\boldsymbol{v}}_i + \dot{\boldsymbol{\omega}}_{i+1} \times \boldsymbol{p}_{i+1} + \boldsymbol{\omega}_{i+1} \times (\boldsymbol{\omega}_{i+1} \times \boldsymbol{p}_{i+1}) + (1 - h_{i+1}) \cdot \left(\frac{\mathrm{d}^{2,(i+1)}}{\mathrm{d}t^2}\boldsymbol{p}_{i+1} + 2 \cdot \boldsymbol{\omega}_{i+1} \times \frac{\mathrm{d}^{(i+1)}}{\mathrm{d}t}\boldsymbol{p}_{i+1} \right)$$

Later we will need the linear acceleration of the centre of mass. First we calculate the linear velocity of the centre of mass:

$$\boldsymbol{v}_{s,i+1} = \frac{\mathrm{d}}{\mathrm{d}t}(\boldsymbol{r}_{i+1} + \boldsymbol{s}_{i+1}) = \boldsymbol{v}_{i+1} + \frac{\mathrm{d}^{(i+1)}}{\mathrm{d}t}\boldsymbol{s}_{i+1} + \boldsymbol{\omega}_{i+1} \times \boldsymbol{s}_{i+1} = \boldsymbol{v}_{i+1} + \boldsymbol{\omega}_{i+1} \times \boldsymbol{s}_{i+1}$$

Now we derivate the vector $\boldsymbol{v}_{s,i+1}$ due to time:

$$\begin{aligned}
\dot{\boldsymbol{v}}_{s,i+1} &= \dot{\boldsymbol{v}}_{i+1} + \frac{\mathrm{d}}{\mathrm{d}t}(\boldsymbol{\omega}_{i+1} \times \boldsymbol{s}_{i+1}) = \dot{\boldsymbol{v}}_{i+1} + \frac{\mathrm{d}}{\mathrm{d}t}(\boldsymbol{\omega}_{i+1}) \times \boldsymbol{s}_{i+1} + \boldsymbol{\omega}_{i+1} \times \frac{\mathrm{d}}{\mathrm{d}t}\boldsymbol{s}_{i+1} = \\
&\dot{\boldsymbol{v}}_{i+1} + \dot{\boldsymbol{\omega}}_{i+1} \times \boldsymbol{s}_{i+1} + \boldsymbol{\omega}_{i+1} \times \left(\frac{\mathrm{d}^{(i+1)}}{\mathrm{d}t}\boldsymbol{s}_{i+1} + \boldsymbol{\omega}_{i+1} \times \boldsymbol{s}_{i+1}\right) = \\
&\dot{\boldsymbol{v}}_{i+1} + \dot{\boldsymbol{\omega}}_{i+1} \times \boldsymbol{s}_{i+1} + \boldsymbol{\omega}_{i+1} \times (\boldsymbol{\omega}_{i+1} \times \boldsymbol{s}_{i+1})
\end{aligned}$$

For numerical calculation all vectors have to be described in the same coordinate system, here K_{i+1}. Hence we use the rotational matrix ${}_{i+1}^{\;\;i}\boldsymbol{A}$:

$$\begin{aligned}
\boldsymbol{\omega}_{i+1} &= {}_{i+1}^{\;\;i}\boldsymbol{A}(\boldsymbol{\omega}_i + h_{i+1} \cdot \boldsymbol{z} \cdot \dot{q}_{i+1}), \\
\dot{\boldsymbol{\omega}}_{i+1} &= {}_{i+1}^{\;\;i}\boldsymbol{A}[\dot{\boldsymbol{\omega}}_i + h_{i+1} \cdot (\boldsymbol{z} \cdot \ddot{q}_{i+1} + \boldsymbol{\omega}_i \times \boldsymbol{z} \cdot \dot{q}_{i+1})], \\
\boldsymbol{v}_{i+1} &= {}_{i+1}^{\;\;i}\boldsymbol{A} \cdot \boldsymbol{v}_i + (1 - h_{i+1}) \cdot \frac{\mathrm{d}^{(i+1)}}{\mathrm{d}t}\boldsymbol{p}_{i+1} + \boldsymbol{\omega}_{i+1} \times \boldsymbol{p}_{i+1}, \\
\dot{\boldsymbol{v}}_{i+1} &= {}_{i+1}^{\;\;i}\boldsymbol{A} \cdot \dot{\boldsymbol{v}}_i + \dot{\boldsymbol{\omega}}_{i+1} \times \boldsymbol{p}_{i+1} + \boldsymbol{\omega}_{i+1} \times (\boldsymbol{\omega}_{i+1} \times \boldsymbol{p}_{i+1}) + \\
&\quad + (1 - h_{i+1}) \cdot \left(\frac{\mathrm{d}^{2,(i+1)}}{\mathrm{d}t^2}\boldsymbol{p}_{i+1} + 2 \cdot \boldsymbol{\omega}_{i+1} \times \frac{\mathrm{d}^{(i+1)}}{\mathrm{d}t}\boldsymbol{p}_{i+1}\right), \\
\boldsymbol{v}_{s,i+1} &= \boldsymbol{v}_{i+1} + \boldsymbol{\omega}_{i+1} \times \boldsymbol{s}_{i+1}, \\
\dot{\boldsymbol{v}}_{s,i+1} &= \dot{\boldsymbol{v}}_{i+1} + \dot{\boldsymbol{\omega}}_{i+1} \times \boldsymbol{s}_{i+1} + \boldsymbol{\omega}_{i+1} \times (\boldsymbol{\omega}_{i+1} \times \boldsymbol{s}_{i+1})
\end{aligned}$$

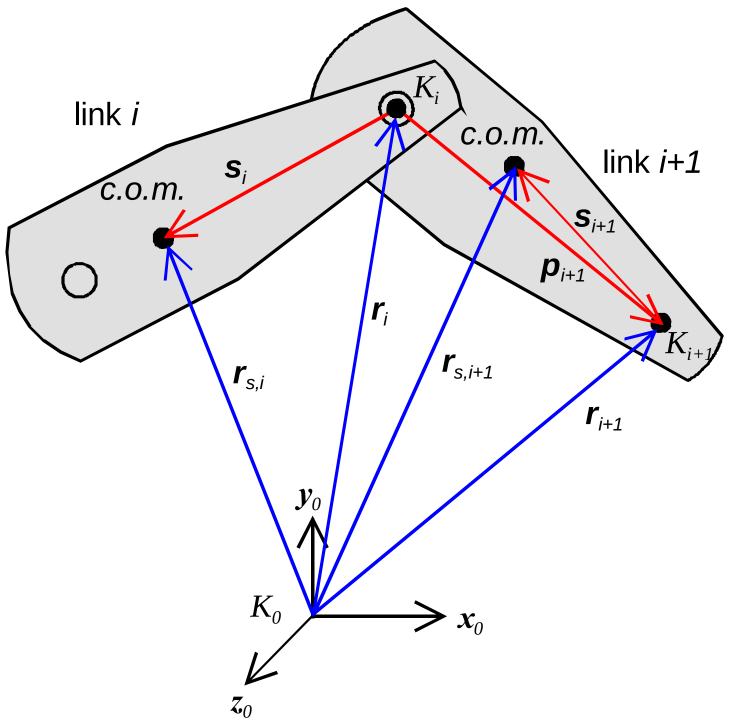


Fig. B1: Kinematical relations between two links and the non moving coordinate system K_0

Momentum/force balance

We consider Fig. B2 and apply the momentum balance to the free body diagram of link *i*

$$\boldsymbol{N}_i = \boldsymbol{n}_i - \boldsymbol{n}_{i+1} + (\boldsymbol{r}_{i-1} - \boldsymbol{r}_{s,i}) \times \boldsymbol{f}_i + (-\boldsymbol{s}_i) \times (-\boldsymbol{f}_{i+1}) = \boldsymbol{n}_i - \boldsymbol{n}_{i+1} + (\boldsymbol{r}_{i-1} - \boldsymbol{r}_{s,i}) \times \boldsymbol{f}_i + \boldsymbol{s}_i \times \boldsymbol{f}_{i+1}$$

With $\boldsymbol{s}_i = \boldsymbol{r}_{s,i} - \boldsymbol{r}_i = \boldsymbol{r}_{s,i} - \boldsymbol{r}_{i-1} - \boldsymbol{p}_i$ it follows for $\boldsymbol{N}_i$:

$$\boldsymbol{N}_i = \boldsymbol{n}_i - \boldsymbol{n}_{i+1} + (\boldsymbol{r}_{i-1} - \boldsymbol{r}_{s,i}) \times \boldsymbol{f}_i - (\boldsymbol{r}_{i-1} - \boldsymbol{r}_{s,i} + \boldsymbol{p}_i) \times \boldsymbol{f}_{i+1} =$$
$$\boldsymbol{n}_i - \boldsymbol{n}_{i+1} + (\boldsymbol{r}_{i-1} - \boldsymbol{r}_{s,i}) \times (\boldsymbol{f}_i - \boldsymbol{f}_{i+1}) - \boldsymbol{p}_i \times \boldsymbol{f}_{i+1}$$

To achieve a recursive formula we put $\boldsymbol{n}_i$ to the left side:

$$\boldsymbol{n}_i = \boldsymbol{n}_{i+1} - (\boldsymbol{r}_{i-1} - \boldsymbol{r}_{s,i}) \times (\boldsymbol{f}_i - \boldsymbol{f}_{i+1}) + \boldsymbol{p}_i \times \boldsymbol{f}_{i+1} + \boldsymbol{N}_i$$

For numerical calculations all vectors must be described in the coordinate system K_i which is fixed on the link *i*:

$$\boldsymbol{F}_i = m_i \cdot \dot{\boldsymbol{v}}_{s,i} + (1-h_i) \cdot \hat{F}_{D,i} \, {}^{i-1}_{i}\boldsymbol{A} \cdot \boldsymbol{z} \cdot \dot{q}_i,$$
$$\boldsymbol{N}_i = \boldsymbol{I}_{\mathrm{SP},i} \cdot \dot{\boldsymbol{\omega}}_i + \boldsymbol{\omega}_i \times (\boldsymbol{I}_{\mathrm{SP},i} \cdot \boldsymbol{\omega}_i) + h_i \cdot F_{D,i} \cdot {}^{i-1}_{i}\boldsymbol{A} \cdot \boldsymbol{z} \cdot \dot{q}_i,$$
$$\boldsymbol{f}_i = {}^{i+1}_{i}\boldsymbol{A} \cdot \boldsymbol{f}_{i+1} + \boldsymbol{F}_i,$$
$$\boldsymbol{n}_i = {}^{i+1}_{i}\boldsymbol{A} \cdot [\boldsymbol{n}_{i+1} + ({}^{i}_{i+1}\boldsymbol{A} \cdot \boldsymbol{p}_i) \times \boldsymbol{f}_{i+1}] + (\boldsymbol{p}_i + \boldsymbol{s}_i) \times \boldsymbol{F}_i + \boldsymbol{N}_i,$$
$$\boldsymbol{\tau}_i = [h_i \cdot \boldsymbol{n}_i^T + (1-h_i) \cdot \boldsymbol{f}_i^T] \cdot {}^{i-1}_{i}\boldsymbol{A} \cdot \boldsymbol{z},$$
$$\text{initial values: } \boldsymbol{f}_{n+1} = -\boldsymbol{f}_{ex}, \boldsymbol{n}_{n+1} = -\boldsymbol{n}_{ex}$$

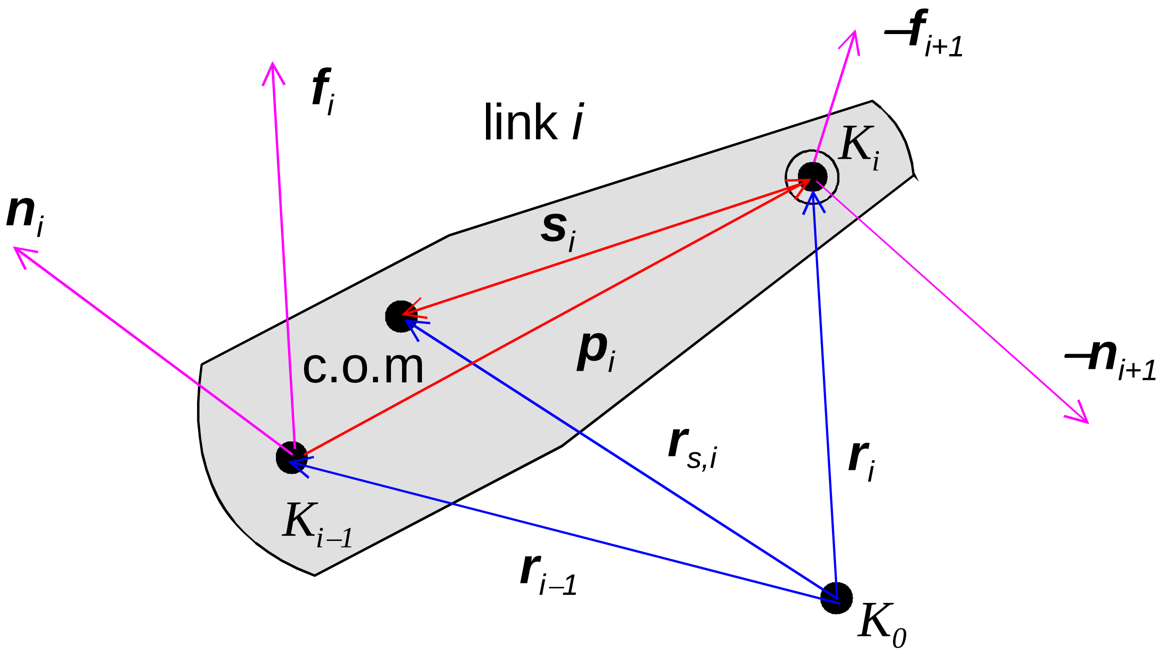


Fig. B2: Free body diagram of link i

Example: Derivation of a vector in a moving coordinate system

D.-H. parameter $\alpha_1=0$, $a_1=l_1$, though ${}_0^1\boldsymbol{A}=\begin{bmatrix}\cos q_1 & -\sin q_1 & 0\\ \sin q_1 & \cos q_1 & 0\\ 0 & 0 & 1\end{bmatrix}$ (see Fig.)

The vector of rotational velocity of link 1 due to non moving base is $\boldsymbol{\omega}_1=\begin{pmatrix}0\\0\\\dot q\end{pmatrix}$

The linear velocity of point A due to non moving base is the derivation of vector $\underline{p}_A$ from origin of K_0 to point A: $\boldsymbol{v}_A=\dfrac{d\,\boldsymbol{p}_A}{dt}$, $\boldsymbol{p}_A=\boldsymbol{r}_1+\boldsymbol{a}$. With $\boldsymbol{r}_1^{(1)}=\begin{pmatrix}\ell_1\\0\\0\end{pmatrix}$ $\boldsymbol{a}^{(1)}=\begin{pmatrix}-\ell_A\\0\\0\end{pmatrix}$ it holds, if all vectors are described in coordinate system 1:

$$\boldsymbol{v}_A^{(1)}=\underbrace{\frac{d\boldsymbol{r}_1^{(1)}}{dt}}_{0}+\boldsymbol{\omega}_1^{(1)}\times\boldsymbol{r}_1^{(1)}+\underbrace{\frac{d\boldsymbol{a}^{(1)}}{dt}}_{0}+\boldsymbol{\omega}_1^{(1)}\times\boldsymbol{a}^{(1)}=\begin{pmatrix}0\\\ell_1\cdot\dot q_1\\0\end{pmatrix}+\begin{pmatrix}0\\-\ell_A\cdot\dot q_1\\0\end{pmatrix}=\begin{pmatrix}0\\(\ell_1-\ell_A)\cdot\dot q_1\\0\end{pmatrix}$$

$$\boldsymbol{v}_A^{(0)}={}_0^1\boldsymbol{A}\cdot\boldsymbol{v}_A^{(1)}=\begin{pmatrix}-\sin q_1\cdot(\ell_1-\ell_A)\cdot\dot q\\\cos q_1\cdot(\ell_1-\ell_A)\cdot\dot q\\0\end{pmatrix}$$

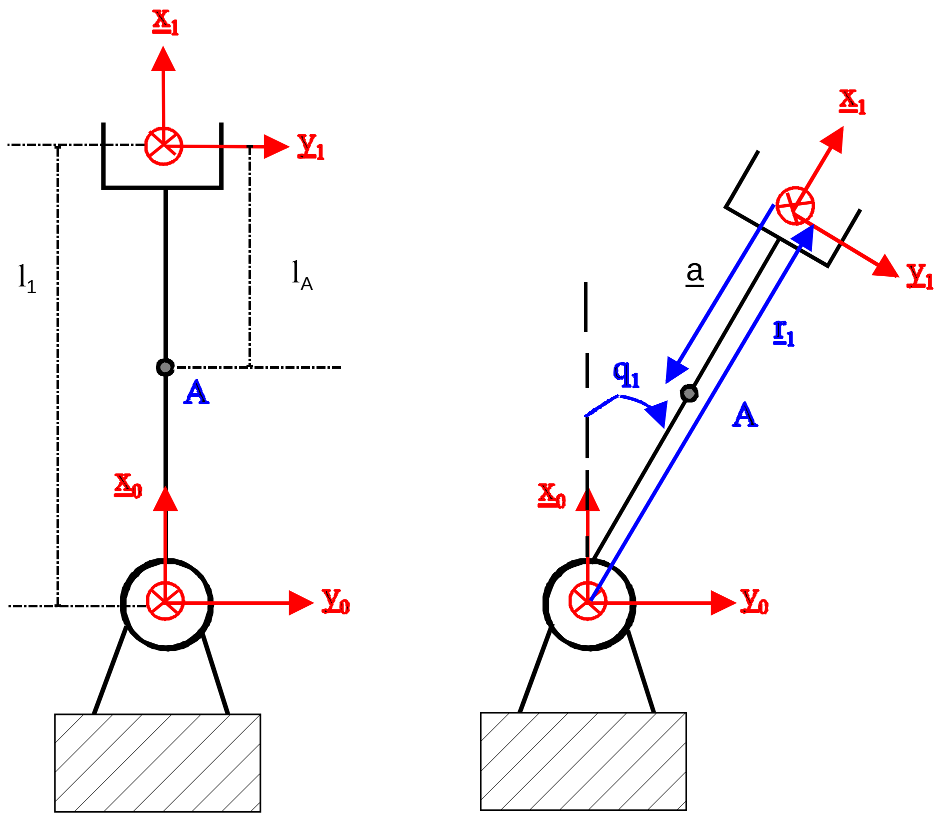


Fig. B3: One joint example for moving coordinate systems

Appendix C: Explicit evaluation of matrix $M^*(q)$ and vector $b^*(q,\dot{q})$

For control purposes we need the inverse model of the whole electromechanics (see Fig. B1, left side). The joint variables are given ore measured and the acting voltages are integrated into the vector $\boldsymbol{U}_S$. The components of $\boldsymbol{U}_S$ are the input signals to the servos of the motors which causes the given or measured motion. From equation (4.16) in script we get

$$\boldsymbol{U}_S = \boldsymbol{K}_M^{-1} \cdot \boldsymbol{S}_G^{-1} \cdot \boldsymbol{\tau} + \boldsymbol{K}_M^{-1} \cdot \left[\boldsymbol{J}_A \cdot \boldsymbol{T}_G^{-1} \cdot \ddot{\boldsymbol{q}} + \boldsymbol{M}_R(\dot{\boldsymbol{q}}) \right]. \qquad \text{(C1)}$$

At a point of time t_k the vectors $\boldsymbol{q}(t_k), \dot{\boldsymbol{q}}(t_k), \ddot{\boldsymbol{q}}(t_k)$ are given and $\boldsymbol{\tau}(t_k)$ can be evaluated as numerical vector with the recursive Newton-Euler-Equation (eq. 4.1, 4.2 in script). Then the equation (C1) has to be evaluated and we get the numerical vector $\boldsymbol{U}_S(t_k)$.

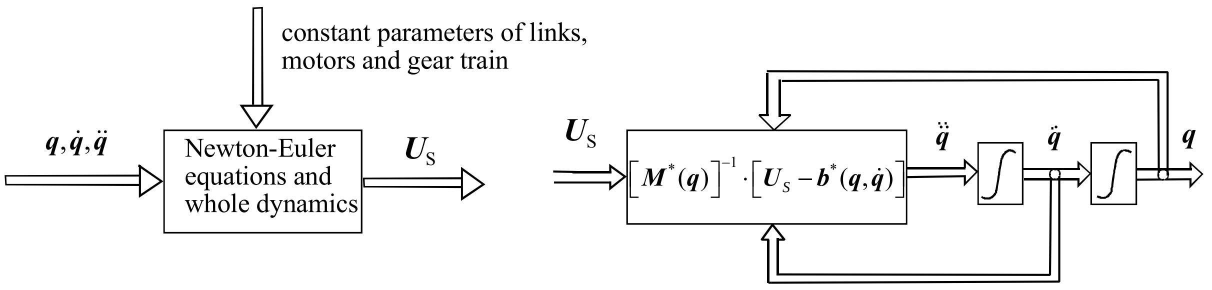

constant parameters of links, motors and gear train
$q,\dot{q},\ddot{q}$
Newton-Euler equations and whole dynamics
U_S
U_S
$\left[M^*(q)\right]^{-1} \cdot \left[U_S - b^*(q,\dot{q})\right]$
$\ddot{q}$
$\dot{q}$
q

Fig. C1: Inverse model (left side) and equations of motion (right sight) of the whole dynamics of the robot system

If we have to form the equation of motion, the input vector is $\boldsymbol{U}_S$ and the output is the motion which is caused by $\boldsymbol{U}_S$ (see Fig. C1, right side). The equation of motion is for example necessary to simulate the control performance of a robot system. In lecture and script we have developed the equation of motion in the form of (4.17) of script

$$\ddot{\boldsymbol{q}} = \left[\boldsymbol{M}^*(\boldsymbol{q})\right]^{-1} \cdot \left[\boldsymbol{U}_S - \boldsymbol{b}^*(\boldsymbol{q},\dot{\boldsymbol{q}})\right] \quad \text{with}$$
$$\boldsymbol{M}^*(\boldsymbol{q}) = \boldsymbol{K}_M^{-1} \cdot \left[\boldsymbol{S}_G^{-1} \cdot \boldsymbol{M}(\boldsymbol{q}) + \boldsymbol{J}_A \cdot \ddot{\boldsymbol{q}}\right], \quad \boldsymbol{b}^*(\boldsymbol{q},\dot{\boldsymbol{q}}) = \boldsymbol{K}_M^{-1} \cdot \left[\boldsymbol{S}_G^{-1} \cdot \boldsymbol{b}(\boldsymbol{q},\dot{\boldsymbol{q}}) + \boldsymbol{M}_R(\dot{\boldsymbol{q}})\right] \qquad \text{(C2)}$$

Our problem is that the inertia matrix $\boldsymbol{M}(\boldsymbol{q})$ and the vector $\boldsymbol{b}(\boldsymbol{q},\dot{\boldsymbol{q}})$ are not explicitly given by applying the Newton-Euler-equations. All other matrices in (C2) are constant and given and the vector $\boldsymbol{M}_R(\dot{\boldsymbol{q}})$ is explicitly given too. From the evaluation of the Newton-Euler-equation we only know the numerical values of the vector $\boldsymbol{\tau}$ of the inverse model at every point of time. The explicit form

$$\boldsymbol{\tau} = \boldsymbol{M}(\boldsymbol{q}) \cdot \ddot{\boldsymbol{q}} + \boldsymbol{b}(\boldsymbol{q},\dot{\boldsymbol{q}}) \qquad \text{(C3)}$$

is unknown. But in the following it is shown, that we can evaluate $\boldsymbol{M}(\boldsymbol{q})$ and $\boldsymbol{b}(\boldsymbol{q},\dot{\boldsymbol{q}})$ explicitly in numerical form for any point of time with the recursive Newton-Euler-Equations. This method is first published in *Walker, M.W.; Orin, D.E.: Efficient Dynamic Computer Simulation of Robotic Mechanisms. Journ. Dynamic Systems, Measurement, Control 104(1982); S. 205-211.*
First we divide the vector $\boldsymbol{b}$ into several parts and we get for (C3):

$$\boldsymbol{\tau} = \boldsymbol{M}(\boldsymbol{q}) \cdot \ddot{\boldsymbol{q}} + \boldsymbol{G}(\boldsymbol{q}) + \boldsymbol{C}(\boldsymbol{q},\dot{\boldsymbol{q}}) + \boldsymbol{F}_D \cdot \dot{\boldsymbol{q}} \,. \tag{C4}$$

The vector $\boldsymbol{G}(\boldsymbol{q})$ is the vector caused by gravity, the vector $\boldsymbol{C}(\boldsymbol{q},\dot{\boldsymbol{q}})$ describes Coriolis- and Centripetal forces/moments, while the matrix $\boldsymbol{F}_D$ contains the coefficients of velocity dependent friction. The vector $\boldsymbol{C}(\boldsymbol{q},\dot{\boldsymbol{q}})$ has the form

$$\boldsymbol{C}(\boldsymbol{q},\dot{\boldsymbol{q}}) = \boldsymbol{D}(\boldsymbol{q}) \cdot \boldsymbol{f}_C(\dot{\boldsymbol{q}}) \tag{C5}$$

Is n the number of joints, $\boldsymbol{D}$ is a nxn_c-matrix, with $n_c = n \cdot (n+1)/2$. The components of the vector $\boldsymbol{f}_C$ are all combinations of 2nd order of the joint velocities without considering the sequence:

$$f_{c,1} = \dot{q}_1^2 \,,\; f_{c,2} = \dot{q}_1 \cdot \dot{q}_2 \,, \cdots ,\; f_{c,n_c} = \dot{q}_n^2 \,.$$

Now we use the principle that $\boldsymbol{\tau}$ is the sum of different physical effects. If we evaluate $\boldsymbol{\tau}$ with the recursive Newton-Euler-equations with given or measured joint coordinates $\boldsymbol{q}(t_k)$ and joint acceleration vector $\ddot{\boldsymbol{q}}(t_k)$ at the point in time t_k and set $\dot{\boldsymbol{q}} = \boldsymbol{0}$ and assume the value g = 0 for the the gravity constant, we get in this case from (C4):

$$\boldsymbol{\tau}(t_k) = \boldsymbol{M}(\boldsymbol{q}(t_k)) \cdot \ddot{\boldsymbol{q}}(t_k) \tag{C6}$$

Nevertheless we have only got the numerical value of $\boldsymbol{\tau}(t_k)$ and not $\boldsymbol{M}(\boldsymbol{q}(t_k))$ in an explicit form. But we can use the upper principle again and say $\boldsymbol{\tau}(t_k)$ is the sum of the influence of the acceleration of each joint. We can write (C6) in the detailed form

$$\begin{aligned}
\tau_1(t_k) &= M_{11}(\boldsymbol{q}(t_k)) \cdot \ddot{q}_1(t_k) + M_{12}(\boldsymbol{q}(t_k)) \cdot \ddot{q}_2(t_k) + \cdots + M_{1n}(\boldsymbol{q}(t_k)) \cdot \ddot{q}_n(t_k) \\
\tau_2(t_k) &= M_{21}(\boldsymbol{q}(t_k)) \cdot \ddot{q}_1(t_k) + M_{22}(\boldsymbol{q}(t_k)) \cdot \ddot{q}_2(t_k) + \cdots + M_{2n}(\boldsymbol{q}(t_k)) \cdot \ddot{q}_n(t_k) \\
&\vdots \\
\tau_n(t_k) &= M_{n1}(\boldsymbol{q}(t_k)) \cdot \ddot{q}_1(t_k) + M_{n2}(\boldsymbol{q}(t_k)) \cdot \ddot{q}_2(t_k) + \cdots + M_{nn}(\boldsymbol{q}(t_k)) \cdot \ddot{q}_n(t_k)
\end{aligned} \tag{C7}$$

If we now calculate $\boldsymbol{\tau}(t_k)$ with the Newton-Euler equations and set, $\ddot{q}_1(t_k) = 1, \ddot{q}_2(t_k) = \ddot{q}_3(t_k) = \cdots \ddot{q}_n(t_k) = 0$ the Newton-Euler equations produce:

$$\begin{aligned}
\tau_1(t_k) &= M_{11}(\boldsymbol{q}(t_k)) \\
\tau_2(t_k) &= M_{21}(\boldsymbol{q}(t_k)) \\
&\vdots \\
\tau_n(t_k) &= M_{n1}(\boldsymbol{q}(t_k))
\end{aligned} \tag{C8}$$

For the point in time t_k the first column of the Matrix $\boldsymbol{M}$ is given. If we define $\ddot{q}_2(t_k) = 1, \ddot{q}_1(t_k) = \ddot{q}_3(t_k) = \cdots \ddot{q}_n(t_k) = 0$, the evaluation of Newton-Euler equations results in the second column of the inertia matrix. In this way all columns of $\boldsymbol{M}$ can be evaluated. To attain the vector $\boldsymbol{b}(\boldsymbol{q},\dot{\boldsymbol{q}})$ in explicit form for point in time t_k we define $\ddot{\boldsymbol{q}}(t_k) = \boldsymbol{0}$, the gravity constant g = 9.81m/s^2 and $\dot{\boldsymbol{q}}(t_k)$ to the given or measured values. How (C3) shows, Newton-Euler-equations produce

$$\boldsymbol{\tau}(t_k) = \boldsymbol{b}\big(\boldsymbol{q}(t_k), \dot{\boldsymbol{q}}(t_k)\big) \tag{C9}$$

In this way we can calculate $\boldsymbol{M}$ and $\boldsymbol{b}$ for each point in time and we can evaluate equation (C2).

Appendix D: Feedback Control with ReduS- Controller

Additionally to the PI-controller the ReDuS control system is proposed for a plant of second order(Fig. C1). The idea of the ReDuS controller is to choose the free parameters α, β, K_I, that a reference transfer function between v_d and v is achieved.

For the plant parameters must be valid:

$$\frac{V(s)}{U(s)} = \frac{1}{a_0 + a_1 \cdot s + a_2 \cdot s^2}; \quad a_0 \geq 0,\ a_1 > 0,\ a_2 \geq 0 \tag{D1}$$

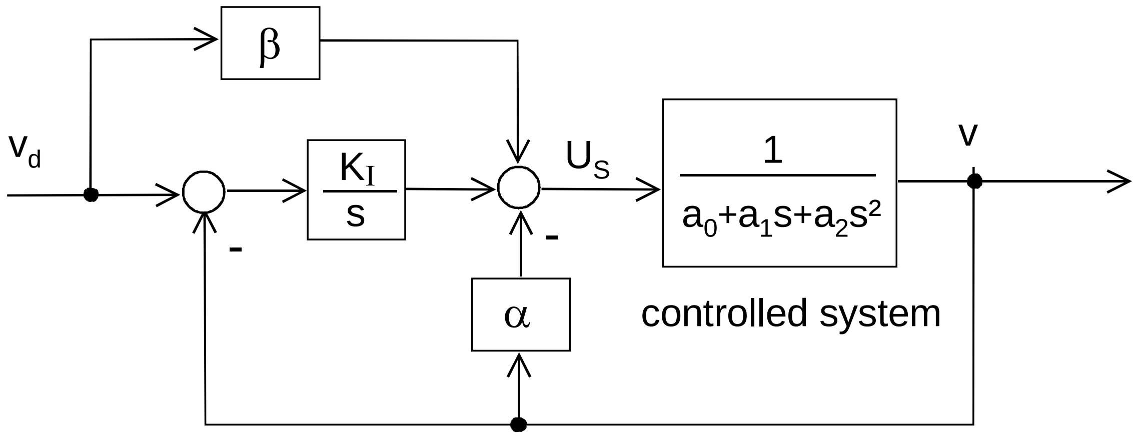


Fig. D1. Structure of ReDuS-Controller.

The application engineer selects the damping parameter d_R and the time constant T_R of a desired transfer function between v_d and v:

$$V(s) = \frac{1}{1 + 2 \cdot d_R \cdot T_R \cdot s + T_R^2 \cdot s^2} \cdot V_d(s) \tag{D2}$$

On the other hand the real transfer function between v_d and v can be calculated on the base of Fig. D1 to:

$$V(s) = \frac{K_I + \beta \cdot s}{K_I + (\alpha + a_0) \cdot s + a_1 \cdot s^2 + a_2 \cdot s^3} \cdot V_d(s) \tag{D3}$$

Equation (D2) and equation (D3) have to be identical:

$$\frac{K_I + \beta \cdot s}{K_I + (\alpha + a_0) \cdot s + a_1 \cdot s^2 + a_2 \cdot s^3} = \frac{1}{1 + 2 \cdot d_R \cdot T_R \cdot s + T_R^2 \cdot s^2}$$

$$K_I + (\beta + 2 \cdot K_I \cdot d_R \cdot T_R) \cdot s + (K_I \cdot T_R^2 + 2 \cdot \beta \cdot d_R \cdot T_R) \cdot s^2 + \beta \cdot T_R^2 \cdot s^3 = K_I + (\alpha + a_0) \cdot s + a_1 \cdot s^2 + a_2 \cdot s^3$$

With this fact the design rules for the three control parameters α, β, K_I are as follows

$$\beta = \frac{a_2}{T_R^2}, \; K_I = \frac{a_1 - 2 \cdot d_R \cdot T_R \cdot \beta}{T_R^2}, \; \alpha = 2 \cdot d_R \cdot T_R \cdot K_I + \beta - a_0 \tag{D4}$$

The control parameters depend on T_R, d_R and depend on the plant parameters a_0, a_1, a_2.
The desired values T_R, d_R must satisfy the stability condition:

$$T_R \cdot a_1 - 2 \cdot d_R \cdot a_2 > 0 \tag{D5}$$